Nyoka Documentation

Release 3.1.0

maintainer

Jul 03, 2020

Contents

CHAPTER 1

PMML44 module

CHAPTER 2

PMML44Super module

class PMML44Super.**AR**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='AR'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='AR'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='AR'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='AR'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='AR'*, *mapping_=None*)

class PMML44Super.**ARDSquaredExponentialKernel**(*description=None*, *gamma='1'*, *noiseVariance='1'*, *Extension=None*, *Lambda=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Lambda(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ARDSquaredExponentialKernel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ARDSquaredExponentialKernel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ARDSquaredExponentialKernel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ARDSquaredExponentialKernel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Lambda()

get_description()

get_gamma()

get_noiseVariance()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Lambda_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Lambda_at(*index*, *value*)

set_Extension(*Extension*)

set_Lambda(*Lambda*)

set_description(*description*)

set_gamma(*gamma*)

set_noiseVariance(*noiseVariance*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ARDSquaredExponentialKernel'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ARIMA**(*RMSE=None*, *transformation='none'*, *constantTerm='0'*, *predictionMethod='conditionalLeastSquares'*, *Extension=None*, *NonseasonalComponent=None*, *SeasonalComponent=None*, *DynamicRegressor=None*, *MaximumLikelihoodStat=None*, *OutlierEffect=None*)

Bases: PMML44Super.GeneratedsSuper

add_DynamicRegressor(*value*)

add_Extension(*value*)

add_OutlierEffect(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ARIMA'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ARIMA'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ARIMA'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ARIMA'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DynamicRegressor()

get_Extension()

get_MaximumLikelihoodStat()

get_NonseasonalComponent()

get_OutlierEffect()

get_RMSE()

get_SeasonalComponent()

get_constantTerm()

get_predictionMethod()

get_transformation()

hasContent_()

insert_DynamicRegressor_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_OutlierEffect_at(*index*, *value*)

replace_DynamicRegressor_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_OutlierEffect_at(*index*, *value*)

set_DynamicRegressor(*DynamicRegressor*)

set_Extension(*Extension*)

set_MaximumLikelihoodStat(*MaximumLikelihoodStat*)

set_NonseasonalComponent(*NonseasonalComponent*)

set_OutlierEffect(*OutlierEffect*)

set_RMSE(*RMSE*)

set_SeasonalComponent(*SeasonalComponent*)

set_constantTerm(*constantTerm*)

set_predictionMethod(*predictionMethod*)

set_transformation(*transformation*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ARIMA'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ARMAPart**(*constant='0'*, *p=None*, *q=None*, *Extension=None*, *AR=None*, *MA=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ARMAPart'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ARMAPart'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ARMAPart'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ARMAPart'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AR()

get_Extension()

get_MA()

get_constant()

get_p()

get_q()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_AR(*AR*)

set_Extension(*Extension*)

set_MA(*MA*)

set_constant(*constant*)

set_p(*p*)

set_q(*q*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ARMAPart'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**AbsoluteExponentialKernel**(*description=None*, *gamma='1'*, *noiseVariance='1'*, *Extension=None*, *Lambda=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Lambda(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='AbsoluteExponentialKernel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='AbsoluteExponentialKernel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='AbsoluteExponentialKernel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='AbsoluteExponentialKernel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Lambda()

get_description()

get_gamma()

get_noiseVariance()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Lambda_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Lambda_at(*index*, *value*)

set_Extension(*Extension*)

set_Lambda(*Lambda*)

set_description(*description*)

set_gamma(*gamma*)

set_noiseVariance(*noiseVariance*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='AbsoluteExponentialKernel'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Adadelta**(*learningRate=None*, *rho=None*, *decayRate=None*, *epsilon=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Adadelta'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Adadelta'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Adadelta'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Adadelta'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_decayRate()

get_epsilon()

get_learningRate()

get_rho()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_decayRate(*decayRate*)

set_epsilon(*epsilon*)

set_learningRate(*learningRate*)

set_rho(*rho*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Adadelta'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Adagrad**(*learningRate=None*, *decayRate=None*, *epsilon=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Adagrad'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Adagrad'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Adagrad'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Adagrad'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_decayRate()

get_epsilon()

get_learningRate()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_decayRate(*decayRate*)

set_epsilon(*epsilon*)

set_learningRate(*learningRate*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Adagrad'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Adam**(*learningRate=None*, *beta_1=None*, *beta_2=None*, *decayRate=None*, *epsilon=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Adam'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Adam'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Adam'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Adam'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_beta_1()

get_beta_2()

get_decayRate()

get_epsilon()

get_learningRate()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_beta_1(*beta_1*)

set_beta_2(*beta_2*)

set_decayRate(*decayRate*)

set_epsilon(*epsilon*)

set_learningRate(*learningRate*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Adam'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Adamax**(*learningRate=None*, *beta_1=None*, *beta_2=None*, *decayRate=None*, *epsilon=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Adamax'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Adamax'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Adamax'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Adamax'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_beta_1()

get_beta_2()

get_decayRate()

get_epsilon()

get_learningRate()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_beta_1(*beta_1*)

set_beta_2(*beta_2*)

set_decayRate(*decayRate*)

set_epsilon(*epsilon*)

set_learningRate(*learningRate*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Adamax'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Aggregate**(*field=None*, *function=None*, *groupField=None*, *sqlWhere=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Aggregate'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Aggregate'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Aggregate'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Aggregate'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_field()

get_function()

get_groupField()

get_sqlWhere()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_field(*field*)

set_function(*function*)

set_groupField(*groupField*)

set_sqlWhere(*sqlWhere*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Aggregate'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**Alternate**(*AnyDistribution=None*, *GaussianDistribution=None*, *PoissonDistribution=None*, *UniformDistribution=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Alternate'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Alternate'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Alternate'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Alternate'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AnyDistribution()

get_GaussianDistribution()

get_PoissonDistribution()

get_UniformDistribution()

hasContent_()

set_AnyDistribution(*AnyDistribution*)

set_GaussianDistribution(*GaussianDistribution*)

set_PoissonDistribution(*PoissonDistribution*)

set_UniformDistribution(*UniformDistribution*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Alternate'*, *mapping_=None*)

class PMML44Super.**Annotation**(*Extension=None*, *valueOf_=None*, *mixedclass_=None*, *content_=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Annotation'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Annotation'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Annotation'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Annotation'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_valueOf_()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_valueOf_(*valueOf_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Annotation'*, *mapping_=None*)

class PMML44Super.**AnomalyDetectionModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *algorithmType=None*, *MiningSchema=None*, *Output=None*, *LocalTransformations=None*, *ParameterList=None*, *ModelVerification=None*, *AssociationModel=None*, *BayesianNetworkModel=None*, *BaselineModel=None*, *ClusteringModel=None*, *DeepNetwork=None*, *AnomalyDetectionModel_member=None*, *GaussianProcessModel=None*, *GeneralRegressionModel=None*, *MiningModel=None*, *NaiveBayesModel=None*, *NearestNeighborModel=None*, *NeuralNetwork=None*, *RegressionModel=None*, *RuleSetModel=None*, *SequenceModel=None*, *Scorecard=None*, *SupportVectorMachineModel=None*, *TextModel=None*, *TimeSeriesModel=None*, *TreeModel=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='AnomalyDetectionModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='AnomalyDetectionModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='AnomalyDetectionModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='AnomalyDetectionModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AnomalyDetectionModel()

get_AssociationModel()

get_BaselineModel()

get_BayesianNetworkModel()

get_ClusteringModel()

get_DeepNetwork()

get_Extension()

get_GaussianProcessModel()

get_GeneralRegressionModel()

get_LocalTransformations()

get_MiningModel()

get_MiningSchema()

get_ModelVerification()

get_NaiveBayesModel()

get_NearestNeighborModel()

get_NeuralNetwork()

get_Output()

get_ParameterList()

get_RegressionModel()

get_RuleSetModel()

get_Scorecard()

get_SequenceModel()

get_SupportVectorMachineModel()

get_TextModel()

get_TimeSeriesModel()

get_TreeModel()

get_algorithmName()

get_algorithmType()

get_functionName()

get_modelName()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_AnomalyDetectionModel(*AnomalyDetectionModel*)

set_AssociationModel(*AssociationModel*)

set_BaselineModel(*BaselineModel*)

set_BayesianNetworkModel(*BayesianNetworkModel*)

set_ClusteringModel(*ClusteringModel*)

set_DeepNetwork(*DeepNetwork*)

set_Extension(*Extension*)

set_GaussianProcessModel(*GaussianProcessModel*)

set_GeneralRegressionModel(*GeneralRegressionModel*)

set_LocalTransformations(*LocalTransformations*)

set_MiningModel(*MiningModel*)

set_MiningSchema(*MiningSchema*)

set_ModelVerification(*ModelVerification*)

set_NaiveBayesModel(*NaiveBayesModel*)

set_NearestNeighborModel(*NearestNeighborModel*)

set_NeuralNetwork(*NeuralNetwork*)

set_Output(*Output*)

set_ParameterList(*ParameterList*)

set_RegressionModel(*RegressionModel*)

set_RuleSetModel(*RuleSetModel*)

set_Scorecard(*Scorecard*)

set_SequenceModel(*SequenceModel*)

set_SupportVectorMachineModel(*SupportVectorMachineModel*)

set_TextModel(*TextModel*)

set_TimeSeriesModel(*TimeSeriesModel*)

set_TreeModel(*TreeModel*)

set_algorithmName(*algorithmName*)

set_algorithmType(*algorithmType*)

set_functionName(*functionName*)

set_modelName(*modelName*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='AnomalyDetectionModel'*, *mapping_=None*)

validate_ALGORITHM_TYPE(*value*)

validate_MINING_FUNCTION(*value*)

class PMML44Super.**Anova**(*target=None*, *Extension=None*, *AnovaRow=None*)

Bases: PMML44Super.GeneratedsSuper

add_AnovaRow(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Anova'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Anova'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Anova'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Anova'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AnovaRow()

get_Extension()

get_target()

hasContent_()

insert_AnovaRow_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_AnovaRow_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_AnovaRow(*AnovaRow*)

set_Extension(*Extension*)

set_target(*target*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Anova'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**AnovaRow**(*type_=None*, *sumOfSquares=None*, *degreesOfFreedom=None*, *meanOfSquares=None*, *fValue=None*, *pValue=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='AnovaRow'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='AnovaRow'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='AnovaRow'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='AnovaRow'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_degreesOfFreedom()

get_fValue()

get_meanOfSquares()

get_pValue()

get_sumOfSquares()

get_type()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_degreesOfFreedom(*degreesOfFreedom*)

set_fValue(*fValue*)

set_meanOfSquares(*meanOfSquares*)

set_pValue(*pValue*)

set_sumOfSquares(*sumOfSquares*)

set_type(*type_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='AnovaRow'*, *mapping_=None*)

validate_NUMBER(*value*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**AntecedentSequence**(*Extension=None*, *SequenceReference=None*, *Time=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='AntecedentSequence'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='AntecedentSequence'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='AntecedentSequence'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='AntecedentSequence'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_SequenceReference()

get_Time()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_SequenceReference(*SequenceReference*)

set_Time(*Time*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='AntecedentSequence'*, *mapping_=None*)

class PMML44Super.**AnyDistribution**(*mean=None*, *variance=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='AnyDistribution'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='AnyDistribution'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='AnyDistribution'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='AnyDistribution'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_mean()

get_variance()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_mean(*mean*)

set_variance(*variance*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='AnyDistribution'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Application**(*name=None*, *version=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Application'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Application'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Application'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Application'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_name()

get_version()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_name(*name*)

set_version(*version*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Application'*, *mapping_=None*)

class PMML44Super.**Apply**(*function=None*, *mapMissingTo=None*, *defaultValue=None*, *invalidValueTreatment='returnInvalid'*, *Extension=None*, *FieldRef=None*, *Constant=None*, *NormContinuous=None*, *NormDiscrete=None*, *Discretize=None*, *MapValues=None*, *TextIndex=None*, *Apply_member=None*, *Aggregate=None*, *Lag=None*)

Bases: PMML44Super.GeneratedsSuper

add_Aggregate(*value*)

add_Apply(*value*)

add_Constant(*value*)

add_Discretize(*value*)

add_Extension(*value*)

add_FieldRef(*value*)

add_Lag(*value*)

add_MapValues(*value*)

add_NormContinuous(*value*)

add_NormDiscrete(*value*)

add_TextIndex(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Apply'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Apply'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Apply'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Apply'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Aggregate()

get_Apply()

get_Constant()

get_Discretize()

get_Extension()

get_FieldRef()

get_Lag()

get_MapValues()

get_NormContinuous()

get_NormDiscrete()

get_TextIndex()

get_defaultValue()

get_function()

get_invalidValueTreatment()

get_mapMissingTo()

hasContent_()

insert_Aggregate_at(*index*, *value*)

insert_Apply_at(*index*, *value*)

insert_Constant_at(*index*, *value*)

insert_Discretize_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_FieldRef_at(*index*, *value*)

insert_Lag_at(*index*, *value*)

insert_MapValues_at(*index*, *value*)

insert_NormContinuous_at(*index*, *value*)

insert_NormDiscrete_at(*index*, *value*)

insert_TextIndex_at(*index*, *value*)

replace_Aggregate_at(*index*, *value*)

replace_Apply_at(*index*, *value*)

replace_Constant_at(*index*, *value*)

replace_Discretize_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_FieldRef_at(*index*, *value*)

replace_Lag_at(*index*, *value*)

replace_MapValues_at(*index*, *value*)

replace_NormContinuous_at(*index*, *value*)

replace_NormDiscrete_at(*index*, *value*)

replace_TextIndex_at(*index*, *value*)

set_Aggregate(*Aggregate*)

set_Apply(*Apply*)

set_Constant(*Constant*)

set_Discretize(*Discretize*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_Lag(*Lag*)

set_MapValues(*MapValues*)

set_NormContinuous(*NormContinuous*)

set_NormDiscrete(*NormDiscrete*)

set_TextIndex(*TextIndex*)

set_defaultValue(*defaultValue*)

set_function(*function*)

set_invalidValueTreatment(*invalidValueTreatment*)

set_mapMissingTo(*mapMissingTo*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Apply'*, *mapping_=None*)

validate_INVALID_VALUE_TREATMENT_METHOD(*value*)

class PMML44Super.**ArrayType**(*n=None*, *type_=None*, *valueOf_=None*, *mixedclass_=None*, *content_=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_='', name_='ArrayType'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ArrayType'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ArrayType'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ArrayType'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_n()

get_type()

get_valueOf_()

hasContent_()

set_n(*n*)

set_type(*type_*)

set_valueOf_(*valueOf_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ArrayType'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**AssociationModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *numberOfTransactions=None*, *maxNumberOfItemsPerTA=None*, *avgNumberOfItemsPerTA=None*, *minimumSupport=None*, *minimumConfidence=None*, *lengthLimit=None*, *numberOfItems=None*, *numberOfItemsets=None*, *numberOfRules=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *LocalTransformations=None*, *Item=None*, *Itemset=None*, *AssociationRule=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_AssociationRule(*value*)

add_Extension(*value*)

add_Item(*value*)

add_Itemset(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='AssociationModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='AssociationModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='AssociationModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='AssociationModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AssociationRule()

get_Extension()

get_Item()

get_Itemset()

get_LocalTransformations()

get_MiningSchema()

get_ModelStats()

get_ModelVerification()

get_Output()

get_algorithmName()

get_avgNumberOfItemsPerTA()

get_functionName()

get_isScorable()

get_lengthLimit()

get_maxNumberOfItemsPerTA()

get_minimumConfidence()

get_minimumSupport()

get_modelName()

get_numberOfItems()

get_numberOfItemsets()

get_numberOfRules()

get_numberOfTransactions()

hasContent_()

insert_AssociationRule_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_Item_at(*index*, *value*)

insert_Itemset_at(*index*, *value*)

replace_AssociationRule_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Item_at(*index*, *value*)

replace_Itemset_at(*index*, *value*)

set_AssociationRule(*AssociationRule*)

set_Extension(*Extension*)

set_Item(*Item*)

set_Itemset(*Itemset*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_algorithmName(*algorithmName*)

set_avgNumberOfItemsPerTA(*avgNumberOfItemsPerTA*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_lengthLimit(*lengthLimit*)

set_maxNumberOfItemsPerTA(*maxNumberOfItemsPerTA*)

set_minimumConfidence(*minimumConfidence*)

set_minimumSupport(*minimumSupport*)

set_modelName(*modelName*)

set_numberOfItems(*numberOfItems*)

set_numberOfItemsets(*numberOfItemsets*)

set_numberOfRules(*numberOfRules*)

set_numberOfTransactions(*numberOfTransactions*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='AssociationModel'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_MINING_FUNCTION(*value*)

validate_PROB_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**AssociationRule**(*antecedent=None*, *consequent=None*, *support=None*, *confidence=None*, *lift=None*, *leverage=None*, *affinity=None*, *id=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='AssociationRule'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='AssociationRule'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='AssociationRule'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='AssociationRule'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_affinity()

get_antecedent()

get_confidence()

get_consequent()

get_id()

get_leverage()

get_lift()

get_support()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_affinity(*affinity*)

set_antecedent(*antecedent*)

set_confidence(*confidence*)

set_consequent(*consequent*)

set_id(*id*)

set_leverage(*leverage*)

set_lift(*lift*)

set_support(*support*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='AssociationRule'*, *mapping_=None*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**Attribute**(*reasonCode=None*, *partialScore=None*, *Extension=None*, *SimplePredicate=None*, *CompoundPredicate=None*, *SimpleSetPredicate=None*, *True_=None*, *False_=None*, *ComplexPartialScore=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Attribute'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Attribute'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Attribute'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Attribute'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ComplexPartialScore()

get_CompoundPredicate()

get_Extension()

get_False()

get_SimplePredicate()

get_SimpleSetPredicate()

get_True()

get_partialScore()

get_reasonCode()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_ComplexPartialScore(*ComplexPartialScore*)

set_CompoundPredicate(*CompoundPredicate*)

set_Extension(*Extension*)

set_False(*False_*)

set_SimplePredicate(*SimplePredicate*)

set_SimpleSetPredicate(*SimpleSetPredicate*)

set_True(*True_*)

set_partialScore(*partialScore*)

set_reasonCode(*reasonCode*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Attribute'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**BaseCumHazardTables**(*maxTime=None*, *Extension=None*, *BaselineStratum=None*, *BaselineCell=None*)

Bases: PMML44Super.GeneratedsSuper

add_BaselineCell(*value*)

add_BaselineStratum(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BaseCumHazardTables'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BaseCumHazardTables'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BaseCumHazardTables'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BaseCumHazardTables'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_BaselineCell()

get_BaselineStratum()

get_Extension()

get_maxTime()

hasContent_()

insert_BaselineCell_at(*index*, *value*)

insert_BaselineStratum_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_BaselineCell_at(*index*, *value*)

replace_BaselineStratum_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_BaselineCell(*BaselineCell*)

set_BaselineStratum(*BaselineStratum*)

set_Extension(*Extension*)

set_maxTime(*maxTime*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BaseCumHazardTables'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Baseline**(*AnyDistribution=None*, *GaussianDistribution=None*, *PoissonDistribution=None*, *UniformDistribution=None*, *CountTable=None*, *NormalizedCountTable=None*, *FieldRef=None*)

Bases: PMML44Super.GeneratedsSuper

add_FieldRef(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Baseline'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Baseline'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Baseline'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Baseline'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AnyDistribution()

get_CountTable()

get_FieldRef()

get_GaussianDistribution()

get_NormalizedCountTable()

get_PoissonDistribution()

get_UniformDistribution()

hasContent_()

insert_FieldRef_at(*index*, *value*)

replace_FieldRef_at(*index*, *value*)

set_AnyDistribution(*AnyDistribution*)

set_CountTable(*CountTable*)

set_FieldRef(*FieldRef*)

set_GaussianDistribution(*GaussianDistribution*)

set_NormalizedCountTable(*NormalizedCountTable*)

set_PoissonDistribution(*PoissonDistribution*)

set_UniformDistribution(*UniformDistribution*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Baseline'*, *mapping_=None*)

class PMML44Super.**BaselineCell**(*time=None*, *cumHazard=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BaselineCell'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BaselineCell'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BaselineCell'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BaselineCell'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_cumHazard()

get_time()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_cumHazard(*cumHazard*)

set_time(*time*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BaselineCell'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**BaselineModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *TestDistributions=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BaselineModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BaselineModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BaselineModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BaselineModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_Targets()

get_TestDistributions()

get_algorithmName()

get_functionName()

get_isScorable()

get_modelName()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_Targets(*Targets*)

set_TestDistributions(*TestDistributions*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BaselineModel'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

class PMML44Super.**BaselineStratum**(*value=None*, *label=None*, *maxTime=None*, *Extension=None*, *BaselineCell=None*)

Bases: PMML44Super.GeneratedsSuper

add_BaselineCell(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BaselineStratum'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BaselineStratum'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BaselineStratum'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BaselineStratum'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_BaselineCell()

get_Extension()

get_label()

get_maxTime()

get_value()

hasContent_()

insert_BaselineCell_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_BaselineCell_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_BaselineCell(*BaselineCell*)

set_Extension(*Extension*)

set_label(*label*)

set_maxTime(*maxTime*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BaselineStratum'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**BayesInput**(*fieldName=None*, *Extension=None*, *TargetValueStats=None*, *DerivedField=None*, *PairCounts=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_PairCounts(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BayesInput'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BayesInput'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BayesInput'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BayesInput'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DerivedField()

get_Extension()

get_PairCounts()

get_TargetValueStats()

get_fieldName()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_PairCounts_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_PairCounts_at(*index*, *value*)

set_DerivedField(*DerivedField*)

set_Extension(*Extension*)

set_PairCounts(*PairCounts*)

set_TargetValueStats(*TargetValueStats*)

set_fieldName(*fieldName*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BayesInput'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**BayesInputs**(*Extension=None*, *BayesInput=None*)

Bases: PMML44Super.GeneratedsSuper

add_BayesInput(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BayesInputs'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BayesInputs'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BayesInputs'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BayesInputs'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_BayesInput()

get_Extension()

hasContent_()

insert_BayesInput_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_BayesInput_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_BayesInput(*BayesInput*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BayesInputs'*, *mapping_=None*)

class PMML44Super.**BayesOutput**(*fieldName=None*, *Extension=None*, *TargetValueCounts=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BayesOutput'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BayesOutput'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BayesOutput'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BayesOutput'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

get_TargetValueCounts()

get_fieldName()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_TargetValueCounts(*TargetValueCounts*)

set_fieldName(*fieldName*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BayesOutput'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**BayesianNetworkModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *modelType='General'*, *inferenceMethod='Other'*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *BayesianNetworkNodes=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BayesianNetworkModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BayesianNetworkModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BayesianNetworkModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BayesianNetworkModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_BayesianNetworkNodes()

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_Targets()

get_algorithmName()

get_functionName()

get_inferenceMethod()

get_isScorable()

get_modelName()

get_modelType()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_BayesianNetworkNodes(*BayesianNetworkNodes*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_Targets(*Targets*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_inferenceMethod(*inferenceMethod*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

set_modelType(*modelType*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BayesianNetworkModel'*, *mapping_=None*)

validate_BN_TYPE(*value*)

validate_INFERENCE_TYPE(*value*)

validate_MINING_FUNCTION(*value*)

class PMML44Super.**BayesianNetworkNodes**(*Extension=None*, *DiscreteNode=None*, *ContinuousNode=None*)

Bases: PMML44Super.GeneratedsSuper

add_ContinuousNode(*value*)

add_DiscreteNode(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BayesianNetworkNodes'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BayesianNetworkNodes'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BayesianNetworkNodes'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BayesianNetworkNodes'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ContinuousNode()

get_DiscreteNode()

get_Extension()

hasContent_()

insert_ContinuousNode_at(*index*, *value*)

insert_DiscreteNode_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_ContinuousNode_at(*index*, *value*)

replace_DiscreteNode_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_ContinuousNode(*ContinuousNode*)

set_DiscreteNode(*DiscreteNode*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BayesianNetworkNodes'*, *mapping_=None*)

class PMML44Super.**BlockIndicator**(*field=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BlockIndicator'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BlockIndicator'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BlockIndicator'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BlockIndicator'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_field()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_field(*field*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BlockIndicator'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**BoundaryValueMeans**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BoundaryValueMeans'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BoundaryValueMeans'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BoundaryValueMeans'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BoundaryValueMeans'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BoundaryValueMeans'*, *mapping_=None*)

class PMML44Super.**BoundaryValues**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='BoundaryValues'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='BoundaryValues'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='BoundaryValues'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='BoundaryValues'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='BoundaryValues'*, *mapping_=None*)

class PMML44Super.**COUNT_TABLE_TYPE**(*sample=None*, *Extension=None*, *FieldValue=None*, *FieldValueCount=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_FieldValue(*value*)

add_FieldValueCount(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='COUNT-TABLE-TYPE'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='COUNT-TABLE-TYPE'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='COUNT-TABLE-TYPE'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='COUNT-TABLE-TYPE'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_FieldValue()

get_FieldValueCount()

get_sample()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_FieldValueCount_at(*index*, *value*)

insert_FieldValue_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_FieldValueCount_at(*index*, *value*)

replace_FieldValue_at(*index*, *value*)

set_Extension(*Extension*)

set_FieldValue(*FieldValue*)

set_FieldValueCount(*FieldValueCount*)

set_sample(*sample*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='COUNT-TABLE-TYPE'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**CategoricalPredictor**(*name=None*, *value=None*, *coefficient=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='CategoricalPredictor'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='CategoricalPredictor'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='CategoricalPredictor'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='CategoricalPredictor'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_coefficient()

get_name()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_coefficient(*coefficient*)

set_name(*name*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='CategoricalPredictor'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Categories**(*Extension=None*, *Category=None*)

Bases: PMML44Super.GeneratedsSuper

add_Category(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Categories'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Categories'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Categories'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Categories'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Category()

get_Extension()

hasContent_()

insert_Category_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_Category_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Category(*Category*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Categories'*, *mapping_=None*)

class PMML44Super.**Category**(*value=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Category'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Category'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Category'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Category'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Category'*, *mapping_=None*)

class PMML44Super.**Characteristic**(*name=None*, *reasonCode=None*, *baselineScore=None*, *Extension=None*, *Attribute=None*)

Bases: PMML44Super.GeneratedsSuper

add_Attribute(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Characteristic'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Characteristic'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Characteristic'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Characteristic'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Attribute()

get_Extension()

get_baselineScore()

get_name()

get_reasonCode()

hasContent_()

insert_Attribute_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_Attribute_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Attribute(*Attribute*)

set_Extension(*Extension*)

set_baselineScore(*baselineScore*)

set_name(*name*)

set_reasonCode(*reasonCode*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Characteristic'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**Characteristics**(*Extension=None*, *Characteristic=None*)

Bases: PMML44Super.GeneratedsSuper

add_Characteristic(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Characteristics'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Characteristics'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Characteristics'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Characteristics'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Characteristic()

get_Extension()

hasContent_()

insert_Characteristic_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_Characteristic_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Characteristic(*Characteristic*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Characteristics'*, *mapping_=None*)

class PMML44Super.**ChildParent**(*childField=None*, *parentField=None*, *parentLevelField=None*, *isRecursive='no'*, *Extension=None*, *FieldColumnPair=None*, *TableLocator=None*, *InlineTable=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_FieldColumnPair(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ChildParent'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ChildParent'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ChildParent'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ChildParent'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_FieldColumnPair()

get_InlineTable()

get_TableLocator()

get_childField()

get_isRecursive()

get_parentField()

get_parentLevelField()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_FieldColumnPair_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_FieldColumnPair_at(*index*, *value*)

set_Extension(*Extension*)

set_FieldColumnPair(*FieldColumnPair*)

set_InlineTable(*InlineTable*)

set_TableLocator(*TableLocator*)

set_childField(*childField*)

set_isRecursive(*isRecursive*)

set_parentField(*parentField*)

set_parentLevelField(*parentLevelField*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ChildParent'*, *mapping_=None*)

class PMML44Super.**ClassLabels**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ClassLabels'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ClassLabels'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ClassLabels'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ClassLabels'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ClassLabels'*, *mapping_=None*)

class PMML44Super.**Cluster**(*id=None*, *name=None*, *size=None*, *Extension=None*, *KohonenMap=None*, *Array=None*, *Partition=None*, *Covariances=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Cluster'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Cluster'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Cluster'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Cluster'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Covariances()

get_Extension()

get_KohonenMap()

get_Partition()

get_id()

get_name()

get_size()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Covariances(*Covariances*)

set_Extension(*Extension*)

set_KohonenMap(*KohonenMap*)

set_Partition(*Partition*)

set_id(*id*)

set_name(*name*)

set_size(*size*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Cluster'*, *mapping_=None*)

class PMML44Super.**ClusteringField**(*field=None*, *isCenterField='true'*, *fieldWeight='1'*, *similarityScale=None*, *compareFunction=None*, *Extension=None*, *Comparisons=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ClusteringField'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ClusteringField'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ClusteringField'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ClusteringField'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Comparisons()

get_Extension()

get_compareFunction()

get_field()

get_fieldWeight()

get_isCenterField()

get_similarityScale()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Comparisons(*Comparisons*)

set_Extension(*Extension*)

set_compareFunction(*compareFunction*)

set_field(*field*)

set_fieldWeight(*fieldWeight*)

set_isCenterField(*isCenterField*)

set_similarityScale(*similarityScale*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ClusteringField'*, *mapping_=None*)

validate_COMPARE_FUNCTION(*value*)

validate_FIELD_NAME(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ClusteringModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *modelClass=None*, *numberOfClusters=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *LocalTransformations=None*, *ComparisonMeasure=None*, *ClusteringField=None*, *MissingValueWeights=None*, *Cluster=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Cluster(*value*)

add_ClusteringField(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ClusteringModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ClusteringModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ClusteringModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ClusteringModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Cluster()

get_ClusteringField()

get_ComparisonMeasure()

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_MissingValueWeights()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_algorithmName()

get_functionName()

get_isScorable()

get_modelClass()

get_modelName()

get_numberOfClusters()

hasContent_()

insert_Cluster_at(*index*, *value*)

insert_ClusteringField_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_Cluster_at(*index*, *value*)

replace_ClusteringField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Cluster(*Cluster*)

set_ClusteringField(*ClusteringField*)

set_ComparisonMeasure(*ComparisonMeasure*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_MissingValueWeights(*MissingValueWeights*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelClass(*modelClass*)

set_modelName(*modelName*)

set_numberOfClusters(*numberOfClusters*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ClusteringModel'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_MINING_FUNCTION(*value*)

class PMML44Super.**ClusteringModelQuality**(*dataName=None*, *SSE=None*, *SSB=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ClusteringModelQuality'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ClusteringModelQuality'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ClusteringModelQuality'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ClusteringModelQuality'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_SSB()

get_SSE()

get_dataName()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_SSB(*SSB*)

set_SSE(*SSE*)

set_dataName(*dataName*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ClusteringModelQuality'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**Coefficient**(*value='0'*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Coefficient'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Coefficient'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Coefficient'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Coefficient'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Coefficient'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Coefficients**(*numberOfCoefficients=None*, *absoluteValue='0'*, *Extension=None*, *Coefficient=None*)

Bases: PMML44Super.GeneratedsSuper

add_Coefficient(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Coefficients'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Coefficients'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Coefficients'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Coefficients'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Coefficient()

get_Extension()

get_absoluteValue()

get_numberOfCoefficients()

hasContent_()

insert_Coefficient_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_Coefficient_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Coefficient(*Coefficient*)

set_Extension(*Extension*)

set_absoluteValue(*absoluteValue*)

set_numberOfCoefficients(*numberOfCoefficients*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Coefficients'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ComparisonMeasure**(*kind=None*, *compareFunction='absDiff'*, *minimum=None*, *maximum=None*, *Extension=None*, *euclidean=None*, *squaredEuclidean=None*, *chebychev=None*, *cityBlock=None*, *minkowski=None*, *simpleMatching=None*, *jaccard=None*, *tanimoto=None*, *binarySimilarity=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ComparisonMeasure'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ComparisonMeasure'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ComparisonMeasure'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ComparisonMeasure'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_binarySimilarity()

get_chebychev()

get_cityBlock()

get_compareFunction()

get_euclidean()

get_jaccard()

get_kind()

get_maximum()

get_minimum()

get_minkowski()

get_simpleMatching()

get_squaredEuclidean()

get_tanimoto()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_binarySimilarity(*binarySimilarity*)

set_chebychev(*chebychev*)

set_cityBlock(*cityBlock*)

set_compareFunction(*compareFunction*)

set_euclidean(*euclidean*)

set_jaccard(*jaccard*)

set_kind(*kind*)

set_maximum(*maximum*)

set_minimum(*minimum*)

set_minkowski(*minkowski*)

set_simpleMatching(*simpleMatching*)

set_squaredEuclidean(*squaredEuclidean*)

set_tanimoto(*tanimoto*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ComparisonMeasure'*, *mapping_=None*)

validate_COMPARE_FUNCTION(*value*)

validate_NUMBER(*value*)

class PMML44Super.**Comparisons**(*Extension=None*, *Matrix=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Comparisons'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Comparisons'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Comparisons'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Comparisons'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Matrix()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Matrix(*Matrix*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Comparisons'*, *mapping_=None*)

class PMML44Super.**ComplexPartialScore**(*Extension=None*, *FieldRef=None*, *Constant=None*, *NormContinuous=None*, *NormDiscrete=None*, *Discretize=None*, *MapValues=None*, *TextIndex=None*, *Apply=None*, *Aggregate=None*, *Lag=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ComplexPartialScore'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ComplexPartialScore'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ComplexPartialScore'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ComplexPartialScore'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Aggregate()

get_Apply()

get_Constant()

get_Discretize()

get_Extension()

get_FieldRef()

get_Lag()

get_MapValues()

get_NormContinuous()

get_NormDiscrete()

get_TextIndex()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Aggregate(*Aggregate*)

set_Apply(*Apply*)

set_Constant(*Constant*)

set_Discretize(*Discretize*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_Lag(*Lag*)

set_MapValues(*MapValues*)

set_NormContinuous(*NormContinuous*)

set_NormDiscrete(*NormDiscrete*)

set_TextIndex(*TextIndex*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ComplexPartialScore'*, *mapping_=None*)

class PMML44Super.**CompoundPredicate**(*booleanOperator=None*, *Extension=None*, *SimplePredicate=None*, *CompoundPredicate_member=None*, *SimpleSetPredicate=None*, *True_=None*, *False_=None*)

Bases: PMML44Super.GeneratedsSuper

add_CompoundPredicate(*value*)

add_Extension(*value*)

add_False(*value*)

add_SimplePredicate(*value*)

add_SimpleSetPredicate(*value*)

add_True(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='CompoundPredicate'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='CompoundPredicate'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='CompoundPredicate'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='CompoundPredicate'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_CompoundPredicate()

get_Extension()

get_False()

get_SimplePredicate()

get_SimpleSetPredicate()

get_True()

get_booleanOperator()

hasContent_()

insert_CompoundPredicate_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_False_at(*index*, *value*)

insert_SimplePredicate_at(*index*, *value*)

insert_SimpleSetPredicate_at(*index*, *value*)

insert_True_at(*index*, *value*)

replace_CompoundPredicate_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_False_at(*index*, *value*)

replace_SimplePredicate_at(*index*, *value*)

replace_SimpleSetPredicate_at(*index*, *value*)

replace_True_at(*index*, *value*)

set_CompoundPredicate(*CompoundPredicate*)

set_Extension(*Extension*)

set_False(*False_*)

set_SimplePredicate(*SimplePredicate*)

set_SimpleSetPredicate(*SimpleSetPredicate*)

set_True(*True_*)

set_booleanOperator(*booleanOperator*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='CompoundPredicate'*, *mapping_=None*)

class PMML44Super.**CompoundRule**(*Extension=None*, *SimplePredicate=None*, *CompoundPredicate=None*, *SimpleSetPredicate=None*, *True_=None*, *False_=None*, *SimpleRule=None*, *CompoundRule_member=None*)

Bases: PMML44Super.GeneratedsSuper

add_CompoundRule(*value*)

add_Extension(*value*)

add_SimpleRule(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='CompoundRule'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='CompoundRule'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='CompoundRule'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='CompoundRule'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_CompoundPredicate()

get_CompoundRule()

get_Extension()

get_False()

get_SimplePredicate()

get_SimpleRule()

get_SimpleSetPredicate()

get_True()

hasContent_()

insert_CompoundRule_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_SimpleRule_at(*index*, *value*)

replace_CompoundRule_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_SimpleRule_at(*index*, *value*)

set_CompoundPredicate(*CompoundPredicate*)

set_CompoundRule(*CompoundRule*)

set_Extension(*Extension*)

set_False(*False_*)

set_SimplePredicate(*SimplePredicate*)

set_SimpleRule(*SimpleRule*)

set_SimpleSetPredicate(*SimpleSetPredicate*)

set_True(*True_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='CompoundRule'*, *mapping_=None*)

class PMML44Super.**Con**(*from_=None*, *weight=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Con'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Con'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Con'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Con'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_from()

get_weight()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_from(*from_*)

set_weight(*weight*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Con'*, *mapping_=None*)

validate_NN_NEURON_IDREF(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ConfusionMatrix**(*Extension=None*, *ClassLabels=None*, *Matrix=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ConfusionMatrix'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ConfusionMatrix'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ConfusionMatrix'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ConfusionMatrix'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ClassLabels()

get_Extension()

get_Matrix()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_ClassLabels(*ClassLabels*)

set_Extension(*Extension*)

set_Matrix(*Matrix*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ConfusionMatrix'*, *mapping_=None*)

class PMML44Super.**ConsequentSequence**(*Extension=None*, *SequenceReference=None*, *Time=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ConsequentSequence'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ConsequentSequence'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ConsequentSequence'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ConsequentSequence'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_SequenceReference()

get_Time()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_SequenceReference(*SequenceReference*)

set_Time(*Time*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ConsequentSequence'*, *mapping_=None*)

class PMML44Super.**Constant**(*dataType=None*, *valueOf_=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Constant'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Constant'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Constant'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Constant'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_dataType()

get_valueOf_()

hasContent_()

set_dataType(*dataType*)

set_valueOf_(*valueOf_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Constant'*, *mapping_=None*)

validate_DATATYPE(*value*)

class PMML44Super.**Constraints**(*minimumNumberOfItems='1'*, *maximumNumberOfItems=None*, *minimumNumberOfAntecedentItems='1'*, *maximumNumberOfAntecedentItems=None*, *minimumNumberOfConsequentItems='1'*, *maximumNumberOfConsequentItems=None*, *minimumSupport='0'*, *minimumConfidence='0'*, *minimumLift='0'*, *minimumTotalSequenceTime='0'*, *maximumTotalSequenceTime=None*, *minimumItemsetSeparationTime='0'*, *maximumItemsetSeparationTime=None*, *minimumAntConsSeparationTime='0'*, *maximumAntConsSeparationTime=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Constraints'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Constraints'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Constraints'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Constraints'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_maximumAntConsSeparationTime()

get_maximumItemsetSeparationTime()

get_maximumNumberOfAntecedentItems()

get_maximumNumberOfConsequentItems()

get_maximumNumberOfItems()

get_maximumTotalSequenceTime()

get_minimumAntConsSeparationTime()

get_minimumConfidence()

get_minimumItemsetSeparationTime()

get_minimumLift()

get_minimumNumberOfAntecedentItems()

get_minimumNumberOfConsequentItems()

get_minimumNumberOfItems()

get_minimumSupport()

get_minimumTotalSequenceTime()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_maximumAntConsSeparationTime(*maximumAntConsSeparationTime*)

set_maximumItemsetSeparationTime(*maximumItemsetSeparationTime*)

set_maximumNumberOfAntecedentItems(*maximumNumberOfAntecedentItems*)

set_maximumNumberOfConsequentItems(*maximumNumberOfConsequentItems*)

set_maximumNumberOfItems(*maximumNumberOfItems*)

set_maximumTotalSequenceTime(*maximumTotalSequenceTime*)

set_minimumAntConsSeparationTime(*minimumAntConsSeparationTime*)

set_minimumConfidence(*minimumConfidence*)

set_minimumItemsetSeparationTime(*minimumItemsetSeparationTime*)

set_minimumLift(*minimumLift*)

set_minimumNumberOfAntecedentItems(*minimumNumberOfAntecedentItems*)

set_minimumNumberOfConsequentItems(*minimumNumberOfConsequentItems*)

set_minimumNumberOfItems(*minimumNumberOfItems*)

set_minimumSupport(*minimumSupport*)

set_minimumTotalSequenceTime(*minimumTotalSequenceTime*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Constraints'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ContStats**(*totalValuesSum=None*, *totalSquaresSum=None*, *Extension=None*, *Interval=None*, *NUM_ARRAY=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Interval(*value*)

add_NUM_ARRAY(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ContStats'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ContStats'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ContStats'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ContStats'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Interval()

get_NUM_ARRAY()

get_totalSquaresSum()

get_totalValuesSum()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Interval_at(*index*, *value*)

insert_NUM_ARRAY_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Interval_at(*index*, *value*)

replace_NUM_ARRAY_at(*index*, *value*)

set_Extension(*Extension*)

set_Interval(*Interval*)

set_NUM_ARRAY(*NUM_ARRAY*)

set_totalSquaresSum(*totalSquaresSum*)

set_totalValuesSum(*totalValuesSum*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ContStats'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**ContinuousConditionalProbability**(*count=None*, *Extension=None*, *ParentValue=None*, *ContinuousDistribution=None*)

Bases: PMML44Super.GeneratedsSuper

add_ContinuousDistribution(*value*)

add_Extension(*value*)

add_ParentValue(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ContinuousConditionalProbability'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ContinuousConditionalProbability'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ContinuousConditionalProbability'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ContinuousConditionalProbability'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ContinuousDistribution()

get_Extension()

get_ParentValue()

get_count()

hasContent_()

insert_ContinuousDistribution_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_ParentValue_at(*index*, *value*)

replace_ContinuousDistribution_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_ParentValue_at(*index*, *value*)

set_ContinuousDistribution(*ContinuousDistribution*)

set_Extension(*Extension*)

set_ParentValue(*ParentValue*)

set_count(*count*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ContinuousConditionalProbability'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ContinuousDistribution**(*Extension=None*, *TriangularDistributionForBN=None*, *NormalDistributionForBN=None*, *LognormalDistributionForBN=None*, *UniformDistributionForBN=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ContinuousDistribution'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ContinuousDistribution'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ContinuousDistribution'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ContinuousDistribution'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LognormalDistributionForBN()

get_NormalDistributionForBN()

get_TriangularDistributionForBN()

get_UniformDistributionForBN()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_LognormalDistributionForBN(*LognormalDistributionForBN*)

set_NormalDistributionForBN(*NormalDistributionForBN*)

set_TriangularDistributionForBN(*TriangularDistributionForBN*)

set_UniformDistributionForBN(*UniformDistributionForBN*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ContinuousDistribution'*, *mapping_=None*)

class PMML44Super.**ContinuousNode**(*name=None*, *count=None*, *Extension=None*, *DerivedField=None*, *ContinuousConditionalProbability=None*, *ContinuousDistribution=None*)

Bases: PMML44Super.GeneratedsSuper

add_ContinuousConditionalProbability(*value*)

add_ContinuousDistribution(*value*)

add_DerivedField(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ContinuousNode'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ContinuousNode'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ContinuousNode'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ContinuousNode'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ContinuousConditionalProbability()

get_ContinuousDistribution()

get_DerivedField()

get_Extension()

get_count()

get_name()

hasContent_()

insert_ContinuousConditionalProbability_at(*index*, *value*)

insert_ContinuousDistribution_at(*index*, *value*)

insert_DerivedField_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_ContinuousConditionalProbability_at(*index*, *value*)

replace_ContinuousDistribution_at(*index*, *value*)

replace_DerivedField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_ContinuousConditionalProbability(*ContinuousConditionalProbability*)

set_ContinuousDistribution(*ContinuousDistribution*)

set_DerivedField(*DerivedField*)

set_Extension(*Extension*)

set_count(*count*)

set_name(*name*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ContinuousNode'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**CorrelationFields**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='CorrelationFields'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='CorrelationFields'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='CorrelationFields'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='CorrelationFields'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='CorrelationFields'*, *mapping_=None*)

class PMML44Super.**CorrelationMethods**(*Extension=None*, *Matrix=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='CorrelationMethods'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='CorrelationMethods'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='CorrelationMethods'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='CorrelationMethods'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Matrix()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Matrix(*Matrix*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='CorrelationMethods'*, *mapping_=None*)

class PMML44Super.**CorrelationValues**(*Extension=None*, *Matrix=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='CorrelationValues'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='CorrelationValues'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='CorrelationValues'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='CorrelationValues'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Matrix()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Matrix(*Matrix*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='CorrelationValues'*, *mapping_=None*)

class PMML44Super.**Correlations**(*Extension=None*, *CorrelationFields=None*, *CorrelationValues=None*, *CorrelationMethods=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Correlations'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Correlations'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Correlations'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Correlations'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_CorrelationFields()

get_CorrelationMethods()

get_CorrelationValues()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_CorrelationFields(*CorrelationFields*)

set_CorrelationMethods(*CorrelationMethods*)

set_CorrelationValues(*CorrelationValues*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Correlations'*, *mapping_=None*)

class PMML44Super.**Counts**(*totalFreq=None*, *missingFreq=None*, *invalidFreq=None*, *cardinality=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Counts'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Counts'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Counts'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Counts'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_cardinality()

get_invalidFreq()

get_missingFreq()

get_totalFreq()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_cardinality(*cardinality*)

set_invalidFreq(*invalidFreq*)

set_missingFreq(*missingFreq*)

set_totalFreq(*totalFreq*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Counts'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**Covariances**(*Extension=None*, *Matrix=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Covariances'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Covariances'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Covariances'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Covariances'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Matrix()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Matrix(*Matrix*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Covariances'*, *mapping_=None*)

class PMML44Super.**CovariateList**(*Extension=None*, *Predictor=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Predictor(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='CovariateList'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='CovariateList'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='CovariateList'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='CovariateList'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Predictor()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Predictor_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Predictor_at(*index*, *value*)

set_Extension(*Extension*)

set_Predictor(*Predictor*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='CovariateList'*, *mapping_=None*)

class PMML44Super.**DataDictionary**(*numberOfFields=None*, *Extension=None*, *DataField=None*, *Taxonomy=None*)

Bases: PMML44Super.GeneratedsSuper

add_DataField(*value*)

add_Extension(*value*)

add_Taxonomy(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DataDictionary'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DataDictionary'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DataDictionary'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DataDictionary'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DataField()

get_Extension()

get_Taxonomy()

get_numberOfFields()

hasContent_()

insert_DataField_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_Taxonomy_at(*index*, *value*)

replace_DataField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Taxonomy_at(*index*, *value*)

set_DataField(*DataField*)

set_Extension(*Extension*)

set_Taxonomy(*Taxonomy*)

set_numberOfFields(*numberOfFields*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DataDictionary'*, *mapping_=None*)

class PMML44Super.**DataField**(*name=None*, *displayName=None*, *optype=None*, *dataType=None*, *mimeType=None*, *taxonomy=None*, *isCyclic='0'*, *Extension=None*, *Interval=None*, *Value=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Interval(*value*)

add_Value(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DataField'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DataField'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DataField'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DataField'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Interval()

get_Value()

get_dataType()

get_displayName()

get_isCyclic()

get_mimeType()

get_name()

get_optype()

get_taxonomy()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Interval_at(*index*, *value*)

insert_Value_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Interval_at(*index*, *value*)

replace_Value_at(*index*, *value*)

set_Extension(*Extension*)

set_Interval(*Interval*)

set_Value(*Value*)

set_dataType(*dataType*)

set_displayName(*displayName*)

set_isCyclic(*isCyclic*)

set_mimeType(*mimeType*)

set_name(*name*)

set_optype(*optype*)

set_taxonomy(*taxonomy*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DataField'*, *mapping_=None*)

validate_DATATYPE(*value*)

validate_FIELD_NAME(*value*)

validate_OPTYPE(*value*)

class PMML44Super.**Decision**(*value=None*, *displayValue=None*, *description=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Decision'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Decision'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Decision'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Decision'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_description()

get_displayValue()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_description(*description*)

set_displayValue(*displayValue*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Decision'*, *mapping_=None*)

class PMML44Super.**DecisionTree**(*modelName=None*, *functionName=None*, *algorithmName=None*, *missingValueStrategy='none'*, *missingValuePenalty='1.0'*, *noTrueChildStrategy='returnNullPrediction'*, *splitCharacteristic='multiSplit'*, *Extension=None*, *Output=None*, *ModelStats=None*, *Targets=None*, *LocalTransformations=None*, *ResultField=None*, *Node=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_ResultField(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DecisionTree'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DecisionTree'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DecisionTree'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DecisionTree'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LocalTransformations()

get_ModelStats()

get_Node()

get_Output()

get_ResultField()

get_Targets()

get_algorithmName()

get_functionName()

get_missingValuePenalty()

get_missingValueStrategy()

get_modelName()

get_noTrueChildStrategy()

get_splitCharacteristic()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_ResultField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_ResultField_at(*index*, *value*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_ModelStats(*ModelStats*)

set_Node(*Node*)

set_Output(*Output*)

set_ResultField(*ResultField*)

set_Targets(*Targets*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_missingValuePenalty(*missingValuePenalty*)

set_missingValueStrategy(*missingValueStrategy*)

set_modelName(*modelName*)

set_noTrueChildStrategy(*noTrueChildStrategy*)

set_splitCharacteristic(*splitCharacteristic*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DecisionTree'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

validate_MISSING_VALUE_STRATEGY(*value*)

validate_NO_TRUE_CHILD_STRATEGY(*value*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**Decisions**(*businessProblem=None*, *description=None*, *Extension=None*, *Decision=None*)

Bases: PMML44Super.GeneratedsSuper

add_Decision(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Decisions'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Decisions'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Decisions'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Decisions'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Decision()

get_Extension()

get_businessProblem()

get_description()

hasContent_()

insert_Decision_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_Decision_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Decision(*Decision*)

set_Extension(*Extension*)

set_businessProblem(*businessProblem*)

set_description(*description*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Decisions'*, *mapping_=None*)

class PMML44Super.**DeepNetwork**(*modelName=None*, *functionName=None*, *algorithmName=None*, *normalizationMethod='none'*, *numberOfLayers=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *TrainingParameters=None*, *NetworkLayer=None*, *NeuralOutputs=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_NetworkLayer(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DeepNetwork'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DeepNetwork'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DeepNetwork'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DeepNetwork'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_NetworkLayer()

get_NeuralOutputs()

get_Output()

get_Targets()

get_TrainingParameters()

get_algorithmName()

get_functionName()

get_isScorable()

get_modelName()

get_normalizationMethod()

get_numberOfLayers()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_NetworkLayer_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_NetworkLayer_at(*index*, *value*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_NetworkLayer(*NetworkLayer*)

set_NeuralOutputs(*NeuralOutputs*)

set_Output(*Output*)

set_Targets(*Targets*)

set_TrainingParameters(*TrainingParameters*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

set_normalizationMethod(*normalizationMethod*)

set_numberOfLayers(*numberOfLayers*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DeepNetwork'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

validate_NN_NORMALIZATION_METHOD(*value*)

class PMML44Super.**DefineFunction**(*name=None*, *optype=None*, *dataType=None*, *Extension=None*, *ParameterField=None*, *FieldRef=None*, *Constant=None*, *NormContinuous=None*, *NormDiscrete=None*, *Discretize=None*, *MapValues=None*, *TextIndex=None*, *Apply=None*, *Aggregate=None*, *Lag=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_ParameterField(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DefineFunction'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DefineFunction'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DefineFunction'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DefineFunction'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Aggregate()

get_Apply()

get_Constant()

get_Discretize()

get_Extension()

get_FieldRef()

get_Lag()

get_MapValues()

get_NormContinuous()

get_NormDiscrete()

get_ParameterField()

get_TextIndex()

get_dataType()

get_name()

get_optype()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_ParameterField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_ParameterField_at(*index*, *value*)

set_Aggregate(*Aggregate*)

set_Apply(*Apply*)

set_Constant(*Constant*)

set_Discretize(*Discretize*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_Lag(*Lag*)

set_MapValues(*MapValues*)

set_NormContinuous(*NormContinuous*)

set_NormDiscrete(*NormDiscrete*)

set_ParameterField(*ParameterField*)

set_TextIndex(*TextIndex*)

set_dataType(*dataType*)

set_name(*name*)

set_optype(*optype*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DefineFunction'*, *mapping_=None*)

validate_DATATYPE(*value*)

validate_OPTYPE(*value*)

class PMML44Super.**Delimiter**(*delimiter=None*, *gap=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Delimiter'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Delimiter'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Delimiter'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Delimiter'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_delimiter()

get_gap()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_delimiter(*delimiter*)

set_gap(*gap*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Delimiter'*, *mapping_=None*)

validate_DELIMITER(*value*)

validate_GAP(*value*)

class PMML44Super.**Denominator**(*Extension=None*, *NonseasonalFactor=None*, *SeasonalFactor=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Denominator'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Denominator'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Denominator'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Denominator'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_NonseasonalFactor()

get_SeasonalFactor()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_NonseasonalFactor(*NonseasonalFactor*)

set_SeasonalFactor(*SeasonalFactor*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Denominator'*, *mapping_=None*)

class PMML44Super.**DerivedField**(*name=None*, *displayName=None*, *optype=None*, *dataType=None*, *datasetName=None*, *trainingBackend=None*, *architectureName=None*, *Extension=None*, *FieldRef=None*, *Constant=None*, *NormContinuous=None*, *NormDiscrete=None*, *Discretize=None*, *MapValues=None*, *TextIndex=None*, *Apply=None*, *Aggregate=None*, *Lag=None*, *Value=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Value(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DerivedField'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DerivedField'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DerivedField'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DerivedField'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Aggregate()

get_Apply()

get_Constant()

get_Discretize()

get_Extension()

get_FieldRef()

get_Lag()

get_MapValues()

get_NormContinuous()

get_NormDiscrete()

get_TextIndex()

get_Value()

get_architectureName()

get_dataType()

get_datasetName()

get_displayName()

get_name()

get_optype()

get_trainingBackend()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Value_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Value_at(*index*, *value*)

set_Aggregate(*Aggregate*)

set_Apply(*Apply*)

set_Constant(*Constant*)

set_Discretize(*Discretize*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_Lag(*Lag*)

set_MapValues(*MapValues*)

set_NormContinuous(*NormContinuous*)

set_NormDiscrete(*NormDiscrete*)

set_TextIndex(*TextIndex*)

set_Value(*Value*)

set_architectureName(*architectureName*)

set_dataType(*dataType*)

set_datasetName(*datasetName*)

set_displayName(*displayName*)

set_name(*name*)

set_optype(*optype*)

set_trainingBackend(*trainingBackend*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DerivedField'*, *mapping_=None*)

validate_ARCHITECTURENAME(*value*)

validate_BACKEND(*value*)

validate_DATATYPE(*value*)

validate_FIELD_NAME(*value*)

validate_OPTYPE(*value*)

class PMML44Super.**DiscrStats**(*modalValue=None*, *Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Array(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DiscrStats'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DiscrStats'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DiscrStats'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DiscrStats'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

get_modalValue()

hasContent_()

insert_Array_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_Array_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

set_modalValue(*modalValue*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DiscrStats'*, *mapping_=None*)

class PMML44Super.**DiscreteConditionalProbability**(*count=None*, *Extension=None*, *ParentValue=None*, *ValueProbability=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_ParentValue(*value*)

add_ValueProbability(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DiscreteConditionalProbability'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DiscreteConditionalProbability'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DiscreteConditionalProbability'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DiscreteConditionalProbability'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_ParentValue()

get_ValueProbability()

get_count()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_ParentValue_at(*index*, *value*)

insert_ValueProbability_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_ParentValue_at(*index*, *value*)

replace_ValueProbability_at(*index*, *value*)

set_Extension(*Extension*)

set_ParentValue(*ParentValue*)

set_ValueProbability(*ValueProbability*)

set_count(*count*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DiscreteConditionalProbability'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**DiscreteNode**(*name=None*, *count=None*, *Extension=None*, *DerivedField=None*, *DiscreteConditionalProbability=None*, *ValueProbability=None*)

Bases: PMML44Super.GeneratedsSuper

add_DerivedField(*value*)

add_DiscreteConditionalProbability(*value*)

add_Extension(*value*)

add_ValueProbability(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DiscreteNode'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DiscreteNode'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DiscreteNode'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DiscreteNode'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DerivedField()

get_DiscreteConditionalProbability()

get_Extension()

get_ValueProbability()

get_count()

get_name()

hasContent_()

insert_DerivedField_at(*index*, *value*)

insert_DiscreteConditionalProbability_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_ValueProbability_at(*index*, *value*)

replace_DerivedField_at(*index*, *value*)

replace_DiscreteConditionalProbability_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_ValueProbability_at(*index*, *value*)

set_DerivedField(*DerivedField*)

set_DiscreteConditionalProbability(*DiscreteConditionalProbability*)

set_Extension(*Extension*)

set_ValueProbability(*ValueProbability*)

set_count(*count*)

set_name(*name*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DiscreteNode'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Discretize**(*field=None*, *mapMissingTo=None*, *defaultValue=None*, *dataType=None*, *Extension=None*, *DiscretizeBin=None*)

Bases: PMML44Super.GeneratedsSuper

add_DiscretizeBin(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Discretize'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Discretize'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Discretize'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Discretize'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DiscretizeBin()

get_Extension()

get_dataType()

get_defaultValue()

get_field()

get_mapMissingTo()

hasContent_()

insert_DiscretizeBin_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_DiscretizeBin_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_DiscretizeBin(*DiscretizeBin*)

set_Extension(*Extension*)

set_dataType(*dataType*)

set_defaultValue(*defaultValue*)

set_field(*field*)

set_mapMissingTo(*mapMissingTo*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Discretize'*, *mapping_=None*)

validate_DATATYPE(*value*)

validate_FIELD_NAME(*value*)

class PMML44Super.**DiscretizeBin**(*binValue=None*, *Extension=None*, *Interval=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DiscretizeBin'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DiscretizeBin'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DiscretizeBin'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DiscretizeBin'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

get_Interval()

get_binValue()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Interval(*Interval*)

set_binValue(*binValue*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DiscretizeBin'*, *mapping_=None*)

class PMML44Super.**DocumentTermMatrix**(*Extension=None*, *Matrix=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DocumentTermMatrix'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DocumentTermMatrix'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DocumentTermMatrix'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DocumentTermMatrix'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

get_Matrix()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Matrix(*Matrix*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DocumentTermMatrix'*, *mapping_=None*)

class PMML44Super.**DynamicRegressor**(*field=None*, *transformation='none'*, *delay='0'*, *futureValuesMethod='constant'*, *targetField=None*, *Extension=None*, *Numerator=None*, *Denominator=None*, *RegressorValues=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='DynamicRegressor'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='DynamicRegressor'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='DynamicRegressor'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='DynamicRegressor'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Denominator()

get_Extension()

get_Numerator()

get_RegressorValues()

get_delay()

get_field()

get_futureValuesMethod()

get_targetField()

get_transformation()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Denominator(*Denominator*)

set_Extension(*Extension*)

set_Numerator(*Numerator*)

set_RegressorValues(*RegressorValues*)

set_delay(*delay*)

set_field(*field*)

set_futureValuesMethod(*futureValuesMethod*)

set_targetField(*targetField*)

set_transformation(*transformation*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='DynamicRegressor'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_INT_NUMBER(*value*)

class PMML44Super.**EventValues**(*Extension=None*, *Value=None*, *Interval=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Interval(*value*)

add_Value(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='EventValues'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='EventValues'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='EventValues'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='EventValues'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Interval()

get_Value()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Interval_at(*index*, *value*)

insert_Value_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Interval_at(*index*, *value*)

replace_Value_at(*index*, *value*)

set_Extension(*Extension*)

set_Interval(*Interval*)

set_Value(*Value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='EventValues'*, *mapping_=None*)

class PMML44Super.**ExponentialSmoothing**(*RMSE=None*, *transformation='none'*, *Level=None*, *Trend_ExpoSmooth=None*, *Seasonality_ExpoSmooth=None*, *TimeValue=None*)

Bases: PMML44Super.GeneratedsSuper

add_TimeValue(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ExponentialSmoothing'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ExponentialSmoothing'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ExponentialSmoothing'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ExponentialSmoothing'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Level()

get_RMSE()

get_Seasonality_ExpoSmooth()

get_TimeValue()

get_Trend_ExpoSmooth()

get_transformation()

hasContent_()

insert_TimeValue_at(*index*, *value*)

replace_TimeValue_at(*index*, *value*)

set_Level(*Level*)

set_RMSE(*RMSE*)

set_Seasonality_ExpoSmooth(*Seasonality_ExpoSmooth*)

set_TimeValue(*TimeValue*)

set_Trend_ExpoSmooth(*Trend_ExpoSmooth*)

set_transformation(*transformation*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ExponentialSmoothing'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Extension**(*extender=None*, *name=None*, *value=None*, *anytypeobjs_=None*)

Bases: PMML44Super.GeneratedsSuper

add_anytypeobjs_(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Extension'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Extension'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Extension'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Extension'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_anytypeobjs_()

get_extender()

get_name()

get_value()

hasContent_()

insert_anytypeobjs_(*index*, *value*)

set_anytypeobjs_(*anytypeobjs_*)

set_extender(*extender*)

set_name(*name*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Extension'*, *mapping_=None*)

class PMML44Super.**FactorList**(*Extension=None*, *Predictor=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Predictor(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='FactorList'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='FactorList'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='FactorList'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FactorList'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Predictor()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Predictor_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Predictor_at(*index*, *value*)

set_Extension(*Extension*)

set_Predictor(*Predictor*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FactorList'*, *mapping_=None*)

class PMML44Super.**False_**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='False'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='False'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='False'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='False'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='False'*, *mapping_=None*)

class PMML44Super.**FieldColumnPair**(*field=None*, *column=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='FieldColumnPair'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='FieldColumnPair'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='FieldColumnPair'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FieldColumnPair'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_column()

get_field()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_column(*column*)

set_field(*field*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FieldColumnPair'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**FieldRef**(*field=None*, *mapMissingTo=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='FieldRef'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='FieldRef'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='FieldRef'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FieldRef'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_field()

get_mapMissingTo()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_field(*field*)

set_mapMissingTo(*mapMissingTo*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FieldRef'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**FieldValue**(*field=None*, *value=None*, *Extension=None*, *FieldValue_member=None*, *FieldValueCount=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_FieldValue(*value*)

add_FieldValueCount(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='FieldValue'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='FieldValue'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='FieldValue'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FieldValue'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_FieldValue()

get_FieldValueCount()

get_field()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_FieldValueCount_at(*index*, *value*)

insert_FieldValue_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_FieldValueCount_at(*index*, *value*)

replace_FieldValue_at(*index*, *value*)

set_Extension(*Extension*)

set_FieldValue(*FieldValue*)

set_FieldValueCount(*FieldValueCount*)

set_field(*field*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FieldValue'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**FieldValueCount**(*field=None*, *value=None*, *count=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='FieldValueCount'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='FieldValueCount'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='FieldValueCount'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FieldValueCount'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_count()

get_field()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_count(*count*)

set_field(*field*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FieldValueCount'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_NUMBER(*value*)

class PMML44Super.**FinalNoise**(*Array=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='FinalNoise'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='FinalNoise'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='FinalNoise'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FinalNoise'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

hasContent_()

set_Array(*Array*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FinalNoise'*, *mapping_=None*)

class PMML44Super.**FinalNu**(*Array=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='FinalNu'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='FinalNu'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='FinalNu'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FinalNu'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

hasContent_()

set_Array(*Array*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FinalNu'*, *mapping_=None*)

class PMML44Super.**FinalOmega**(*Matrix=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='FinalOmega'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='FinalOmega'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='FinalOmega'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FinalOmega'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Matrix()

hasContent_()

set_Matrix(*Matrix*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FinalOmega'*, *mapping_=None*)

class PMML44Super.**FinalPredictedNoise**(*Array=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_='', name_='FinalPredictedNoise', namespacedef_='', pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='', name_='FinalPredictedNoise'*)

exportChildren(*outfile*, *level*, *namespace_='', name_='FinalPredictedNoise', fromsubclass_=False, pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FinalPredictedNoise'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

hasContent_()

set_Array(*Array*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FinalPredictedNoise'*, *mapping_=None*)

class PMML44Super.**FinalStateVector**(*Array=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_='', name_='FinalStateVector', namespacedef_='', pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='', name_='FinalStateVector'*)

exportChildren(*outfile*, *level*, *namespace_='', name_='FinalStateVector', fromsubclass_=False, pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FinalStateVector'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

hasContent_()

set_Array(*Array*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FinalStateVector'*, *mapping_=None*)

class PMML44Super.**FinalTheta**(*Theta=None*)

Bases: PMML44Super.GeneratedsSuper

add_Theta(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='FinalTheta'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='FinalTheta'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='FinalTheta'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='FinalTheta'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Theta()

hasContent_()

insert_Theta_at(*index*, *value*)

replace_Theta_at(*index*, *value*)

set_Theta(*Theta*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='FinalTheta'*, *mapping_=None*)

class PMML44Super.**GARCH**(*Extension=None*, *ARMAPart=None*, *GARCHPart=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='GARCH'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='GARCH'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='GARCH'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='GARCH'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ARMAPart()

get_Extension()

get_GARCHPart()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_ARMAPart(*ARMAPart*)

set_Extension(*Extension*)

set_GARCHPart(*GARCHPart*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='GARCH'*, *mapping_=None*)

class PMML44Super.**GARCHPart**(*constant='0'*, *gp=None*, *gq=None*, *Extension=None*, *ResidualSquareCoefficients=None*, *VarianceCoefficients=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='GARCHPart'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='GARCHPart'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='GARCHPart'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='GARCHPart'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_ResidualSquareCoefficients()

get_VarianceCoefficients()

get_constant()

get_gp()

get_gq()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_ResidualSquareCoefficients(*ResidualSquareCoefficients*)

set_VarianceCoefficients(*VarianceCoefficients*)

set_constant(*constant*)

set_gp(*gp*)

set_gq(*gq*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='GARCHPart'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**GaussianDistribution**(*mean=None*, *variance=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='GaussianDistribution'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='GaussianDistribution'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='GaussianDistribution'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='GaussianDistribution'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_mean()

get_variance()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_mean(*mean*)

set_variance(*variance*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='GaussianDistribution'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**GaussianProcessModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *optimizer=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *RadialBasisKernel=None*, *ARDSquaredExponentialKernel=None*, *AbsoluteExponentialKernel=None*, *GeneralizedExponentialKernel=None*, *TrainingInstances=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='GaussianProcessModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='GaussianProcessModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='GaussianProcessModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='GaussianProcessModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ARDSquaredExponentialKernel()

get_AbsoluteExponentialKernel()

get_Extension()

get_GeneralizedExponentialKernel()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_RadialBasisKernel()

get_Targets()

get_TrainingInstances()

get_algorithmName()

get_functionName()

get_isScorable()

get_modelName()

get_optimizer()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_ARDSquaredExponentialKernel(*ARDSquaredExponentialKernel*)

set_AbsoluteExponentialKernel(*AbsoluteExponentialKernel*)

set_Extension(*Extension*)

set_GeneralizedExponentialKernel(*GeneralizedExponentialKernel*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_RadialBasisKernel(*RadialBasisKernel*)

set_Targets(*Targets*)

set_TrainingInstances(*TrainingInstances*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

set_optimizer(*optimizer*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='GaussianProcessModel'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

class PMML44Super.**GeneralRegressionModel**(*targetVariableName=None*, *modelType=None*, *modelName=None*, *functionName=None*, *algorithmName=None*, *targetReferenceCategory=None*, *cumulativeLink=None*, *linkFunction=None*, *linkParameter=None*, *trialsVariable=None*, *trialsValue=None*, *distribution=None*, *distParameter=None*, *offsetVariable=None*, *offsetValue=None*, *modelDF=None*, *endTimeVariable=None*, *startTimeVariable=None*, *subjectIDVariable=None*, *statusVariable=None*, *baselineStrataVariable=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *ParameterList=None*, *FactorList=None*, *CovariateList=None*, *PPMatrix=None*, *PCovMatrix=None*, *ParamMatrix=None*, *EventValues=None*, *BaseCumHazardTables=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='GeneralRegressionModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='GeneralRegressionModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='GeneralRegressionModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='GeneralRegressionModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_BaseCumHazardTables()

get_CovariateList()

get_EventValues()

get_Extension()

get_FactorList()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_PCovMatrix()

get_PPMatrix()

get_ParamMatrix()

get_ParameterList()

get_Targets()

get_algorithmName()

get_baselineStrataVariable()

get_cumulativeLink()

get_distParameter()

get_distribution()

get_endTimeVariable()

get_functionName()

get_isScorable()

get_linkFunction()

get_linkParameter()

get_modelDF()

get_modelName()

get_modelType()

get_offsetValue()

get_offsetVariable()

get_startTimeVariable()

get_statusVariable()

get_subjectIDVariable()

get_targetReferenceCategory()

get_targetVariableName()

get_trialsValue()

get_trialsVariable()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_BaseCumHazardTables(*BaseCumHazardTables*)

set_CovariateList(*CovariateList*)

set_EventValues(*EventValues*)

set_Extension(*Extension*)

set_FactorList (*FactorList*)

set_LocalTransformations (*LocalTransformations*)

set_MiningSchema (*MiningSchema*)

set_ModelExplanation (*ModelExplanation*)

set_ModelStats (*ModelStats*)

set_ModelVerification (*ModelVerification*)

set_Output (*Output*)

set_PCovMatrix (*PCovMatrix*)

set_PPMatrix (*PPMatrix*)

set_ParamMatrix (*ParamMatrix*)

set_ParameterList (*ParameterList*)

set_Targets (*Targets*)

set_algorithmName (*algorithmName*)

set_baselineStrataVariable (*baselineStrataVariable*)

set_cumulativeLink (*cumulativeLink*)

set_distParameter (*distParameter*)

set_distribution (*distribution*)

set_endTimeVariable (*endTimeVariable*)

set_functionName (*functionName*)

set_isScorable (*isScorable*)

set_linkFunction (*linkFunction*)

set_linkParameter (*linkParameter*)

set_modelDF (*modelDF*)

set_modelName (*modelName*)

set_modelType (*modelType*)

set_offsetValue (*offsetValue*)

set_offsetVariable (*offsetVariable*)

set_startTimeVariable (*startTimeVariable*)

set_statusVariable (*statusVariable*)

set_subjectIDVariable (*subjectIDVariable*)

set_targetReferenceCategory (*targetReferenceCategory*)

set_targetVariableName (*targetVariableName*)

set_trialsValue (*trialsValue*)

set_trialsVariable (*trialsVariable*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='GeneralRegressionModel'*, *mapping_=None*)

validate_CUMULATIVE_LINK_FUNCTION(*value*)

validate_FIELD_NAME(*value*)

validate_INT_NUMBER(*value*)

validate_LINK_FUNCTION(*value*)

validate_MINING_FUNCTION(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**GeneralizedExponentialKernel**(*description=None*, *gamma='1'*, *noiseVariance='1'*, *degree='1'*, *Extension=None*, *Lambda=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Lambda(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='GeneralizedExponentialKernel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='GeneralizedExponentialKernel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='GeneralizedExponentialKernel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='GeneralizedExponentialKernel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Lambda()

get_degree()

get_description()

get_gamma()

get_noiseVariance()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Lambda_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Lambda_at(*index*, *value*)

set_Extension(*Extension*)

set_Lambda(*Lambda*)

set_degree(*degree*)

set_description(*description*)

set_gamma(*gamma*)

set_noiseVariance(*noiseVariance*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='GeneralizedExponentialKernel'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**HVector**(*Array=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='HVector'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='HVector'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='HVector'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='HVector'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

hasContent_()

set_Array(*Array*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='HVector'*, *mapping_=None*)

class PMML44Super.**Header**(*copyright=None*, *description=None*, *modelVersion=None*, *Extension=None*, *Application=None*, *Annotation=None*, *Timestamp=None*)

Bases: PMML44Super.GeneratedsSuper

add_Annotation(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Header'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Header'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Header'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Header'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Annotation()

get_Application()

get_Extension()

get_Timestamp()

get_copyright()

get_description()

get_modelVersion()

hasContent_()

insert_Annotation_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_Annotation_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Annotation(*Annotation*)

set_Application(*Application*)

set_Extension(*Extension*)

set_Timestamp(*Timestamp*)

set_copyright(*copyright*)

set_description(*description*)

set_modelVersion(*modelVersion*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Header'*, *mapping_=None*)

class PMML44Super.**INT_Entries**

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='INT-Entries'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='INT-Entries'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='INT-Entries'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='INT-Entries'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

hasContent_()

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='INT-Entries'*, *mapping_=None*)

class PMML44Super.**INT_SparseArray**(*n=None*, *defaultValue='0'*, *Indices=None*, *INT_Entries=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='INT-SparseArray'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='INT-SparseArray'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='INT-SparseArray'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='INT-SparseArray'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_INT_Entries()

get_Indices()

get_defaultValue()

get_n()

hasContent_()

set_INT_Entries(*INT_Entries*)

set_Indices(*Indices*)

set_defaultValue(*defaultValue*)

set_n(*n*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='INT-SparseArray'*, *mapping_=None*)

validate_INT_Entries(*value*)

validate_INT_NUMBER(*value*)

validate_Indices(*value*)

class PMML44Super.**Indices**

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Indices'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Indices'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Indices'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Indices'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

hasContent_()

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Indices'*, *mapping_=None*)

class PMML44Super.**InlineTable**(*Extension=None*, *row=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_row(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='InlineTable'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='InlineTable'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='InlineTable'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='InlineTable'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_row()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_row_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_row_at(*index*, *value*)

set_Extension(*Extension*)

set_row(*row*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='InlineTable'*, *mapping_=None*)

class PMML44Super.**InstanceField**(*field=None*, *column=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='InstanceField'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='InstanceField'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='InstanceField'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='InstanceField'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_column()

get_field()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_column(*column*)

set_field(*field*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='InstanceField'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**InstanceFields**(*Extension=None*, *InstanceField=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_InstanceField(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='InstanceFields'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='InstanceFields'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='InstanceFields'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='InstanceFields'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_InstanceField()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_InstanceField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_InstanceField_at(*index*, *value*)

set_Extension(*Extension*)

set_InstanceField(*InstanceField*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='InstanceFields'*, *mapping_=None*)

class PMML44Super.**Interval**(*closure=None*, *leftMargin=None*, *rightMargin=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Interval'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Interval'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Interval'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Interval'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_closure()

get_leftMargin()

get_rightMargin()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_closure(*closure*)

set_leftMargin(*leftMargin*)

set_rightMargin(*rightMargin*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Interval'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**Item**(*id=None*, *value=None*, *field=None*, *category=None*, *mappedValue=None*, *weight=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Item'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Item'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Item'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Item'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_category()

get_field()

get_id()

get_mappedValue()

get_value()

get_weight()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_category(*category*)

set_field(*field*)

set_id(*id*)

set_mappedValue(*mappedValue*)

set_value(*value*)

set_weight(*weight*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Item'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ItemRef**(*itemRef=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ItemRef'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ItemRef'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ItemRef'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ItemRef'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_itemRef()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_itemRef(*itemRef*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ItemRef'*, *mapping_=None*)

class PMML44Super.**Itemset**(*id=None*, *support=None*, *numberOfItems=None*, *Extension=None*, *ItemRef=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_ItemRef(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Itemset'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Itemset'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Itemset'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Itemset'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_ItemRef()

get_id()

get_numberOfItems()

get_support()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_ItemRef_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_ItemRef_at(*index*, *value*)

set_Extension(*Extension*)

set_ItemRef(*ItemRef*)

set_id(*id*)

set_numberOfItems(*numberOfItems*)

set_support(*support*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Itemset'*, *mapping_=None*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**KNNInput**(*field=None*, *fieldWeight='1'*, *compareFunction=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='KNNInput'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='KNNInput'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='KNNInput'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='KNNInput'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_compareFunction()

get_field()

get_fieldWeight()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_compareFunction(*compareFunction*)

set_field(*field*)

set_fieldWeight(*fieldWeight*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='KNNInput'*, *mapping_=None*)

validate_COMPARE_FUNCTION(*value*)

validate_FIELD_NAME(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**KNNInputs**(*Extension=None*, *KNNInput=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_KNNInput(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='KNNInputs'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='KNNInputs'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='KNNInputs'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='KNNInputs'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_KNNInput()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_KNNInput_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_KNNInput_at(*index*, *value*)

set_Extension(*Extension*)

set_KNNInput(*KNNInput*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='KNNInputs'*, *mapping_=None*)

class PMML44Super.**KalmanState**(*FinalOmega=None*, *FinalStateVector=None*, *HVector=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='KalmanState'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='KalmanState'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='KalmanState'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='KalmanState'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_FinalOmega()

get_FinalStateVector()

get_HVector()

hasContent_()

set_FinalOmega(*FinalOmega*)

set_FinalStateVector(*FinalStateVector*)

set_HVector(*HVector*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='KalmanState'*, *mapping_=None*)

class PMML44Super.**KohonenMap**(*coord1=None*, *coord2=None*, *coord3=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='KohonenMap'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='KohonenMap'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='KohonenMap'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='KohonenMap'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_coord1()

get_coord2()

get_coord3()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_coord1(*coord1*)

set_coord2(*coord2*)

set_coord3(*coord3*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='KohonenMap'*, *mapping_=None*)

class PMML44Super.**Lag**(*field=None*, *n=1*, *aggregate='none'*, *Extension=None*, *BlockIndicator=None*)

Bases: PMML44Super.GeneratedsSuper

add_BlockIndicator(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Lag'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Lag'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Lag'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Lag'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_BlockIndicator()

get_Extension()

get_aggregate()

get_field()

get_n()

hasContent_()

insert_BlockIndicator_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_BlockIndicator_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_BlockIndicator(*BlockIndicator*)

set_Extension(*Extension*)

set_aggregate(*aggregate*)

set_field(*field*)

set_n(*n*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Lag'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**Lambda**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Lambda'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Lambda'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Lambda'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Lambda'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Lambda'*, *mapping_=None*)

class PMML44Super.**LayerBias**(*biasShape=None*, *biasFlattenAxis=None*, *Extension=None*, *valueOf_=None*, *mixedclass_=None*, *content_=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='LayerBias'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='LayerBias'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='LayerBias'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='LayerBias'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_biasFlattenAxis()

get_biasShape()

get_valueOf_()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_biasFlattenAxis(*biasFlattenAxis*)

set_biasShape(*biasShape*)

set_valueOf_(*valueOf_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='LayerBias'*, *mapping_=None*)

class PMML44Super.**LayerParameters**(*activationFunction=None*, *inputDimension=None*, *outputDimension=None*, *featureMaps=None*, *kernel=None*, *pad=None*, *stride=None*, *dilationRate=None*, *poolSize=None*, *depthMultiplier=None*, *paddingDims=None*, *croppingDims=None*, *upsamplingSize=None*, *batchNormalizationEpsilon=None*, *flattenAxis=None*, *batchNormalizationAxis=None*, *batchNormalizationMomentum=None*, *batchNormalizationCenter=None*, *batchNormalizationScale=None*, *gaussianNoiseStdev=None*, *gaussianDropoutRate=None*, *alphaDropoutRate=None*, *alphaDropoutSeed=None*, *betaInitializer=None*, *gammaInitializer=None*, *movingMeanInitializer=None*, *movingVarianceInitializer=None*, *betaRegularizer=None*, *gammaRegularizer=None*, *betaConstraint=None*, *gammaConstraint=None*, *kernelInitializer=None*, *biasInitializer=None*, *kernelRegularizer=None*, *biasRegularizer=None*, *kernelConstraint=None*, *biasConstraint=None*, *depthwiseConstraint=None*, *pointwiseConstraint=None*, *batchSize=None*, *dropoutRate=None*, *dropoutNoiseShape=None*, *dropoutSeed=None*, *generalLUAlpha=None*, *reshapeTarget=None*, *permuteDims=None*, *repeatVectorTimes=None*, *activityRegularizerL1=None*, *activityRegularizerL2=None*, *maskValue=None*, *mergeLayerOp=None*, *mergeLayerDotOperationAxis=None*, *mergeLayerDotNormalize=None*, *mergeLayerConcatOperationAxes=None*, *slicingAxis=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='LayerParameters'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='LayerParameters'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='LayerParameters'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='LayerParameters'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_activationFunction()

get_activityRegularizerL1()

get_activityRegularizerL2()

get_alphaDropoutRate()

get_alphaDropoutSeed()

get_batchNormalizationAxis()

get_batchNormalizationCenter()

get_batchNormalizationEpsilon()

get_batchNormalizationMomentum()

get_batchNormalizationScale()

get_batchSize()

get_betaConstraint()

get_betaInitializer()

get_betaRegularizer()

get_biasConstraint()

get_biasInitializer()

get_biasRegularizer()

get_croppingDims()

get_depthMultiplier()

get_depthwiseConstraint()

get_dilationRate()

get_dropoutNoiseShape()

get_dropoutRate()

get_dropoutSeed()

get_featureMaps()

get_flattenAxis()

get_gammaConstraint()

get_gammaInitializer()

get_gammaRegularizer()

get_gaussianDropoutRate()

get_gaussianNoiseStdev()

get_generalLUAlpha()

get_inputDimension()

get_kernel()

get_kernelConstraint()

get_kernelInitializer()

get_kernelRegularizer()

get_maskValue()

get_mergeLayerConcatOperationAxes()

get_mergeLayerDotNormalize()

get_mergeLayerDotOperationAxis()

get_mergeLayerOp()

get_movingMeanInitializer()

get_movingVarianceInitializer()

get_outputDimension()

get_pad()

get_paddingDims()

get_permuteDims()

get_pointwiseConstraint()

get_poolSize()

get_repeatVectorTimes()

get_reshapeTarget()

get_slicingAxis()

get_stride()

get_upsamplingSize()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_activationFunction(*activationFunction*)

set_activityRegularizerL1(*activityRegularizerL1*)

set_activityRegularizerL2(*activityRegularizerL2*)

set_alphaDropoutRate(*alphaDropoutRate*)

set_alphaDropoutSeed(*alphaDropoutSeed*)

set_batchNormalizationAxis(*batchNormalizationAxis*)

set_batchNormalizationCenter(*batchNormalizationCenter*)

set_batchNormalizationEpsilon(*batchNormalizationEpsilon*)

set_batchNormalizationMomentum(*batchNormalizationMomentum*)

set_batchNormalizationScale(*batchNormalizationScale*)

set_batchSize(*batchSize*)

set_betaConstraint(*betaConstraint*)

set_betaInitializer(*betaInitializer*)

set_betaRegularizer(*betaRegularizer*)

set_biasConstraint(*biasConstraint*)

set_biasInitializer(*biasInitializer*)

set_biasRegularizer(*biasRegularizer*)

set_croppingDims(*croppingDims*)

set_depthMultiplier(*depthMultiplier*)

set_depthwiseConstraint(*depthwiseConstraint*)

set_dilationRate(*dilationRate*)

set_dropoutNoiseShape(*dropoutNoiseShape*)

set_dropoutRate(*dropoutRate*)

set_dropoutSeed(*dropoutSeed*)

set_featureMaps(*featureMaps*)

set_flattenAxis(*flattenAxis*)

set_gammaConstraint(*gammaConstraint*)

set_gammaInitializer(*gammaInitializer*)

set_gammaRegularizer(*gammaRegularizer*)

set_gaussianDropoutRate(*gaussianDropoutRate*)

set_gaussianNoiseStdev(*gaussianNoiseStdev*)

set_generalLUAlpha(*generalLUAlpha*)

set_inputDimension(*inputDimension*)

set_kernel(*kernel*)

set_kernelConstraint(*kernelConstraint*)

set_kernelInitializer(*kernelInitializer*)

set_kernelRegularizer(*kernelRegularizer*)

set_maskValue(*maskValue*)

set_mergeLayerConcatOperationAxes(*mergeLayerConcatOperationAxes*)

set_mergeLayerDotNormalize(*mergeLayerDotNormalize*)

set_mergeLayerDotOperationAxis(*mergeLayerDotOperationAxis*)

set_mergeLayerOp(*mergeLayerOp*)

set_movingMeanInitializer(*movingMeanInitializer*)

set_movingVarianceInitializer(*movingVarianceInitializer*)

set_outputDimension(*outputDimension*)

set_pad(*pad*)

set_paddingDims(*paddingDims*)

set_permuteDims(*permuteDims*)

set_pointwiseConstraint(*pointwiseConstraint*)

set_poolSize(*poolSize*)

set_repeatVectorTimes(*repeatVectorTimes*)

set_reshapeTarget(*reshapeTarget*)

set_slicingAxis(*slicingAxis*)

set_stride(*stride*)

set_upsamplingSize(*upsamplingSize*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='LayerParameters'*, *mapping_=None*)

validate_DNN_ACTIVATION_FUNCTION(*value*)

validate_INT_NUMBER(*value*)

validate_LAYEROP_TYPE(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**LayerWeights**(*weightsShape=None*, *weightsFlattenAxis=None*, *Extension=None*, *valueOf_=None*, *mixedclass_=None*, *content_=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='LayerWeights'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='LayerWeights'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='LayerWeights'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='LayerWeights'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_valueOf_()

get_weightsFlattenAxis()

get_weightsShape()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_valueOf_(*valueOf_*)

set_weightsFlattenAxis(*weightsFlattenAxis*)

set_weightsShape(*weightsShape*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='LayerWeights'*, *mapping_=None*)

class PMML44Super.**Level**(*alpha=None*, *smoothedValue=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Level'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Level'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Level'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Level'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_alpha()

get_smoothedValue()

hasContent_()

set_alpha(*alpha*)

set_smoothedValue(*smoothedValue*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Level'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**LiftData**(*targetFieldValue=None*, *targetFieldDisplayValue=None*, *rankingQuality=None*, *Extension=None*, *ModelLiftGraph=None*, *OptimumLiftGraph=None*, *RandomLiftGraph=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='LiftData'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='LiftData'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='LiftData'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='LiftData'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_ModelLiftGraph()

get_OptimumLiftGraph()

get_RandomLiftGraph()

get_rankingQuality()

get_targetFieldDisplayValue()

get_targetFieldValue()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_ModelLiftGraph(*ModelLiftGraph*)

set_OptimumLiftGraph(*OptimumLiftGraph*)

set_RandomLiftGraph(*RandomLiftGraph*)

set_rankingQuality(*rankingQuality*)

set_targetFieldDisplayValue(*targetFieldDisplayValue*)

set_targetFieldValue(*targetFieldValue*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='LiftData'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**LiftGraph**(*Extension=None*, *XCoordinates=None*, *YCoordinates=None*, *BoundaryValues=None*, *BoundaryValueMeans=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='LiftGraph'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='LiftGraph'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='LiftGraph'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='LiftGraph'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_BoundaryValueMeans()

get_BoundaryValues()

get_Extension()

get_XCoordinates()

get_YCoordinates()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_BoundaryValueMeans(*BoundaryValueMeans*)

set_BoundaryValues(*BoundaryValues*)

set_Extension(*Extension*)

set_XCoordinates(*XCoordinates*)

set_YCoordinates(*YCoordinates*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='LiftGraph'*, *mapping_=None*)

class PMML44Super.**LinearKernelType**(*description=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='LinearKernelType'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='LinearKernelType'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='LinearKernelType'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='LinearKernelType'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_description()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_description(*description*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='LinearKernelType'*, *mapping_=None*)

class PMML44Super.**LinearNorm**(*orig=None*, *norm=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='LinearNorm'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='LinearNorm'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='LinearNorm'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='LinearNorm'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_norm()

get_orig()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_norm(*norm*)

set_orig(*orig*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='LinearNorm'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**LocalTransformations**(*Extension=None*, *DerivedField=None*)

Bases: PMML44Super.GeneratedsSuper

add_DerivedField(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='LocalTransformations'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='LocalTransformations'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='LocalTransformations'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='LocalTransformations'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DerivedField()

get_Extension()

hasContent_()

insert_DerivedField_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_DerivedField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_DerivedField(*DerivedField*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='LocalTransformations'*, *mapping_=None*)

class PMML44Super.**LognormalDistributionForBN**(*Extension=None*, *Mean=None*, *Variance=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='LognormalDistributionForBN'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='LognormalDistributionForBN'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='LognormalDistributionForBN'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='LognormalDistributionForBN'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Mean()

get_Variance()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Mean(*Mean*)

set_Variance(*Variance*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='LognormalDistributionForBN'*, *mapping_=None*)

class PMML44Super.**Losses**(*loss=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Losses'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Losses'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Losses'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Losses'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_loss()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_loss(*loss*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Losses'*, *mapping_=None*)

validate_LOSS_TYPE(*value*)

class PMML44Super.**Lower**(*Extension=None*, *FieldRef=None*, *Constant=None*, *NormContinuous=None*, *NormDiscrete=None*, *Discretize=None*, *MapValues=None*, *TextIndex=None*, *Apply=None*, *Aggregate=None*, *Lag=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Lower'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Lower'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Lower'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Lower'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Aggregate()

get_Apply()

get_Constant()

get_Discretize()

get_Extension()

get_FieldRef()

get_Lag()

get_MapValues()

get_NormContinuous()

get_NormDiscrete()

get_TextIndex()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Aggregate(*Aggregate*)

set_Apply(*Apply*)

set_Constant(*Constant*)

set_Discretize(*Discretize*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_Lag(*Lag*)

set_MapValues(*MapValues*)

set_NormContinuous(*NormContinuous*)

set_NormDiscrete(*NormDiscrete*)

set_TextIndex(*TextIndex*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Lower'*, *mapping_=None*)

class PMML44Super.**MA**(*Extension=None*, *MACoefficients=None*, *Residuals=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MA'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MA'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MA'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MA'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_MACoefficients()

get_Residuals()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_MACoefficients(*MACoefficients*)

set_Residuals(*Residuals*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MA'*, *mapping_=None*)

class PMML44Super.**MACoefficients**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MACoefficients'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MACoefficients'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MACoefficients'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MACoefficients'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MACoefficients'*, *mapping_=None*)

class PMML44Super.**MapValues**(*mapMissingTo=None*, *defaultValue=None*, *outputColumn=None*, *dataType=None*, *Extension=None*, *FieldColumnPair=None*, *TableLocator=None*, *InlineTable=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_FieldColumnPair(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MapValues'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MapValues'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MapValues'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MapValues'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_FieldColumnPair()

get_InlineTable()

get_TableLocator()

get_dataType()

get_defaultValue()

get_mapMissingTo()

get_outputColumn()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_FieldColumnPair_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_FieldColumnPair_at(*index*, *value*)

set_Extension(*Extension*)

set_FieldColumnPair(*FieldColumnPair*)

set_InlineTable(*InlineTable*)

set_TableLocator(*TableLocator*)

set_dataType(*dataType*)

set_defaultValue(*defaultValue*)

set_mapMissingTo(*mapMissingTo*)

set_outputColumn(*outputColumn*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MapValues'*, *mapping_=None*)

validate_DATATYPE(*value*)

class PMML44Super.**MatCell**(*row=None*, *col=None*, *valueOf_=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MatCell'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MatCell'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MatCell'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MatCell'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_col()

get_row()

get_valueOf_()

hasContent_()

set_col(*col*)

set_row(*row*)

set_valueOf_(*valueOf_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MatCell'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**Matrix**(*kind='any'*, *nbRows=None*, *nbCols=None*, *diagDefault=None*, *offDiagDefault=None*, *Array=None*, *MatCell=None*)

Bases: PMML44Super.GeneratedsSuper

add_Array(*value*)

add_MatCell(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Matrix'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Matrix'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Matrix'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Matrix'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_MatCell()

get_diagDefault()

get_kind()

get_nbCols()

get_nbRows()

get_offDiagDefault()

hasContent_()

insert_Array_at(*index*, *value*)

insert_MatCell_at(*index*, *value*)

replace_Array_at(*index*, *value*)

replace_MatCell_at(*index*, *value*)

set_Array(*Array*)

set_MatCell(*MatCell*)

set_diagDefault(*diagDefault*)

set_kind(*kind*)

set_nbCols(*nbCols*)

set_nbRows(*nbRows*)

set_offDiagDefault(*offDiagDefault*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Matrix'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**MaximumLikelihoodStat**(*method=None*, *periodDeficit='0'*, *KalmanState=None*, *ThetaRecursionState=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MaximumLikelihoodStat'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MaximumLikelihoodStat'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MaximumLikelihoodStat'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MaximumLikelihoodStat'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_KalmanState()

get_ThetaRecursionState()

get_method()

get_periodDeficit()

hasContent_()

set_KalmanState(*KalmanState*)

set_ThetaRecursionState(*ThetaRecursionState*)

set_method(*method*)

set_periodDeficit(*periodDeficit*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MaximumLikelihoodStat'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**Mean**(*Extension=None*, *FieldRef=None*, *Constant=None*, *NormContinuous=None*, *NormDiscrete=None*, *Discretize=None*, *MapValues=None*, *TextIndex=None*, *Apply=None*, *Aggregate=None*, *Lag=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Mean'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Mean'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Mean'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Mean'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Aggregate()

get_Apply()

get_Constant()

get_Discretize()

get_Extension()

get_FieldRef()

get_Lag()

get_MapValues()

get_NormContinuous()

get_NormDiscrete()

get_TextIndex()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Aggregate(*Aggregate*)

set_Apply(*Apply*)

set_Constant(*Constant*)

set_Discretize(*Discretize*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_Lag(*Lag*)

set_MapValues(*MapValues*)

set_NormContinuous(*NormContinuous*)

set_NormDiscrete(*NormDiscrete*)

set_TextIndex(*TextIndex*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Mean'*, *mapping_=None*)

class PMML44Super.**MeasurementMatrix**(*Extension=None*, *Matrix=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MeasurementMatrix'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MeasurementMatrix'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MeasurementMatrix'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MeasurementMatrix'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Matrix()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Matrix(*Matrix*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MeasurementMatrix'*, *mapping_=None*)

class PMML44Super.**Metrics**(*top_k_categories_for_accuracy=None*, *metric=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Metrics'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Metrics'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Metrics'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Metrics'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_metric()

get_top_k_categories_for_accuracy()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_metric(*metric*)

set_top_k_categories_for_accuracy(*top_k_categories_for_accuracy*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Metrics'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_METRIC_TYPE(*value*)

class PMML44Super.**MiningBuildTask**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MiningBuildTask'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MiningBuildTask'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MiningBuildTask'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MiningBuildTask'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MiningBuildTask'*, *mapping_=None*)

class PMML44Super.**MiningField**(*name=None*, *usageType='active'*, *optype=None*, *importance=None*, *outliers='asIs'*, *lowValue=None*, *highValue=None*, *missingValueReplacement=None*, *missingValueTreatment=None*, *invalidValueTreatment='returnInvalid'*, *invalidValueReplacement=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MiningField'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MiningField'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MiningField'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MiningField'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_highValue()

get_importance()

get_invalidValueReplacement()

get_invalidValueTreatment()

get_lowValue()

get_missingValueReplacement()

get_missingValueTreatment()

get_name()

get_optype()

get_outliers()

get_usageType()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_highValue(*highValue*)

set_importance(*importance*)

set_invalidValueReplacement(*invalidValueReplacement*)

set_invalidValueTreatment(*invalidValueTreatment*)

set_lowValue(*lowValue*)

set_missingValueReplacement(*missingValueReplacement*)

set_missingValueTreatment(*missingValueTreatment*)

set_name(*name*)

set_optype(*optype*)

set_outliers(*outliers*)

set_usageType(*usageType*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MiningField'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_FIELD_USAGE_TYPE(*value*)

validate_INVALID_VALUE_TREATMENT_METHOD(*value*)

validate_MISSING_VALUE_TREATMENT_METHOD(*value*)

validate_NUMBER(*value*)

validate_OPTYPE(*value*)

validate_OUTLIER_TREATMENT_METHOD(*value*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**MiningModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *Regression=None*, *DecisionTree=None*, *Segmentation=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_DecisionTree(*value*)

add_Extension(*value*)

add_Regression(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MiningModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MiningModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MiningModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MiningModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DecisionTree()

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_Regression()

get_Segmentation()

get_Targets()

get_algorithmName()

get_functionName()

get_isScorable()

get_modelName()

hasContent_()

insert_DecisionTree_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_Regression_at(*index*, *value*)

replace_DecisionTree_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Regression_at(*index*, *value*)

set_DecisionTree(*DecisionTree*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_Regression(*Regression*)

set_Segmentation(*Segmentation*)

set_Targets(*Targets*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MiningModel'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

class PMML44Super.**MiningSchema**(*Extension=None*, *MiningField=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_MiningField(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MiningSchema'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MiningSchema'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MiningSchema'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MiningSchema'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_MiningField()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_MiningField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_MiningField_at(*index*, *value*)

set_Extension(*Extension*)

set_MiningField(*MiningField*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MiningSchema'*, *mapping_=None*)

class PMML44Super.**MissingValueWeights**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MissingValueWeights'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MissingValueWeights'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MissingValueWeights'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MissingValueWeights'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MissingValueWeights'*, *mapping_=None*)

class PMML44Super.**ModelExplanation**(*Extension=None*, *PredictiveModelQuality=None*, *ClusteringModelQuality=None*, *Correlations=None*)

Bases: PMML44Super.GeneratedsSuper

add_ClusteringModelQuality(*value*)

add_Extension(*value*)

add_PredictiveModelQuality(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ModelExplanation'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ModelExplanation'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ModelExplanation'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ModelExplanation'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ClusteringModelQuality()

get_Correlations()

get_Extension()

get_PredictiveModelQuality()

hasContent_()

insert_ClusteringModelQuality_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_PredictiveModelQuality_at(*index*, *value*)

replace_ClusteringModelQuality_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_PredictiveModelQuality_at(*index*, *value*)

set_ClusteringModelQuality(*ClusteringModelQuality*)

set_Correlations(*Correlations*)

set_Extension(*Extension*)

set_PredictiveModelQuality(*PredictiveModelQuality*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ModelExplanation'*, *mapping_=None*)

class PMML44Super.**ModelLiftGraph**(*Extension=None*, *LiftGraph=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ModelLiftGraph'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ModelLiftGraph'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ModelLiftGraph'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ModelLiftGraph'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LiftGraph()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_LiftGraph(*LiftGraph*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ModelLiftGraph'*, *mapping_=None*)

class PMML44Super.**ModelStats**(*Extension=None*, *UnivariateStats=None*, *MultivariateStats=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_MultivariateStats(*value*)

add_UnivariateStats(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ModelStats'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ModelStats'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ModelStats'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ModelStats'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_MultivariateStats()

get_UnivariateStats()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_MultivariateStats_at(*index*, *value*)

insert_UnivariateStats_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_MultivariateStats_at(*index*, *value*)

replace_UnivariateStats_at(*index*, *value*)

set_Extension(*Extension*)

set_MultivariateStats(*MultivariateStats*)

set_UnivariateStats(*UnivariateStats*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ModelStats'*, *mapping_=None*)

class PMML44Super.**ModelVerification**(*recordCount=None*, *fieldCount=None*, *Extension=None*, *VerificationFields=None*, *InlineTable=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ModelVerification'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ModelVerification'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ModelVerification'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ModelVerification'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_InlineTable()

get_VerificationFields()

get_fieldCount()

get_recordCount()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_InlineTable(*InlineTable*)

set_VerificationFields(*VerificationFields*)

set_fieldCount(*fieldCount*)

set_recordCount(*recordCount*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ModelVerification'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**MultivariateStat**(*name=None*, *category=None*, *exponent='1'*, *isIntercept=False*, *importance=None*, *stdError=None*, *tValue=None*, *chiSquareValue=None*, *fStatistic=None*, *dF=None*, *pValueAlpha=None*, *pValueInitial=None*, *pValueFinal=None*, *confidenceLevel='0.95'*, *confidenceLowerBound=None*, *confidenceUpperBound=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MultivariateStat'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MultivariateStat'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MultivariateStat'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MultivariateStat'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_category()

get_chiSquareValue()

get_confidenceLevel()

get_confidenceLowerBound()

get_confidenceUpperBound()

get_dF()

get_exponent()

get_fStatistic()

get_importance()

get_isIntercept()

get_name()

get_pValueAlpha()

get_pValueFinal()

get_pValueInitial()

get_stdError()

get_tValue()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_category(*category*)

set_chiSquareValue(*chiSquareValue*)

set_confidenceLevel(*confidenceLevel*)

set_confidenceLowerBound(*confidenceLowerBound*)

set_confidenceUpperBound(*confidenceUpperBound*)

set_dF(*dF*)

set_exponent(*exponent*)

set_fStatistic(*fStatistic*)

set_importance(*importance*)

set_isIntercept(*isIntercept*)

set_name(*name*)

set_pValueAlpha(*pValueAlpha*)

set_pValueFinal(*pValueFinal*)

set_pValueInitial(*pValueInitial*)

set_stdError(*stdError*)

set_tValue(*tValue*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MultivariateStat'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_NUMBER(*value*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**MultivariateStats**(*targetCategory=None*, *Extension=None*, *MultivariateStat=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_MultivariateStat(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='MultivariateStats'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='MultivariateStats'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='MultivariateStats'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='MultivariateStats'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_MultivariateStat()

get_targetCategory()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_MultivariateStat_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_MultivariateStat_at(*index*, *value*)

set_Extension(*Extension*)

set_MultivariateStat(*MultivariateStat*)

set_targetCategory(*targetCategory*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='MultivariateStats'*, *mapping_=None*)

class PMML44Super.**Nadam**(*learningRate=None*, *beta_1=None*, *beta_2=None*, *schedule_decay=None*, *epsilon=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Nadam'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Nadam'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Nadam'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Nadam'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_beta_1()

get_beta_2()

get_epsilon()

get_learningRate()

get_schedule_decay()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_beta_1(*beta_1*)

set_beta_2(*beta_2*)

set_epsilon(*epsilon*)

set_learningRate(*learningRate*)

set_schedule_decay(*schedule_decay*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Nadam'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**NaiveBayesModel**(*modelName=None*, *threshold=None*, *functionName=None*, *algorithmName=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *BayesInputs=None*, *BayesOutput=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NaiveBayesModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NaiveBayesModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NaiveBayesModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NaiveBayesModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_BayesInputs()

get_BayesOutput()

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_Targets()

get_algorithmName()

get_functionName()

get_isScorable()

get_modelName()

get_threshold()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_BayesInputs(*BayesInputs*)

set_BayesOutput(*BayesOutput*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_Targets(*Targets*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

set_threshold(*threshold*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NaiveBayesModel'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**NearestNeighborModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *numberOfNeighbors=None*, *continuousScoringMethod='average'*, *categoricalScoringMethod='majorityVote'*, *instanceIdVariable=None*, *threshold='0.001'*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *TrainingInstances=None*, *ComparisonMeasure=None*, *KNNInputs=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NearestNeighborModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NearestNeighborModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NearestNeighborModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NearestNeighborModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ComparisonMeasure()

get_Extension()

get_KNNInputs()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_Targets()

get_TrainingInstances()

get_algorithmName()

get_categoricalScoringMethod()

get_continuousScoringMethod()

get_functionName()

get_instanceIdVariable()

get_isScorable()

get_modelName()

get_numberOfNeighbors()

get_threshold()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_ComparisonMeasure(*ComparisonMeasure*)

set_Extension(*Extension*)

set_KNNInputs(*KNNInputs*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_Targets(*Targets*)

set_TrainingInstances(*TrainingInstances*)

set_algorithmName(*algorithmName*)

set_categoricalScoringMethod(*categoricalScoringMethod*)

set_continuousScoringMethod(*continuousScoringMethod*)

set_functionName(*functionName*)

set_instanceIdVariable(*instanceIdVariable*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

set_numberOfNeighbors(*numberOfNeighbors*)

set_threshold(*threshold*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NearestNeighborModel'*, *mapping_=None*)

validate_CAT_SCORING_METHOD(*value*)

validate_CONT_SCORING_METHOD(*value*)

validate_FIELD_NAME(*value*)

validate_INT_NUMBER(*value*)

validate_MINING_FUNCTION(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**NetworkLayer**(*normalizationMethod='none'*, *layerType=None*, *layerId=None*, *connectionLayerId=None*, *inputFieldName=None*, *Extension=None*, *LayerParameters=None*, *LayerWeights=None*, *LayerBias=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NetworkLayer'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NetworkLayer'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NetworkLayer'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NetworkLayer'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LayerBias()

get_LayerParameters()

get_LayerWeights()

get_connectionLayerId()

get_inputFieldName()

get_layerId()

get_layerType()

get_normalizationMethod()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_LayerBias(*LayerBias*)

set_LayerParameters(*LayerParameters*)

set_LayerWeights(*LayerWeights*)

set_connectionLayerId(*connectionLayerId*)

set_inputFieldName(*inputFieldName*)

set_layerId(*layerId*)

set_layerType(*layerType*)

set_normalizationMethod(*normalizationMethod*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NetworkLayer'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_LAYER_TYPE(*value*)

validate_NN_NORMALIZATION_METHOD(*value*)

class PMML44Super.**NeuralInput**(*id=None*, *Extension=None*, *DerivedField=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NeuralInput'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NeuralInput'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NeuralInput'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NeuralInput'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DerivedField()

get_Extension()

get_id()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_DerivedField(*DerivedField*)

set_Extension(*Extension*)

set_id(*id*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NeuralInput'*, *mapping_=None*)

validate_NN_NEURON_ID(*value*)

class PMML44Super.**NeuralInputs**(*numberOfInputs=None*, *Extension=None*, *NeuralInput=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_NeuralInput(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NeuralInputs'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NeuralInputs'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NeuralInputs'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NeuralInputs'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_NeuralInput()

get_numberOfInputs()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_NeuralInput_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_NeuralInput_at(*index*, *value*)

set_Extension(*Extension*)

set_NeuralInput(*NeuralInput*)

set_numberOfInputs(*numberOfInputs*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NeuralInputs'*, *mapping_=None*)

class PMML44Super.**NeuralLayer**(*numberOfNeurons=None*, *activationFunction=None*, *threshold=None*, *width=None*, *altitude=None*, *normalizationMethod=None*, *Extension=None*, *Neuron=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Neuron(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NeuralLayer'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NeuralLayer'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NeuralLayer'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NeuralLayer'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Neuron()

get_activationFunction()

get_altitude()

get_normalizationMethod()

get_numberOfNeurons()

get_threshold()

get_width()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Neuron_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Neuron_at(*index*, *value*)

set_Extension(*Extension*)

set_Neuron(*Neuron*)

set_activationFunction(*activationFunction*)

set_altitude(*altitude*)

set_normalizationMethod(*normalizationMethod*)

set_numberOfNeurons(*numberOfNeurons*)

set_threshold(*threshold*)

set_width(*width*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NeuralLayer'*, *mapping_=None*)

validate_ACTIVATION_FUNCTION(*value*)

validate_NN_NORMALIZATION_METHOD(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**NeuralNetwork**(*modelName=None*, *functionName=None*, *algorithmName=None*, *activationFunction=None*, *normalizationMethod='none'*, *threshold='0'*, *width=None*, *altitude='1.0'*, *numberOfLayers=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *NeuralInputs=None*, *NeuralLayer=None*, *NeuralOutputs=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_NeuralLayer(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NeuralNetwork'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NeuralNetwork'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NeuralNetwork'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NeuralNetwork'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_NeuralInputs()

get_NeuralLayer()

get_NeuralOutputs()

get_Output()

get_Targets()

get_activationFunction()

get_algorithmName()

get_altitude()

get_functionName()

get_isScorable()

get_modelName()

get_normalizationMethod()

get_numberOfLayers()

get_threshold()

get_width()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_NeuralLayer_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_NeuralLayer_at(*index*, *value*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_NeuralInputs(*NeuralInputs*)

set_NeuralLayer(*NeuralLayer*)

set_NeuralOutputs(*NeuralOutputs*)

set_Output(*Output*)

set_Targets(*Targets*)

set_activationFunction(*activationFunction*)

set_algorithmName(*algorithmName*)

set_altitude(*altitude*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

set_normalizationMethod(*normalizationMethod*)

set_numberOfLayers(*numberOfLayers*)

set_threshold(*threshold*)

set_width(*width*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NeuralNetwork'*, *mapping_=None*)

validate_ACTIVATION_FUNCTION(*value*)

validate_MINING_FUNCTION(*value*)

validate_NN_NORMALIZATION_METHOD(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**NeuralOutput**(*outputNeuron=None*, *Extension=None*, *DerivedField=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NeuralOutput'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NeuralOutput'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NeuralOutput'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NeuralOutput'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DerivedField()

get_Extension()

get_outputNeuron()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_DerivedField(*DerivedField*)

set_Extension(*Extension*)

set_outputNeuron(*outputNeuron*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NeuralOutput'*, *mapping_=None*)

validate_NN_NEURON_IDREF(*value*)

class PMML44Super.**NeuralOutputs**(*numberOfOutputs=None*, *Extension=None*, *NeuralOutput=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_NeuralOutput(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NeuralOutputs'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NeuralOutputs'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NeuralOutputs'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NeuralOutputs'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_NeuralOutput()

get_numberOfOutputs()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_NeuralOutput_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_NeuralOutput_at(*index*, *value*)

set_Extension(*Extension*)

set_NeuralOutput(*NeuralOutput*)

set_numberOfOutputs(*numberOfOutputs*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NeuralOutputs'*, *mapping_=None*)

class PMML44Super.**Neuron**(*id=None*, *bias=None*, *width=None*, *altitude=None*, *Extension=None*, *Con=None*)

Bases: PMML44Super.GeneratedsSuper

add_Con(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Neuron'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Neuron'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Neuron'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Neuron'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Con()

get_Extension()

get_altitude()

get_bias()

get_id()

get_width()

hasContent_()

insert_Con_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_Con_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Con(*Con*)

set_Extension(*Extension*)

set_altitude(*altitude*)

set_bias(*bias*)

set_id(*id*)

set_width(*width*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Neuron'*, *mapping_=None*)

validate_NN_NEURON_ID(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Node**(*id=None*, *score=None*, *recordCount=None*, *defaultChild=None*, *SimplePredicate=None*, *CompoundPredicate=None*, *SimpleSetPredicate=None*, *True_=None*, *False_=None*, *Partition=None*, *ScoreDistribution=None*, *Node_member=None*, *Extension=None*, *Regression=None*, *DecisionTree=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Node(*value*)

add_ScoreDistribution(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Node'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Node'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Node'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Node'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_CompoundPredicate()

get_DecisionTree()

get_Extension()

get_False()

get_Node()

get_Partition()

get_Regression()

get_ScoreDistribution()

get_SimplePredicate()

get_SimpleSetPredicate()

get_True()

get_defaultChild()

get_id()

get_recordCount()

get_score()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Node_at(*index*, *value*)

insert_ScoreDistribution_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Node_at(*index*, *value*)

replace_ScoreDistribution_at(*index*, *value*)

set_CompoundPredicate(*CompoundPredicate*)

set_DecisionTree(*DecisionTree*)

set_Extension(*Extension*)

set_False(*False_*)

set_Node(*Node*)

set_Partition(*Partition*)

set_Regression(*Regression*)

set_ScoreDistribution(*ScoreDistribution*)

set_SimplePredicate(*SimplePredicate*)

set_SimpleSetPredicate(*SimpleSetPredicate*)

set_True(*True_*)

set_defaultChild(*defaultChild*)

set_id(*id*)

set_recordCount(*recordCount*)

set_score(*score*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Node'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**NonseasonalComponent**(*p=None*, *d=None*, *q=None*, *Extension=None*, *AR=None*, *MA=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NonseasonalComponent'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NonseasonalComponent'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NonseasonalComponent'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NonseasonalComponent'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AR()

get_Extension()

get_MA()

get_d()

get_p()

get_q()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_AR(*AR*)

set_Extension(*Extension*)

set_MA(*MA*)

set_d(*d*)

set_p(*p*)

set_q(*q*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NonseasonalComponent'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**NonseasonalFactor**(*difference='0'*, *maximumOrder=None*, *Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NonseasonalFactor'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NonseasonalFactor'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NonseasonalFactor'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NonseasonalFactor'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

get_difference()

get_maximumOrder()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

set_difference(*difference*)

set_maximumOrder(*maximumOrder*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NonseasonalFactor'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**NormContinuous**(*mapMissingTo=None*, *field=None*, *outliers='asIs'*, *Extension=None*, *LinearNorm=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_LinearNorm(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NormContinuous'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NormContinuous'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NormContinuous'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NormContinuous'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LinearNorm()

get_field()

get_mapMissingTo()

get_outliers()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_LinearNorm_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_LinearNorm_at(*index*, *value*)

set_Extension(*Extension*)

set_LinearNorm(*LinearNorm*)

set_field(*field*)

set_mapMissingTo(*mapMissingTo*)

set_outliers(*outliers*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NormContinuous'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_NUMBER(*value*)

validate_OUTLIER_TREATMENT_METHOD(*value*)

class PMML44Super.**NormDiscrete**(*field=None*, *value=None*, *mapMissingTo=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NormDiscrete'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NormDiscrete'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NormDiscrete'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NormDiscrete'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_field()

get_mapMissingTo()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_field(*field*)

set_mapMissingTo(*mapMissingTo*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NormDiscrete'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_NUMBER(*value*)

class PMML44Super.**NormalDistributionForBN**(*Extension=None*, *Mean=None*, *Variance=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NormalDistributionForBN'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NormalDistributionForBN'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NormalDistributionForBN'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NormalDistributionForBN'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Mean()

get_Variance()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Mean(*Mean*)

set_Variance(*Variance*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NormalDistributionForBN'*, *mapping_=None*)

class PMML44Super.**Numerator**(*Extension=None*, *NonseasonalFactor=None*, *SeasonalFactor=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Numerator'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Numerator'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Numerator'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Numerator'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_NonseasonalFactor()

get_SeasonalFactor()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_NonseasonalFactor(*NonseasonalFactor*)

set_SeasonalFactor(*SeasonalFactor*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Numerator'*, *mapping_=None*)

class PMML44Super.**NumericInfo**(*minimum=None*, *maximum=None*, *mean=None*, *standardDeviation=None*, *median=None*, *interQuartileRange=None*, *Extension=None*, *Quantile=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Quantile(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NumericInfo'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NumericInfo'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NumericInfo'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NumericInfo'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Quantile()

get_interQuartileRange()

get_maximum()

get_mean()

get_median()

get_minimum()

get_standardDeviation()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Quantile_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Quantile_at(*index*, *value*)

set_Extension(*Extension*)

set_Quantile(*Quantile*)

set_interQuartileRange(*interQuartileRange*)

set_maximum(*maximum*)

set_mean(*mean*)

set_median(*median*)

set_minimum(*minimum*)

set_standardDeviation(*standardDeviation*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NumericInfo'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**NumericPredictor**(*name=None*, *exponent='1'*, *coefficient=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='NumericPredictor'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='NumericPredictor'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='NumericPredictor'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='NumericPredictor'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_coefficient()

get_exponent()

get_name()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_coefficient(*coefficient*)

set_exponent(*exponent*)

set_name(*name*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='NumericPredictor'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Optimizers**(*clipnorm=None*, *clipvalue=None*, *Extension=None*, *SGD=None*, *RMSprop=None*, *Adagrad=None*, *Adadelta=None*, *Adam=None*, *Adamax=None*, *Nadam=None*)

Bases: PMML44Super.GeneratedsSuper

add_Adadelta(*value*)

add_Adagrad(*value*)

add_Adam(*value*)

add_Adamax(*value*)

add_Extension(*value*)

add_Nadam(*value*)

add_RMSprop(*value*)

add_SGD(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Optimizers'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Optimizers'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Optimizers'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Optimizers'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Adadelta()

get_Adagrad()

get_Adam()

get_Adamax()

get_Extension()

get_Nadam()

get_RMSprop()

get_SGD()

get_clipnorm()

get_clipvalue()

hasContent_()

insert_Adadelta_at(*index*, *value*)

insert_Adagrad_at(*index*, *value*)

insert_Adam_at(*index*, *value*)

insert_Adamax_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_Nadam_at(*index*, *value*)

insert_RMSprop_at(*index*, *value*)

insert_SGD_at(*index*, *value*)

replace_Adadelta_at(*index*, *value*)

replace_Adagrad_at(*index*, *value*)

replace_Adam_at(*index*, *value*)

replace_Adamax_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Nadam_at(*index*, *value*)

replace_RMSprop_at(*index*, *value*)

replace_SGD_at(*index*, *value*)

set_Adadelta(*Adadelta*)

set_Adagrad(*Adagrad*)

set_Adam(*Adam*)

set_Adamax(*Adamax*)

set_Extension(*Extension*)

set_Nadam(*Nadam*)

set_RMSprop(*RMSprop*)

set_SGD(*SGD*)

set_clipnorm(*clipnorm*)

set_clipvalue(*clipvalue*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Optimizers'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**OptimumLiftGraph**(*Extension=None*, *LiftGraph=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='OptimumLiftGraph'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='OptimumLiftGraph'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='OptimumLiftGraph'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='OptimumLiftGraph'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LiftGraph()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_LiftGraph(*LiftGraph*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='OptimumLiftGraph'*, *mapping_=None*)

class PMML44Super.**OutlierEffect**(*type_=None*, *startTime=None*, *magnitude=None*, *dampingCoefficient=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='OutlierEffect'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='*', *name_='OutlierEffect'*)

exportChildren(*outfile*, *level*, *namespace_='*', *name_='OutlierEffect'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='OutlierEffect'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_dampingCoefficient()

get_magnitude()

get_startTime()

get_type()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_dampingCoefficient(*dampingCoefficient*)

set_magnitude(*magnitude*)

set_startTime(*startTime*)

set_type(*type_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='OutlierEffect'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Output**(*Extension=None*, *OutputField=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_OutputField(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_='*', *name_='Output'*, *namespacedef_='*', *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='*', *name_='Output'*)

exportChildren(*outfile*, *level*, *namespace_='*', *name_='Output'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Output'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_OutputField()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_OutputField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_OutputField_at(*index*, *value*)

set_Extension(*Extension*)

set_OutputField(*OutputField*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Output'*, *mapping_=None*)

class PMML44Super.**OutputField**(*name=None*, *displayName=None*, *optype=None*, *dataType=None*, *targetField=None*, *feature='predictedValue'*, *value=None*, *numTopCategories=None*, *threshold=None*, *ruleFeature='consequent'*, *algorithm='exclusiveRecommendation'*, *rank='1'*, *rankBasis='confidence'*, *rankOrder='descending'*, *isMultiValued='0'*, *segmentId=None*, *isFinalResult=True*, *Extension=None*, *Decisions=None*, *FieldRef=None*, *Constant=None*, *NormContinuous=None*, *NormDiscrete=None*, *Discretize=None*, *MapValues=None*, *TextIndex=None*, *Apply=None*, *Aggregate=None*, *Lag=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='OutputField'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='OutputField'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='OutputField'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='OutputField'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Aggregate()

get_Apply()

get_Constant()

get_Decisions()

get_Discretize()

get_Extension()

get_FieldRef()

get_Lag()

get_MapValues()

get_NormContinuous()

get_NormDiscrete()

get_TextIndex()

get_algorithm()

get_dataType()

get_displayName()

get_feature()

get_isFinalResult()

get_isMultiValued()

get_name()

get_numTopCategories()

get_optype()

get_rank()

get_rankBasis()

get_rankOrder()

get_ruleFeature()

get_segmentId()

get_targetField()

get_threshold()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Aggregate(*Aggregate*)

set_Apply(*Apply*)

set_Constant(*Constant*)

set_Decisions(*Decisions*)

set_Discretize(*Discretize*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_Lag(*Lag*)

set_MapValues(*MapValues*)

set_NormContinuous(*NormContinuous*)

set_NormDiscrete(*NormDiscrete*)

set_TextIndex(*TextIndex*)

set_algorithm(*algorithm*)

set_dataType(*dataType*)

set_displayName(*displayName*)

set_feature(*feature*)

set_isFinalResult(*isFinalResult*)

set_isMultiValued(*isMultiValued*)

set_name(*name*)

set_numTopCategories(*numTopCategories*)

set_optype(*optype*)

set_rank(*rank*)

set_rankBasis(*rankBasis*)

set_rankOrder(*rankOrder*)

set_ruleFeature(*ruleFeature*)

set_segmentId(*segmentId*)

set_targetField(*targetField*)

set_threshold(*threshold*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='OutputField'*, *mapping_=None*)

validate_DATATYPE(*value*)

validate_FIELD_NAME(*value*)

validate_INT_NUMBER(*value*)

validate_OPTYPE(*value*)

validate_REAL_NUMBER(*value*)

validate_RESULT_FEATURE(*value*)

validate_RULE_FEATURE(*value*)

class PMML44Super.**PCell**(*targetCategory=None*, *parameterName=None*, *beta=None*, *df=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PCell'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PCell'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PCell'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PCell'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_beta()

get_df()

get_parameterName()

get_targetCategory()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_beta(*beta*)

set_df(*df*)

set_parameterName(*parameterName*)

set_targetCategory(*targetCategory*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PCell'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**PCovCell**(*pRow=None*, *pCol=None*, *tRow=None*, *tCol=None*, *value=None*, *targetCategory=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PCovCell'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PCovCell'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PCovCell'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PCovCell'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_pCol()

get_pRow()

get_tCol()

get_tRow()

get_targetCategory()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_pCol(*pCol*)

set_pRow(*pRow*)

set_tCol(*tCol*)

set_tRow(*tRow*)

set_targetCategory(*targetCategory*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PCovCell'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**PCovMatrix**(*type_=None*, *Extension=None*, *PCovCell=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_PCovCell(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PCovMatrix'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PCovMatrix'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PCovMatrix'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PCovMatrix'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_PCovCell()

get_type()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_PCovCell_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_PCovCell_at(*index*, *value*)

set_Extension(*Extension*)

set_PCovCell(*PCovCell*)

set_type(*type_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PCovMatrix'*, *mapping_=None*)

class PMML44Super.**PMML**(*version=None*, *Header=None*, *MiningBuildTask=None*, *DataDictionary=None*, *TransformationDictionary=None*, *AssociationModel=None*, *BayesianNetworkModel=None*, *BaselineModel=None*, *ClusteringModel=None*, *DeepNetwork=None*, *AnomalyDetectionModel=None*, *GaussianProcessModel=None*, *GeneralRegressionModel=None*, *MiningModel=None*, *NaiveBayesModel=None*, *NearestNeighborModel=None*, *NeuralNetwork=None*, *RegressionModel=None*, *RuleSetModel=None*, *SequenceModel=None*, *Scorecard=None*, *SupportVectorMachineModel=None*, *TextModel=None*, *TimeSeriesModel=None*, *TreeModel=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_AnomalyDetectionModel(*value*)

add_AssociationModel(*value*)

add_BaselineModel(*value*)

add_BayesianNetworkModel(*value*)

add_ClusteringModel(*value*)

add_DeepNetwork(*value*)

add_Extension(*value*)

add_GaussianProcessModel(*value*)

add_GeneralRegressionModel(*value*)

add_MiningModel(*value*)

add_NaiveBayesModel(*value*)

add_NearestNeighborModel(*value*)

add_NeuralNetwork(*value*)

add_RegressionModel(*value*)

add_RuleSetModel(*value*)

add_Scorecard(*value*)

add_SequenceModel(*value*)

add_SupportVectorMachineModel(*value*)

add_TextModel(*value*)

add_TimeSeriesModel(*value*)

add_TreeModel(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PMML'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PMML'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PMML'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PMML'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AnomalyDetectionModel()

get_AssociationModel()

get_BaselineModel()

get_BayesianNetworkModel()

get_ClusteringModel()

get_DataDictionary()

get_DeepNetwork()

get_Extension()

get_GaussianProcessModel()

get_GeneralRegressionModel()

get_Header()

get_MiningBuildTask()

get_MiningModel()

get_NaiveBayesModel()

get_NearestNeighborModel()

get_NeuralNetwork()

get_RegressionModel()

get_RuleSetModel()

get_Scorecard()

get_SequenceModel()

get_SupportVectorMachineModel()

get_TextModel()

get_TimeSeriesModel()

get_TransformationDictionary()

get_TreeModel()

get_version()

hasContent_()

insert_AnomalyDetectionModel_at(*index*, *value*)

insert_AssociationModel_at(*index*, *value*)

insert_BaselineModel_at(*index*, *value*)

insert_BayesianNetworkModel_at(*index*, *value*)

insert_ClusteringModel_at(*index*, *value*)

insert_DeepNetwork_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_GaussianProcessModel_at(*index*, *value*)

insert_GeneralRegressionModel_at(*index*, *value*)

insert_MiningModel_at(*index*, *value*)

insert_NaiveBayesModel_at(*index*, *value*)

insert_NearestNeighborModel_at(*index*, *value*)

insert_NeuralNetwork_at(*index*, *value*)

insert_RegressionModel_at(*index*, *value*)

insert_RuleSetModel_at(*index*, *value*)

insert_Scorecard_at(*index*, *value*)

insert_SequenceModel_at(*index*, *value*)

insert_SupportVectorMachineModel_at(*index*, *value*)

insert_TextModel_at(*index*, *value*)

insert_TimeSeriesModel_at(*index*, *value*)

insert_TreeModel_at(*index*, *value*)

replace_AnomalyDetectionModel_at(*index*, *value*)

replace_AssociationModel_at(*index*, *value*)

replace_BaselineModel_at(*index*, *value*)

replace_BayesianNetworkModel_at(*index*, *value*)

replace_ClusteringModel_at(*index*, *value*)

replace_DeepNetwork_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_GaussianProcessModel_at(*index*, *value*)

replace_GeneralRegressionModel_at(*index*, *value*)

replace_MiningModel_at(*index*, *value*)

replace_NaiveBayesModel_at(*index*, *value*)

replace_NearestNeighborModel_at(*index*, *value*)

replace_NeuralNetwork_at(*index*, *value*)

replace_RegressionModel_at(*index*, *value*)

replace_RuleSetModel_at(*index*, *value*)

replace_Scorecard_at(*index*, *value*)

replace_SequenceModel_at(*index*, *value*)

replace_SupportVectorMachineModel_at(*index*, *value*)

replace_TextModel_at(*index*, *value*)

replace_TimeSeriesModel_at(*index*, *value*)

replace_TreeModel_at(*index*, *value*)

set_AnomalyDetectionModel(*AnomalyDetectionModel*)

set_AssociationModel(*AssociationModel*)

set_BaselineModel(*BaselineModel*)

set_BayesianNetworkModel(*BayesianNetworkModel*)

set_ClusteringModel(*ClusteringModel*)

set_DataDictionary(*DataDictionary*)

set_DeepNetwork(*DeepNetwork*)

set_Extension(*Extension*)

set_GaussianProcessModel(*GaussianProcessModel*)

set_GeneralRegressionModel(*GeneralRegressionModel*)

set_Header(*Header*)

set_MiningBuildTask(*MiningBuildTask*)

set_MiningModel(*MiningModel*)

set_NaiveBayesModel(*NaiveBayesModel*)

set_NearestNeighborModel(*NearestNeighborModel*)

set_NeuralNetwork(*NeuralNetwork*)

set_RegressionModel(*RegressionModel*)

set_RuleSetModel(*RuleSetModel*)

set_Scorecard(*Scorecard*)

set_SequenceModel(*SequenceModel*)

set_SupportVectorMachineModel(*SupportVectorMachineModel*)

set_TextModel(*TextModel*)

set_TimeSeriesModel(*TimeSeriesModel*)

set_TransformationDictionary(*TransformationDictionary*)

set_TreeModel(*TreeModel*)

set_version(*version*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PMML'*, *mapping_=None*)

class PMML44Super.**PPCell**(*value=None*, *predictorName=None*, *parameterName=None*, *targetCategory=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PPCell'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PPCell'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PPCell'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PPCell'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_parameterName()

get_predictorName()

get_targetCategory()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_parameterName(*parameterName*)

set_predictorName(*predictorName*)

set_targetCategory(*targetCategory*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PPCell'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**PPMatrix**(*Extension=None*, *PPCell=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_PPCell(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PPMatrix'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PPMatrix'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PPMatrix'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PPMatrix'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_PPCell()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_PPCell_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_PPCell_at(*index*, *value*)

set_Extension(*Extension*)

set_PPCell(*PPCell*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PPMatrix'*, *mapping_=None*)

class PMML44Super.**PairCounts**(*value=None*, *Extension=None*, *TargetValueCounts=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PairCounts'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PairCounts'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PairCounts'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PairCounts'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_TargetValueCounts()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_TargetValueCounts(*TargetValueCounts*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PairCounts'*, *mapping_=None*)

class PMML44Super.**ParamMatrix**(*Extension=None*, *PCell=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_PCell(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ParamMatrix'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ParamMatrix'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ParamMatrix'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ParamMatrix'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

get_PCell()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_PCell_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_PCell_at(*index*, *value*)

set_Extension(*Extension*)

set_PCell(*PCell*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ParamMatrix'*, *mapping_=None*)

class PMML44Super.**Parameter**(*name=None*, *value=None*, *label=None*, *referencePoint='0'*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Parameter'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Parameter'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Parameter'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Parameter'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

get_label()

get_name()

get_referencePoint()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_label(*label*)

set_name(*name*)

set_referencePoint(*referencePoint*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Parameter'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ParameterField**(*name=None*, *optype=None*, *dataType=None*, *displayName=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ParameterField'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ParameterField'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ParameterField'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ParameterField'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_dataType()

get_displayName()

get_name()

get_optype()

hasContent_()

set_dataType(*dataType*)

set_displayName(*displayName*)

set_name(*name*)

set_optype(*optype*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ParameterField'*, *mapping_=None*)

validate_DATATYPE(*value*)

validate_FIELD_NAME(*value*)

validate_OPTYPE(*value*)

class PMML44Super.**ParameterList**(*Extension=None*, *Parameter=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Parameter(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ParameterList'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ParameterList'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ParameterList'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ParameterList'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Parameter()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Parameter_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Parameter_at(*index*, *value*)

set_Extension(*Extension*)

set_Parameter(*Parameter*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ParameterList'*, *mapping_=None*)

class PMML44Super.**ParentValue**(*parent=None*, *value=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ParentValue'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ParentValue'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ParentValue'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ParentValue'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_parent()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_parent(*parent*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ParentValue'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**Partition**(*name=None*, *size=None*, *Extension=None*, *PartitionFieldStats=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_PartitionFieldStats(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Partition'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Partition'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Partition'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Partition'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_PartitionFieldStats()

get_name()

get_size()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_PartitionFieldStats_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_PartitionFieldStats_at(*index*, *value*)

set_Extension(*Extension*)

set_PartitionFieldStats(*PartitionFieldStats*)

set_name(*name*)

set_size(*size*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Partition'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**PartitionFieldStats**(*field=None*, *weighted='0'*, *Extension=None*, *Counts=None*, *NumericInfo=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Array(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PartitionFieldStats'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PartitionFieldStats'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PartitionFieldStats'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PartitionFieldStats'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Counts()

get_Extension()

get_NumericInfo()

get_field()

get_weighted()

hasContent_()

insert_Array_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_Array_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Counts(*Counts*)

set_Extension(*Extension*)

set_NumericInfo(*NumericInfo*)

set_field(*field*)

set_weighted(*weighted*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PartitionFieldStats'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**PastVariances**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PastVariances'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PastVariances'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PastVariances'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PastVariances'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PastVariances'*, *mapping_=None*)

class PMML44Super.**PoissonDistribution**(*mean=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PoissonDistribution'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PoissonDistribution'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PoissonDistribution'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PoissonDistribution'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_mean()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_mean(*mean*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PoissonDistribution'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**PolynomialKernelType**(*description=None*, *gamma='1'*, *coef0='1'*, *degree='1'*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PolynomialKernelType'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PolynomialKernelType'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PolynomialKernelType'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PolynomialKernelType'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_coef0()

get_degree()

get_description()

get_gamma()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_coef0(*coef0*)

set_degree(*degree*)

set_description(*description*)

set_gamma(*gamma*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PolynomialKernelType'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**PredictiveModelQuality**(*targetField=None*, *dataName=None*, *dataUsage='training'*, *meanError=None*, *meanAbsoluteError=None*, *meanSquaredError=None*, *rootMeanSquaredError=None*, *r_squared=None*, *adj_r_squared=None*, *sumSquaredError=None*, *sumSquaredRegression=None*, *numOfRecords=None*, *numOfRecordsWeighted=None*, *numOfPredictors=None*, *degreesOfFreedom=None*, *fStatistic=None*, *AIC=None*, *BIC=None*, *AICc=None*, *Extension=None*, *ConfusionMatrix=None*, *LiftData=None*, *ROC=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_LiftData(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PredictiveModelQuality'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='*', *name_='PredictiveModelQuality'*)

exportChildren(*outfile*, *level*, *namespace_='*', *name_='PredictiveModelQuality'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PredictiveModelQuality'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AIC()

get_AICc()

get_BIC()

get_ConfusionMatrix()

get_Extension()

get_LiftData()

get_ROC()

get_adj_r_squared()

get_dataName()

get_dataUsage()

get_degreesOfFreedom()

get_fStatistic()

get_meanAbsoluteError()

get_meanError()

get_meanSquaredError()

get_numOfPredictors()

get_numOfRecords()

get_numOfRecordsWeighted()

get_r_squared()

get_rootMeanSquaredError()

get_sumSquaredError()

get_sumSquaredRegression()

get_targetField()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_LiftData_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_LiftData_at(*index*, *value*)

set_AIC(*AIC*)

set_AICc(*AICc*)

set_BIC(*BIC*)

set_ConfusionMatrix(*ConfusionMatrix*)

set_Extension(*Extension*)

set_LiftData(*LiftData*)

set_ROC(*ROC*)

set_adj_r_squared(*adj_r_squared*)

set_dataName(*dataName*)

set_dataUsage(*dataUsage*)

set_degreesOfFreedom(*degreesOfFreedom*)

set_fStatistic(*fStatistic*)

set_meanAbsoluteError(*meanAbsoluteError*)

set_meanError(*meanError*)

set_meanSquaredError(*meanSquaredError*)

set_numOfPredictors(*numOfPredictors*)

set_numOfRecords(*numOfRecords*)

set_numOfRecordsWeighted(*numOfRecordsWeighted*)

set_r_squared(*r_squared*)

set_rootMeanSquaredError(*rootMeanSquaredError*)

set_sumSquaredError(*sumSquaredError*)

set_sumSquaredRegression(*sumSquaredRegression*)

set_targetField(*targetField*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PredictiveModelQuality'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_NUMBER(*value*)

class PMML44Super.**Predictor**(*name=None*, *contrastMatrixType=None*, *Extension=None*, *Categories=None*, *Matrix=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Predictor'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Predictor'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Predictor'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Predictor'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Categories()

get_Extension()

get_Matrix()

get_contrastMatrixType()

get_name()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Categories(*Categories*)

set_Extension(*Extension*)

set_Matrix(*Matrix*)

set_contrastMatrixType(*contrastMatrixType*)

set_name(*name*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Predictor'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**PredictorTerm**(*name=None*, *coefficient=None*, *Extension=None*, *FieldRef=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_FieldRef(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PredictorTerm'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PredictorTerm'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PredictorTerm'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PredictorTerm'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

get_FieldRef()

get_coefficient()

get_name()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_FieldRef_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_FieldRef_at(*index*, *value*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_coefficient(*coefficient*)

set_name(*name*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PredictorTerm'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**PsiVector**(*targetField=None*, *variance=None*, *Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='PsiVector'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='PsiVector'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='PsiVector'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='PsiVector'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Array()

get_Extension()

get_targetField()

get_variance()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

set_targetField(*targetField*)

set_variance(*variance*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='PsiVector'*, *mapping_=None*)

class PMML44Super.**Quantile**(*quantileLimit=None*, *quantileValue=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Quantile'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Quantile'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Quantile'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Quantile'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_quantileLimit()

get_quantileValue()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_quantileLimit(*quantileLimit*)

set_quantileValue(*quantileValue*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Quantile'*, *mapping_=None*)

validate_NUMBER(*value*)

validate_PERCENTAGE_NUMBER(*value*)

class PMML44Super.**REAL_Entries**

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='REAL-Entries'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='REAL-Entries'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='REAL-Entries'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='REAL-Entries'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

hasContent_()

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='REAL-Entries'*, *mapping_=None*)

class PMML44Super.**REAL_SparseArray**(*n=None*, *defaultValue='0'*, *Indices=None*, *REAL_Entries=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='REAL-SparseArray'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='REAL-SparseArray'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='REAL-SparseArray'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='REAL-SparseArray'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Indices()

get_REAL_Entries()

get_defaultValue()

get_n()

hasContent_()

set_Indices(*Indices*)

set_REAL_Entries(*REAL_Entries*)

set_defaultValue(*defaultValue*)

set_n(*n*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='REAL-SparseArray'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_Indices(*value*)

validate_REAL_Entries(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**RMSprop**(*learningRate=None*, *rho=None*, *decayRate=None*, *epsilon=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='RMSprop'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='RMSprop'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='RMSprop'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='RMSprop'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_decayRate()

get_epsilon()

get_learningRate()

get_rho()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_decayRate(*decayRate*)

set_epsilon(*epsilon*)

set_learningRate(*learningRate*)

set_rho(*rho*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='RMSprop'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ROC**(*positiveTargetFieldValue=None*, *positiveTargetFieldDisplayValue=None*, *negativeTargetFieldValue=None*, *negativeTargetFieldDisplayValue=None*, *Extension=None*, *ROCGraph=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ROC'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ROC'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ROC'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ROC'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_ROCGraph()

get_negativeTargetFieldDisplayValue()

get_negativeTargetFieldValue()

get_positiveTargetFieldDisplayValue()

get_positiveTargetFieldValue()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_ROCGraph(*ROCGraph*)

set_negativeTargetFieldDisplayValue(*negativeTargetFieldDisplayValue*)

set_negativeTargetFieldValue(*negativeTargetFieldValue*)

set_positiveTargetFieldDisplayValue(*positiveTargetFieldDisplayValue*)

set_positiveTargetFieldValue(*positiveTargetFieldValue*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ROC'*, *mapping_=None*)

class PMML44Super.**ROCGraph**(*Extension=None*, *XCoordinates=None*, *YCoordinates=None*, *BoundaryValues=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ROCGraph'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ROCGraph'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ROCGraph'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ROCGraph'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_BoundaryValues()

get_Extension()

get_XCoordinates()

get_YCoordinates()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_BoundaryValues(*BoundaryValues*)

set_Extension(*Extension*)

set_XCoordinates(*XCoordinates*)

set_YCoordinates(*YCoordinates*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ROCGraph'*, *mapping_=None*)

class PMML44Super.**RadialBasisKernel**(*description=None*, *gamma='1'*, *noiseVariance='1'*, *lambda_='1'*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='RadialBasisKernel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='RadialBasisKernel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='RadialBasisKernel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='RadialBasisKernel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_description()

get_gamma()

get_lambda()

get_noiseVariance()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_description(*description*)

set_gamma(*gamma*)

set_lambda(*lambda_*)

set_noiseVariance(*noiseVariance*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='RadialBasisKernel'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**RadialBasisKernelType**(*description=None*, *gamma='1'*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='RadialBasisKernelType'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='RadialBasisKernelType'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='RadialBasisKernelType'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='RadialBasisKernelType'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_description()

get_gamma()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_description(*description*)

set_gamma(*gamma*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='RadialBasisKernelType'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**RandomLiftGraph**(*Extension=None*, *LiftGraph=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='RandomLiftGraph'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='RandomLiftGraph'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='RandomLiftGraph'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='RandomLiftGraph'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LiftGraph()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_LiftGraph(*LiftGraph*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='RandomLiftGraph'*, *mapping_=None*)

class PMML44Super.**Regression**(*modelName=None*, *functionName=None*, *algorithmName=None*, *normalizationMethod='none'*, *Extension=None*, *Output=None*, *ModelStats=None*, *Targets=None*, *LocalTransformations=None*, *ResultField=None*, *RegressionTable=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_RegressionTable(*value*)

add_ResultField(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Regression'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Regression'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Regression'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Regression'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LocalTransformations()

get_ModelStats()

get_Output()

get_RegressionTable()

get_ResultField()

get_Targets()

get_algorithmName()

get_functionName()

get_modelName()

get_normalizationMethod()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_RegressionTable_at(*index*, *value*)

insert_ResultField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_RegressionTable_at(*index*, *value*)

replace_ResultField_at(*index*, *value*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_ModelStats(*ModelStats*)

set_Output(*Output*)

set_RegressionTable(*RegressionTable*)

set_ResultField(*ResultField*)

set_Targets(*Targets*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_modelName(*modelName*)

set_normalizationMethod(*normalizationMethod*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Regression'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

validate_REGRESSIONNORMALIZATIONMETHOD(*value*)

class PMML44Super.**RegressionModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *modelType=None*, *targetFieldName=None*, *normalizationMethod='none'*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *RegressionTable=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_RegressionTable(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='RegressionModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='RegressionModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='RegressionModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='RegressionModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_RegressionTable()

get_Targets()

get_algorithmName()

get_functionName()

get_isScorable()

get_modelName()

get_modelType()

get_normalizationMethod()

get_targetFieldName()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_RegressionTable_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_RegressionTable_at(*index*, *value*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_RegressionTable(*RegressionTable*)

set_Targets(*Targets*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

set_modelType(*modelType*)

set_normalizationMethod(*normalizationMethod*)

set_targetFieldName(*targetFieldName*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='RegressionModel'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_MINING_FUNCTION(*value*)

validate_REGRESSIONNORMALIZATIONMETHOD(*value*)

class PMML44Super.**RegressionTable**(*intercept=None*, *targetCategory=None*, *Extension=None*, *NumericPredictor=None*, *CategoricalPredictor=None*, *PredictorTerm=None*)

Bases: PMML44Super.GeneratedsSuper

add_CategoricalPredictor(*value*)

add_Extension(*value*)

add_NumericPredictor(*value*)

add_PredictorTerm(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='RegressionTable'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='RegressionTable'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='RegressionTable'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='RegressionTable'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_CategoricalPredictor()

get_Extension()

get_NumericPredictor()

get_PredictorTerm()

get_intercept()

get_targetCategory()

hasContent_()

insert_CategoricalPredictor_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_NumericPredictor_at(*index*, *value*)

insert_PredictorTerm_at(*index*, *value*)

replace_CategoricalPredictor_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_NumericPredictor_at(*index*, *value*)

replace_PredictorTerm_at(*index*, *value*)

set_CategoricalPredictor(*CategoricalPredictor*)

set_Extension(*Extension*)

set_NumericPredictor(*NumericPredictor*)

set_PredictorTerm(*PredictorTerm*)

set_intercept(*intercept*)

set_targetCategory(*targetCategory*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='RegressionTable'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**RegressorValues**(*Extension=None*, *TimeSeries=None*, *TrendCoefficients=None*, *TransferFunctionValues=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='RegressorValues'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='RegressorValues'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='RegressorValues'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='RegressorValues'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_TimeSeries()

get_TransferFunctionValues()

get_TrendCoefficients()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_TimeSeries(*TimeSeries*)

set_TransferFunctionValues(*TransferFunctionValues*)

set_TrendCoefficients(*TrendCoefficients*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='RegressorValues'*, *mapping_=None*)

class PMML44Super.**ResidualSquareCoefficients**(*Extension=None*, *Residuals=None*, *MACoefficients=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ResidualSquareCoefficients'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ResidualSquareCoefficients'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ResidualSquareCoefficients'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ResidualSquareCoefficients'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_MACoefficients()

get_Residuals()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_MACoefficients(*MACoefficients*)

set_Residuals(*Residuals*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ResidualSquareCoefficients'*, *mapping_=None*)

class PMML44Super.**Residuals**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Residuals'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Residuals'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Residuals'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Residuals'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Residuals'*, *mapping_=None*)

class PMML44Super.**ResultField**(*name=None*, *displayName=None*, *optype=None*, *dataType=None*, *feature=None*, *value=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ResultField'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ResultField'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ResultField'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ResultField'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_dataType()

get_displayName()

get_feature()

get_name()

get_optype()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_dataType(*dataType*)

set_displayName(*displayName*)

set_feature(*feature*)

set_name(*name*)

set_optype(*optype*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ResultField'*, *mapping_=None*)

validate_DATATYPE(*value*)

validate_FIELD_NAME(*value*)

validate_OPTYPE(*value*)

validate_RESULT_FEATURE(*value*)

class PMML44Super.**RuleSelectionMethod**(*criterion=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='RuleSelectionMethod'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='RuleSelectionMethod'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='RuleSelectionMethod'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='RuleSelectionMethod'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

get_criterion()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_criterion(*criterion*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='RuleSelectionMethod'*, *mapping_=None*)

class PMML44Super.**RuleSet**(*recordCount=None*, *nbCorrect=None*, *defaultScore=None*, *defaultConfidence=None*, *Extension=None*, *RuleSelectionMethod=None*, *ScoreDistribution=None*, *SimpleRule=None*, *CompoundRule=None*)

Bases: PMML44Super.GeneratedsSuper

add_CompoundRule(*value*)

add_Extension(*value*)

add_RuleSelectionMethod(*value*)

add_ScoreDistribution(*value*)

add_SimpleRule(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='RuleSet'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='RuleSet'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='RuleSet'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='RuleSet'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_CompoundRule()

get_Extension()

get_RuleSelectionMethod()

get_ScoreDistribution()

get_SimpleRule()

get_defaultConfidence()

get_defaultScore()

get_nbCorrect()

get_recordCount()

hasContent_()

insert_CompoundRule_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_RuleSelectionMethod_at(*index*, *value*)

insert_ScoreDistribution_at(*index*, *value*)

insert_SimpleRule_at(*index*, *value*)

replace_CompoundRule_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_RuleSelectionMethod_at(*index*, *value*)

replace_ScoreDistribution_at(*index*, *value*)

replace_SimpleRule_at(*index*, *value*)

set_CompoundRule(*CompoundRule*)

set_Extension(*Extension*)

set_RuleSelectionMethod(*RuleSelectionMethod*)

set_ScoreDistribution(*ScoreDistribution*)

set_SimpleRule(*SimpleRule*)

set_defaultConfidence(*defaultConfidence*)

set_defaultScore(*defaultScore*)

set_nbCorrect(*nbCorrect*)

set_recordCount(*recordCount*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='RuleSet'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**RuleSetModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *RuleSet=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='RuleSetModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='RuleSetModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='RuleSetModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='RuleSetModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_RuleSet()

get_Targets()

get_algorithmName()

get_functionName()

get_isScorable()

get_modelName()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_RuleSet(*RuleSet*)

set_Targets(*Targets*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='RuleSetModel'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

class PMML44Super.**SGD**(*learningRate=None*, *momentum=None*, *decayRate=None*, *nesterov=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SGD'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SGD'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SGD'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SGD'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_decayRate()

get_learningRate()

get_momentum()

get_nesterov()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_decayRate(*decayRate*)

set_learningRate(*learningRate*)

set_momentum(*momentum*)

set_nesterov(*nesterov*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SGD'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**ScoreDistribution**(*value=None*, *recordCount=None*, *confidence=None*, *probability=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ScoreDistribution'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ScoreDistribution'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ScoreDistribution'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ScoreDistribution'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_confidence()

get_probability()

get_recordCount()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_confidence(*confidence*)

set_probability(*probability*)

set_recordCount(*recordCount*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ScoreDistribution'*, *mapping_=None*)

validate_NUMBER(*value*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**Scorecard**(*modelName=None*, *functionName=None*, *algorithmName=None*, *initialScore='0'*, *useReasonCodes=True*, *reasonCodeAlgorithm='pointsBelow'*, *baselineScore=None*, *baselineMethod='other'*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *Characteristics=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Scorecard'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Scorecard'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Scorecard'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Scorecard'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Characteristics()

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_Targets()

get_algorithmName()

get_baselineMethod()

get_baselineScore()

get_functionName()

get_initialScore()

get_isScorable()

get_modelName()

get_reasonCodeAlgorithm()

get_useReasonCodes()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Characteristics(*Characteristics*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_Targets(*Targets*)

set_algorithmName(*algorithmName*)

set_baselineMethod(*baselineMethod*)

set_baselineScore(*baselineScore*)

set_functionName(*functionName*)

set_initialScore(*initialScore*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

set_reasonCodeAlgorithm(*reasonCodeAlgorithm*)

set_useReasonCodes(*useReasonCodes*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Scorecard'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

validate_NUMBER(*value*)

class PMML44Super.**SeasonalComponent**(*P=None*, *D=None*, *Q=None*, *period=None*, *Extension=None*, *AR=None*, *MA=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SeasonalComponent'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SeasonalComponent'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SeasonalComponent'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SeasonalComponent'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AR()

get_D()

get_Extension()

get_MA()

get_P()

get_Q()

get_period()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_AR(*AR*)

set_D(*D*)

set_Extension(*Extension*)

set_MA(*MA*)

set_P(*P*)

set_Q(*Q*)

set_period(*period*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SeasonalComponent'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**SeasonalFactor**(*difference='0'*, *maximumOrder=None*, *Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SeasonalFactor'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SeasonalFactor'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SeasonalFactor'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SeasonalFactor'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

get_difference()

get_maximumOrder()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

set_difference(*difference*)

set_maximumOrder(*maximumOrder*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SeasonalFactor'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**SeasonalTrendDecomposition**

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SeasonalTrendDecomposition'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SeasonalTrendDecomposition'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SeasonalTrendDecomposition'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SeasonalTrendDecomposition'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

hasContent_()

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SeasonalTrendDecomposition'*, *mapping_=None*)

class PMML44Super.**Seasonality_ExpoSmooth**(*type_=None*, *period=None*, *unit=None*, *phase=None*, *delta=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Seasonality_ExpoSmooth'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Seasonality_ExpoSmooth'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Seasonality_ExpoSmooth'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Seasonality_ExpoSmooth'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_delta()

get_period()

get_phase()

get_type()

get_unit()

hasContent_()

set_Array(*Array*)

set_delta(*delta*)

set_period(*period*)

set_phase(*phase*)

set_type(*type_*)

set_unit(*unit*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Seasonality_ExpoSmooth'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Segment**(*id=None*, *weight='1'*, *Extension=None*, *SimplePredicate=None*, *CompoundPredicate=None*, *SimpleSetPredicate=None*, *True_=None*, *False_=None*, *AssociationModel=None*, *BayesianNetworkModel=None*, *BaselineModel=None*, *ClusteringModel=None*, *DeepNetwork=None*, *AnomalyDetectionModel=None*, *GaussianProcessModel=None*, *GeneralRegressionModel=None*, *MiningModel=None*, *NaiveBayesModel=None*, *NearestNeighborModel=None*, *NeuralNetwork=None*, *RegressionModel=None*, *RuleSetModel=None*, *SequenceModel=None*, *Scorecard=None*, *SupportVectorMachineModel=None*, *TextModel=None*, *TimeSeriesModel=None*, *TreeModel=None*, *VariableWeight=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Segment'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Segment'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Segment'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Segment'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AnomalyDetectionModel()

get_AssociationModel()

get_BaselineModel()

get_BayesianNetworkModel()

get_ClusteringModel()

get_CompoundPredicate()

get_DeepNetwork()

get_Extension()

get_False()

get_GaussianProcessModel()

get_GeneralRegressionModel()

get_MiningModel()

get_NaiveBayesModel()

get_NearestNeighborModel()

get_NeuralNetwork()

get_RegressionModel()

get_RuleSetModel()

get_Scorecard()

get_SequenceModel()

get_SimplePredicate()

get_SimpleSetPredicate()

get_SupportVectorMachineModel()

get_TextModel()

get_TimeSeriesModel()

get_TreeModel()

get_True()

get_VariableWeight()

get_id()

get_weight()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_AnomalyDetectionModel(*AnomalyDetectionModel*)

set_AssociationModel(*AssociationModel*)

set_BaselineModel(*BaselineModel*)

set_BayesianNetworkModel(*BayesianNetworkModel*)

set_ClusteringModel(*ClusteringModel*)

set_CompoundPredicate(*CompoundPredicate*)

set_DeepNetwork(*DeepNetwork*)

set_Extension(*Extension*)

set_False(*False_*)

set_GaussianProcessModel(*GaussianProcessModel*)

set_GeneralRegressionModel(*GeneralRegressionModel*)

set_MiningModel(*MiningModel*)

set_NaiveBayesModel(*NaiveBayesModel*)

set_NearestNeighborModel(*NearestNeighborModel*)

set_NeuralNetwork(*NeuralNetwork*)

set_RegressionModel(*RegressionModel*)

set_RuleSetModel(*RuleSetModel*)

set_Scorecard(*Scorecard*)

set_SequenceModel(*SequenceModel*)

set_SimplePredicate(*SimplePredicate*)

set_SimpleSetPredicate(*SimpleSetPredicate*)

set_SupportVectorMachineModel(*SupportVectorMachineModel*)

set_TextModel(*TextModel*)

set_TimeSeriesModel(*TimeSeriesModel*)

set_TreeModel(*TreeModel*)

set_True(*True_*)

set_VariableWeight(*VariableWeight*)

set_id(*id*)

set_weight(*weight*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Segment'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**Segmentation**(*multipleModelMethod=None*, *missingThreshold='1'*, *Extension=None*, *Segment=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Segment(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Segmentation'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Segmentation'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Segmentation'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Segmentation'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Segment()

get_missingThreshold()

get_multipleModelMethod()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Segment_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Segment_at(*index*, *value*)

set_Extension(*Extension*)

set_Segment(*Segment*)

set_missingThreshold(*missingThreshold*)

set_multipleModelMethod(*multipleModelMethod*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Segmentation'*, *mapping_=None*)

validate_MULTIPLE_MODEL_METHOD(*value*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**Sequence**(*id=None*, *numberOfSets=None*, *occurrence=None*, *support=None*, *Extension=None*, *Delimiter=None*, *SetReference=None*, *Time=None*)

Bases: PMML44Super.GeneratedsSuper

add_Delimiter(*value*)

add_Extension(*value*)

add_SetReference(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Sequence'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Sequence'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Sequence'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Sequence'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Delimiter()

get_Extension()

get_SetReference()

get_Time()

get_id()

get_numberOfSets()

get_occurrence()

get_support()

hasContent_()

insert_Delimiter_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_SetReference_at(*index*, *value*)

replace_Delimiter_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_SetReference_at(*index*, *value*)

set_Delimiter(*Delimiter*)

set_Extension(*Extension*)

set_SetReference(*SetReference*)

set_Time(*Time*)

set_id(*id*)

set_numberOfSets(*numberOfSets*)

set_occurrence(*occurrence*)

set_support(*support*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Sequence'*, *mapping_=None*)

validate_ELEMENT_ID(*value*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**SequenceModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *numberOfTransactions=None*, *maxNumberOfItemsPerTransaction=None*, *avgNumberOfItemsPerTransaction=None*, *numberOfTransactionGroups=None*, *maxNumberOfTAsPerTAGroup=None*, *avgNumberOfTAsPerTAGroup=None*, *isScorable=True*, *MiningSchema=None*, *ModelStats=None*, *LocalTransformations=None*, *Constraints=None*, *Item=None*, *Itemset=None*, *SetPredicate=None*, *Sequence=None*, *SequenceRule=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Item(*value*)

add_Itemset(*value*)

add_Sequence(*value*)

add_SequenceRule(*value*)

add_SetPredicate(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SequenceModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SequenceModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SequenceModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SequenceModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Constraints()

get_Extension()

get_Item()

get_Itemset()

get_LocalTransformations()

get_MiningSchema()

get_ModelStats()

get_Sequence()

get_SequenceRule()

get_SetPredicate()

get_algorithmName()

get_avgNumberOfItemsPerTransaction()

get_avgNumberOfTAsPerTAGroup()

get_functionName()

get_isScorable()

get_maxNumberOfItemsPerTransaction()

get_maxNumberOfTAsPerTAGroup()

get_modelName()

get_numberOfTransactionGroups()

get_numberOfTransactions()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Item_at(*index*, *value*)

insert_Itemset_at(*index*, *value*)

insert_SequenceRule_at(*index*, *value*)

insert_Sequence_at(*index*, *value*)

insert_SetPredicate_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Item_at(*index*, *value*)

replace_Itemset_at(*index*, *value*)

replace_SequenceRule_at(*index*, *value*)

replace_Sequence_at(*index*, *value*)

replace_SetPredicate_at(*index*, *value*)

set_Constraints(*Constraints*)

set_Extension(*Extension*)

set_Item(*Item*)

set_Itemset(*Itemset*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelStats(*ModelStats*)

set_Sequence(*Sequence*)

set_SequenceRule(*SequenceRule*)

set_SetPredicate(*SetPredicate*)

set_algorithmName(*algorithmName*)

set_avgNumberOfItemsPerTransaction(*avgNumberOfItemsPerTransaction*)

set_avgNumberOfTAsPerTAGroup(*avgNumberOfTAsPerTAGroup*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_maxNumberOfItemsPerTransaction(*maxNumberOfItemsPerTransaction*)

set_maxNumberOfTAsPerTAGroup(*maxNumberOfTAsPerTAGroup*)

set_modelName(*modelName*)

set_numberOfTransactionGroups(*numberOfTransactionGroups*)

set_numberOfTransactions(*numberOfTransactions*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SequenceModel'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_MINING_FUNCTION(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**SequenceReference**(*seqId=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SequenceReference'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SequenceReference'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SequenceReference'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SequenceReference'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_seqId()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_seqId(*seqId*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SequenceReference'*, *mapping_=None*)

validate_ELEMENT_ID(*value*)

class PMML44Super.**SequenceRule**(*id=None*, *numberOfSets=None*, *occurrence=None*, *support=None*, *confidence=None*, *lift=None*, *Extension=None*, *AntecedentSequence=None*, *Delimiter=None*, *ConsequentSequence=None*, *Time=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SequenceRule'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SequenceRule'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SequenceRule'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SequenceRule'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AntecedentSequence()

get_ConsequentSequence()

get_Delimiter()

get_Extension()

get_Time()

get_confidence()

get_id()

get_lift()

get_numberOfSets()

get_occurrence()

get_support()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_AntecedentSequence(*AntecedentSequence*)

set_ConsequentSequence(*ConsequentSequence*)

set_Delimiter(*Delimiter*)

set_Extension(*Extension*)

set_Time(*Time*)

set_confidence(*confidence*)

set_id(*id*)

set_lift(*lift*)

set_numberOfSets(*numberOfSets*)

set_occurrence(*occurrence*)

set_support(*support*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SequenceRule'*, *mapping_=None*)

validate_ELEMENT_ID(*value*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**SetPredicate**(*id=None*, *field=None*, *operator=None*, *Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SetPredicate'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SetPredicate'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SetPredicate'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SetPredicate'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

get_field()

get_id()

get_operator()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

set_field(*field*)

set_id(*id*)

set_operator(*operator*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SetPredicate'*, *mapping_=None*)

validate_ELEMENT_ID(*value*)

validate_FIELD_NAME(*value*)

class PMML44Super.**SetReference**(*setId=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SetReference'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SetReference'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SetReference'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SetReference'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_setId()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_setId(*setId*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SetReference'*, *mapping_=None*)

validate_ELEMENT_ID(*value*)

class PMML44Super.**SigmoidKernelType**(*description=None*, *gamma='1'*, *coef0='1'*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SigmoidKernelType'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SigmoidKernelType'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SigmoidKernelType'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SigmoidKernelType'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_coef0()

get_description()

get_gamma()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_coef0(*coef0*)

set_description(*description*)

set_gamma(*gamma*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SigmoidKernelType'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**SimplePredicate**(*field=None*, *operator=None*, *value=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_='', name_='SimplePredicate', namespacedef_='', pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='', name_='SimplePredicate'*)

exportChildren(*outfile*, *level*, *namespace_='', name_='SimplePredicate', fromsubclass_=False, pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SimplePredicate'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

get_field()

get_operator()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_field(*field*)

set_operator(*operator*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None, name_='SimplePredicate', mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**SimpleRule**(*id=None, score=None, recordCount=None, nbCorrect=None, confidence='1', weight='1', Extension=None, SimplePredicate=None, CompoundPredicate=None, SimpleSetPredicate=None, True_=None, False_=None, ScoreDistribution=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_ScoreDistribution(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_='', name_='SimpleRule', namespacedef_='', pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='', name_='SimpleRule'*)

exportChildren(*outfile*, *level*, *namespace_='', name_='SimpleRule', fromsubclass_=False, pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SimpleRule'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_CompoundPredicate()

get_Extension()

get_False()

get_ScoreDistribution()

get_SimplePredicate()

get_SimpleSetPredicate()

get_True()

get_confidence()

get_id()

get_nbCorrect()

get_recordCount()

get_score()

get_weight()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_ScoreDistribution_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_ScoreDistribution_at(*index*, *value*)

set_CompoundPredicate(*CompoundPredicate*)

set_Extension(*Extension*)

set_False(*False_*)

set_ScoreDistribution(*ScoreDistribution*)

set_SimplePredicate(*SimplePredicate*)

set_SimpleSetPredicate(*SimpleSetPredicate*)

set_True(*True_*)

set_confidence(*confidence*)

set_id(*id*)

set_nbCorrect(*nbCorrect*)

set_recordCount(*recordCount*)

set_score(*score*)

set_weight(*weight*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SimpleRule'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**SimpleSetPredicate**(*field=None*, *booleanOperator=None*, *Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SimpleSetPredicate'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SimpleSetPredicate'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SimpleSetPredicate'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SimpleSetPredicate'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

get_booleanOperator()

get_field()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

set_booleanOperator(*booleanOperator*)

set_field(*field*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SimpleSetPredicate'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**SpectralAnalysis**

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SpectralAnalysis'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SpectralAnalysis'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SpectralAnalysis'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SpectralAnalysis'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

hasContent_()

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SpectralAnalysis'*, *mapping_=None*)

class PMML44Super.**StateSpaceModel**(*variance=None*, *period='none'*, *intercept='0'*, *Extension=None*, *StateVector=None*, *TransitionMatrix=None*, *MeasurementMatrix=None*, *PsiVector=None*, *DynamicRegressor=None*)

Bases: PMML44Super.GeneratedsSuper

add_DynamicRegressor(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='StateSpaceModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='StateSpaceModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='StateSpaceModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='StateSpaceModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DynamicRegressor()

get_Extension()

get_MeasurementMatrix()

get_PsiVector()

get_StateVector()

get_TransitionMatrix()

get_intercept()

get_period()

get_variance()

hasContent_()

insert_DynamicRegressor_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_DynamicRegressor_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_DynamicRegressor(*DynamicRegressor*)

set_Extension(*Extension*)

set_MeasurementMatrix(*MeasurementMatrix*)

set_PsiVector(*PsiVector*)

set_StateVector(*StateVector*)

set_TransitionMatrix(*TransitionMatrix*)

set_intercept(*intercept*)

set_period(*period*)

set_variance(*variance*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='StateSpaceModel'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**StateVector**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='StateVector'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='StateVector'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='StateVector'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='StateVector'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='StateVector'*, *mapping_=None*)

class PMML44Super.**SupportVector**(*vectorId=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SupportVector'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SupportVector'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SupportVector'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SupportVector'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_vectorId()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_vectorId(*vectorId*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SupportVector'*, *mapping_=None*)

validate_VECTOR_ID(*value*)

class PMML44Super.**SupportVectorMachine**(*targetCategory=None*, *alternateTargetCategory=None*, *threshold=None*, *Extension=None*, *SupportVectors=None*, *Coefficients=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SupportVectorMachine'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SupportVectorMachine'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SupportVectorMachine'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SupportVectorMachine'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Coefficients()

get_Extension()

get_SupportVectors()

get_alternateTargetCategory()

get_targetCategory()

get_threshold()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Coefficients(*Coefficients*)

set_Extension(*Extension*)

set_SupportVectors(*SupportVectors*)

set_alternateTargetCategory(*alternateTargetCategory*)

set_targetCategory(*targetCategory*)

set_threshold(*threshold*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SupportVectorMachine'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**SupportVectorMachineModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *threshold='0'*, *svmRepresentation='SupportVectors'*, *classificationMethod='OneAgainstAll'*, *maxWins=False*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *LinearKernelType=None*, *PolynomialKernelType=None*, *RadialBasisKernelType=None*, *SigmoidKernelType=None*, *VectorDictionary=None*, *SupportVectorMachine=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_SupportVectorMachine(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SupportVectorMachineModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SupportVectorMachineModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SupportVectorMachineModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SupportVectorMachineModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LinearKernelType()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_PolynomialKernelType()

get_RadialBasisKernelType()

get_SigmoidKernelType()

get_SupportVectorMachine()

get_Targets()

get_VectorDictionary()

get_algorithmName()

get_classificationMethod()

get_functionName()

get_isScorable()

get_maxWins()

get_modelName()

get_svmRepresentation()

get_threshold()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_SupportVectorMachine_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_SupportVectorMachine_at(*index*, *value*)

set_Extension(*Extension*)

set_LinearKernelType(*LinearKernelType*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_PolynomialKernelType(*PolynomialKernelType*)

set_RadialBasisKernelType(*RadialBasisKernelType*)

set_SigmoidKernelType(*SigmoidKernelType*)

set_SupportVectorMachine(*SupportVectorMachine*)

set_Targets(*Targets*)

set_VectorDictionary(*VectorDictionary*)

set_algorithmName(*algorithmName*)

set_classificationMethod(*classificationMethod*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_maxWins(*maxWins*)

set_modelName(*modelName*)

set_svmRepresentation(*svmRepresentation*)

set_threshold(*threshold*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SupportVectorMachineModel'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

validate_REAL_NUMBER(*value*)

validate_SVM_CLASSIFICATION_METHOD(*value*)

validate_SVM_REPRESENTATION(*value*)

class PMML44Super.**SupportVectors**(*numberOfSupportVectors=None*, *numberOfAttributes=None*, *Extension=None*, *SupportVector=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_SupportVector(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='SupportVectors'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='SupportVectors'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='SupportVectors'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='SupportVectors'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_SupportVector()

get_numberOfAttributes()

get_numberOfSupportVectors()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_SupportVector_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_SupportVector_at(*index*, *value*)

set_Extension(*Extension*)

set_SupportVector(*SupportVector*)

set_numberOfAttributes(*numberOfAttributes*)

set_numberOfSupportVectors(*numberOfSupportVectors*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='SupportVectors'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**TableLocator**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TableLocator'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TableLocator'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TableLocator'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TableLocator'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TableLocator'*, *mapping_=None*)

class PMML44Super.**Target**(*field=None*, *optype=None*, *castInteger=None*, *min=None*, *max=None*, *rescaleConstant=0*, *rescaleFactor=1*, *Extension=None*, *TargetValue=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_TargetValue(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Target'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Target'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Target'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Target'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_TargetValue()

get_castInteger()

get_field()

get_max()

get_min()

get_optype()

get_rescaleConstant()

get_rescaleFactor()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_TargetValue_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_TargetValue_at(*index*, *value*)

set_Extension(*Extension*)

set_TargetValue(*TargetValue*)

set_castInteger(*castInteger*)

set_field(*field*)

set_max(*max*)

set_min(*min*)

set_optype(*optype*)

set_rescaleConstant(*rescaleConstant*)

set_rescaleFactor(*rescaleFactor*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Target'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

validate_OPTYPE(*value*)

class PMML44Super.**TargetValue**(*value=None*, *displayValue=None*, *priorProbability=None*, *defaultValue=None*, *Extension=None*, *Partition=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TargetValue'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TargetValue'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TargetValue'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TargetValue'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Partition()

get_defaultValue()

get_displayValue()

get_priorProbability()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Partition(*Partition*)

set_defaultValue(*defaultValue*)

set_displayValue(*displayValue*)

set_priorProbability(*priorProbability*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TargetValue'*, *mapping_=None*)

validate_NUMBER(*value*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**TargetValueCount**(*value=None*, *count=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TargetValueCount'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TargetValueCount'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TargetValueCount'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TargetValueCount'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_count()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_count(*count*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TargetValueCount'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**TargetValueCounts**(*Extension=None*, *TargetValueCount=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_TargetValueCount(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TargetValueCounts'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TargetValueCounts'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TargetValueCounts'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TargetValueCounts'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_TargetValueCount()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_TargetValueCount_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_TargetValueCount_at(*index*, *value*)

set_Extension(*Extension*)

set_TargetValueCount(*TargetValueCount*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TargetValueCounts'*, *mapping_=None*)

class PMML44Super.**TargetValueStat**(*value=None*, *Extension=None*, *AnyDistribution=None*, *GaussianDistribution=None*, *PoissonDistribution=None*, *UniformDistribution=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TargetValueStat'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TargetValueStat'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TargetValueStat'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TargetValueStat'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_AnyDistribution()

get_Extension()

get_GaussianDistribution()

get_PoissonDistribution()

get_UniformDistribution()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_AnyDistribution(*AnyDistribution*)

set_Extension(*Extension*)

set_GaussianDistribution(*GaussianDistribution*)

set_PoissonDistribution(*PoissonDistribution*)

set_UniformDistribution(*UniformDistribution*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TargetValueStat'*, *mapping_=None*)

class PMML44Super.**TargetValueStats**(*Extension=None*, *TargetValueStat=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_TargetValueStat(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TargetValueStats'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TargetValueStats'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TargetValueStats'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TargetValueStats'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_TargetValueStat()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_TargetValueStat_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_TargetValueStat_at(*index*, *value*)

set_Extension(*Extension*)

set_TargetValueStat(*TargetValueStat*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TargetValueStats'*, *mapping_=None*)

class PMML44Super.**Targets**(*Extension=None*, *Target=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_Target(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Targets'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Targets'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Targets'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Targets'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Target()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_Target_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_Target_at(*index*, *value*)

set_Extension(*Extension*)

set_Target(*Target*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Targets'*, *mapping_=None*)

class PMML44Super.**Taxonomy**(*name=None*, *Extension=None*, *ChildParent=None*)

Bases: PMML44Super.GeneratedsSuper

add_ChildParent(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Taxonomy'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Taxonomy'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Taxonomy'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Taxonomy'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ChildParent()

get_Extension()

get_name()

hasContent_()

insert_ChildParent_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_ChildParent_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_ChildParent(*ChildParent*)

set_Extension(*Extension*)

set_name(*name*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Taxonomy'*, *mapping_=None*)

class PMML44Super.**TestDistributions**(*field=None*, *testStatistic=None*, *resetValue='0.0'*, *windowSize='0'*, *weightField=None*, *normalizationScheme=None*, *Extension=None*, *Baseline=None*, *Alternate=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TestDistributions'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TestDistributions'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TestDistributions'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TestDistributions'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Alternate()

get_Baseline()

get_Extension()

get_field()

get_normalizationScheme()

get_resetValue()

get_testStatistic()

get_weightField()

get_windowSize()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Alternate(*Alternate*)

set_Baseline(*Baseline*)

set_Extension(*Extension*)

set_field(*field*)

set_normalizationScheme(*normalizationScheme*)

set_resetValue(*resetValue*)

set_testStatistic(*testStatistic*)

set_weightField(*weightField*)

set_windowSize(*windowSize*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TestDistributions'*, *mapping_=None*)

validate_BASELINE_TEST_STATISTIC(*value*)

validate_FIELD_NAME(*value*)

validate_INT_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**TextCorpus**(*Extension=None*, *TextDocument=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_TextDocument(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TextCorpus'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TextCorpus'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TextCorpus'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TextCorpus'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_TextDocument()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_TextDocument_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_TextDocument_at(*index*, *value*)

set_Extension(*Extension*)

set_TextDocument(*TextDocument*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TextCorpus'*, *mapping_=None*)

class PMML44Super.**TextDictionary**(*Extension=None*, *Taxonomy=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TextDictionary'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TextDictionary'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TextDictionary'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TextDictionary'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

get_Taxonomy()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

set_Taxonomy(*Taxonomy*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TextDictionary'*, *mapping_=None*)

class PMML44Super.**TextDocument**(*id=None*, *name=None*, *length=None*, *file=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TextDocument'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TextDocument'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TextDocument'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TextDocument'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_file()

get_id()

get_length()

get_name()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_file(*file*)

set_id(*id*)

set_length(*length*)

set_name(*name*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TextDocument'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**TextIndex**(*textField=None*, *localTermWeights='termFrequency'*, *isCaseSensitive=False*, *maxLevenshteinDistance=0*, *countHits='allHits'*, *wordSeparatorCharacterRE='\s+'*, *tokenize=True*, *Extension=None*, *TextIndexNormalization=None*, *FieldRef=None*, *Constant=None*, *NormContinuous=None*, *NormDiscrete=None*, *Discretize=None*, *MapValues=None*, *TextIndex_member=None*, *Apply=None*, *Aggregate=None*, *Lag=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_TextIndexNormalization(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TextIndex'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TextIndex'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TextIndex'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TextIndex'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Aggregate()

get_Apply()

get_Constant()

get_Discretize()

get_Extension()

get_FieldRef()

get_Lag()

get_MapValues()

get_NormContinuous()

get_NormDiscrete()

get_TextIndex()

get_TextIndexNormalization()

get_countHits()

get_isCaseSensitive()

get_localTermWeights()

get_maxLevenshteinDistance()

get_textField()

get_tokenize()

get_wordSeparatorCharacterRE()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_TextIndexNormalization_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_TextIndexNormalization_at(*index*, *value*)

set_Aggregate(*Aggregate*)

set_Apply(*Apply*)

set_Constant(*Constant*)

set_Discretize(*Discretize*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_Lag(*Lag*)

set_MapValues(*MapValues*)

set_NormContinuous(*NormContinuous*)

set_NormDiscrete(*NormDiscrete*)

set_TextIndex(*TextIndex*)

set_TextIndexNormalization(*TextIndexNormalization*)

set_countHits(*countHits*)

set_isCaseSensitive(*isCaseSensitive*)

set_localTermWeights(*localTermWeights*)

set_maxLevenshteinDistance(*maxLevenshteinDistance*)

set_textField(*textField*)

set_tokenize(*tokenize*)

set_wordSeparatorCharacterRE(*wordSeparatorCharacterRE*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TextIndex'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**TextIndexNormalization**(*inField='string'*, *outField='stem'*, *regexField='regex'*, *recursive=False*, *isCaseSensitive=None*, *maxLevenshteinDistance=None*, *wordSeparatorCharacterRE=None*, *tokenize=None*, *Extension=None*, *TableLocator=None*, *InlineTable=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TextIndexNormalization'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TextIndexNormalization'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TextIndexNormalization'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TextIndexNormalization'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_InlineTable()

get_TableLocator()

get_inField()

get_isCaseSensitive()

get_maxLevenshteinDistance()

get_outField()

get_recursive()

get_regexField()

get_tokenize()

get_wordSeparatorCharacterRE()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_InlineTable(*InlineTable*)

set_TableLocator(*TableLocator*)

set_inField(*inField*)

set_isCaseSensitive(*isCaseSensitive*)

set_maxLevenshteinDistance(*maxLevenshteinDistance*)

set_outField(*outField*)

set_recursive(*recursive*)

set_regexField(*regexField*)

set_tokenize(*tokenize*)

set_wordSeparatorCharacterRE(*wordSeparatorCharacterRE*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TextIndexNormalization'*, *mapping_=None*)

class PMML44Super.**TextModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *numberOfTerms=None*, *numberOfDocuments=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *TextDictionary=None*, *TextCorpus=None*, *DocumentTermMatrix=None*, *TextModelNormalization=None*, *TextModelSimiliarity=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TextModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TextModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TextModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TextModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DocumentTermMatrix()

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_Targets()

get_TextCorpus()

get_TextDictionary()

get_TextModelNormalization()

get_TextModelSimiliarity()

get_algorithmName()

get_functionName()

get_isScorable()

get_modelName()

get_numberOfDocuments()

get_numberOfTerms()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_DocumentTermMatrix(*DocumentTermMatrix*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_Targets(*Targets*)

set_TextCorpus(*TextCorpus*)

set_TextDictionary(*TextDictionary*)

set_TextModelNormalization(*TextModelNormalization*)

set_TextModelSimiliarity(*TextModelSimiliarity*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

set_numberOfDocuments(*numberOfDocuments*)

set_numberOfTerms(*numberOfTerms*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TextModel'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

class PMML44Super.**TextModelNormalization**(*localTermWeights='termFrequency'*, *globalTermWeights='inverseDocumentFrequency'*, *documentNormalization='none'*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TextModelNormalization'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TextModelNormalization'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TextModelNormalization'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TextModelNormalization'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_documentNormalization()

get_globalTermWeights()

get_localTermWeights()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_documentNormalization(*documentNormalization*)

set_globalTermWeights(*globalTermWeights*)

set_localTermWeights(*localTermWeights*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TextModelNormalization'*, *mapping_=None*)

class PMML44Super.**TextModelSimiliarity**(*similarityType=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TextModelSimiliarity'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TextModelSimiliarity'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TextModelSimiliarity'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TextModelSimiliarity'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_similarityType()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_similarityType(*similarityType*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TextModelSimiliarity'*, *mapping_=None*)

class PMML44Super.**Theta**(*i=None*, *j=None*, *theta=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Theta'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Theta'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Theta'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Theta'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_i()

get_j()

get_theta()

hasContent_()

set_i(*i*)

set_j(*j*)

set_theta(*theta*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Theta'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**ThetaRecursionState**(*FinalNoise=None*, *FinalPredictedNoise=None*, *FinalTheta=None*, *FinalNu=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ThetaRecursionState'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ThetaRecursionState'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ThetaRecursionState'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ThetaRecursionState'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_FinalNoise()

get_FinalNu()

get_FinalPredictedNoise()

get_FinalTheta()

hasContent_()

set_FinalNoise(*FinalNoise*)

set_FinalNu(*FinalNu*)

set_FinalPredictedNoise(*FinalPredictedNoise*)

set_FinalTheta(*FinalTheta*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ThetaRecursionState'*, *mapping_=None*)

class PMML44Super.**Time**(*min=None*, *max=None*, *mean=None*, *standardDeviation=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Time'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Time'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Time'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Time'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_max()

get_mean()

get_min()

get_standardDeviation()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_max(*max*)

set_mean(*mean*)

set_min(*min*)

set_standardDeviation(*standardDeviation*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Time'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**TimeAnchor**(*type_=None*, *offset=None*, *stepsize=None*, *displayName=None*, *TimeCycle=None*, *TimeException=None*)

Bases: PMML44Super.GeneratedsSuper

add_TimeCycle(*value*)

add_TimeException(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TimeAnchor'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TimeAnchor'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TimeAnchor'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TimeAnchor'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_TimeCycle()

get_TimeException()

get_displayName()

get_offset()

get_stepsize()

get_type()

hasContent_()

insert_TimeCycle_at(*index*, *value*)

insert_TimeException_at(*index*, *value*)

replace_TimeCycle_at(*index*, *value*)

replace_TimeException_at(*index*, *value*)

set_TimeCycle(*TimeCycle*)

set_TimeException(*TimeException*)

set_displayName(*displayName*)

set_offset(*offset*)

set_stepsize(*stepsize*)

set_type(*type_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TimeAnchor'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_TIME_ANCHOR(*value*)

class PMML44Super.**TimeCycle**(*length=None*, *type_=None*, *displayName=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TimeCycle'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TimeCycle'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TimeCycle'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TimeCycle'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_displayName()

get_length()

get_type()

hasContent_()

set_Array(*Array*)

set_displayName(*displayName*)

set_length(*length*)

set_type(*type_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TimeCycle'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_VALID_TIME_SPEC(*value*)

class PMML44Super.**TimeException**(*type_=None*, *count=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TimeException'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TimeException'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TimeException'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TimeException'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_count()

get_type()

hasContent_()

set_Array(*Array*)

set_count(*count*)

set_type(*type_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TimeException'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_TIME_EXCEPTION_TYPE(*value*)

class PMML44Super.**TimeSeries**(*usage='original', startTime=None, endTime=None, interpolationMethod='none', field=None, TimeAnchor=None, TimeValue=None*)

Bases: PMML44Super.GeneratedsSuper

add_TimeValue(*value*)

build(*node*)

buildAttributes(*node, attrs, already_processed*)

buildChildren(*child_, node, nodeName_, fromsubclass_=False*)

export(*outfile, level, namespace_='', name_='TimeSeries', namespacedef_='', pretty_print=True*)

exportAttributes(*outfile, level, already_processed, namespace_='', name_='TimeSeries'*)

exportChildren(*outfile, level, namespace_='', name_='TimeSeries', fromsubclass_=False, pretty_print=True*)

exportLiteral(*outfile, level, name_='TimeSeries'*)

exportLiteralAttributes(*outfile, level, already_processed, name_*)

exportLiteralChildren(*outfile, level, name_*)

static factory(**args_, **kwargs_*)

get_TimeAnchor()

get_TimeValue()

get_endTime()

get_field()

get_interpolationMethod()

get_startTime()

get_usage()

hasContent_()

insert_TimeValue_at(*index, value*)

replace_TimeValue_at(*index, value*)

set_TimeAnchor(*TimeAnchor*)

set_TimeValue(*TimeValue*)

set_endTime(*endTime*)

set_field(*field*)

set_interpolationMethod(*interpolationMethod*)

set_startTime(*startTime*)

set_usage(*usage*)

subclass = None

superclass = None

to_etree(*parent_element=None, name_='TimeSeries', mapping_=None*)

validate_FIELD_NAME(*value*)

validate_INTERPOLATION_METHOD(*value*)

validate_REAL_NUMBER(*value*)

validate_TIMESERIES_USAGE(*value*)

class PMML44Super.**TimeSeriesModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *bestFit=None*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *LocalTransformations=None*, *TimeSeries=None*, *SpectralAnalysis=None*, *ARIMA=None*, *ExponentialSmoothing=None*, *SeasonalTrendDecomposition=None*, *StateSpaceModel=None*, *GARCH=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_TimeSeries(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TimeSeriesModel'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TimeSeriesModel'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TimeSeriesModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TimeSeriesModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_ARIMA()

get_ExponentialSmoothing()

get_Extension()

get_GARCH()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Output()

get_SeasonalTrendDecomposition()

get_SpectralAnalysis()

get_StateSpaceModel()

get_TimeSeries()

get_algorithmName()

get_bestFit()

get_functionName()

get_isScorable()

get_modelName()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_TimeSeries_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_TimeSeries_at(*index*, *value*)

set_ARIMA(*ARIMA*)

set_ExponentialSmoothing(*ExponentialSmoothing*)

set_Extension(*Extension*)

set_GARCH(*GARCH*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Output(*Output*)

set_SeasonalTrendDecomposition(*SeasonalTrendDecomposition*)

set_SpectralAnalysis(*SpectralAnalysis*)

set_StateSpaceModel(*StateSpaceModel*)

set_TimeSeries(*TimeSeries*)

set_algorithmName(*algorithmName*)

set_bestFit(*bestFit*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_modelName(*modelName*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TimeSeriesModel'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

validate_TIMESERIES_ALGORITHM(*value*)

class PMML44Super.**TimeValue**(*index=None*, *time=None*, *value=None*, *standardError=None*, *Timestamp=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TimeValue'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TimeValue'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TimeValue'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TimeValue'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Timestamp()

get_index()

get_standardError()

get_time()

get_value()

hasContent_()

set_Timestamp(*Timestamp*)

set_index(*index*)

set_standardError(*standardError*)

set_time(*time*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TimeValue'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

validate_NUMBER(*value*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**Timestamp**(*Extension=None*, *valueOf_=None*, *mixedclass_=None*, *content_=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Timestamp'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Timestamp'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Timestamp'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Timestamp'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_valueOf_()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_valueOf_(*valueOf_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Timestamp'*, *mapping_=None*)

class PMML44Super.**TrainingInstances**(*isTransformed=False*, *recordCount=None*, *fieldCount=None*, *Extension=None*, *InstanceFields=None*, *TableLocator=None*, *InlineTable=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TrainingInstances'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TrainingInstances'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TrainingInstances'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TrainingInstances'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_InlineTable()

get_InstanceFields()

get_TableLocator()

get_fieldCount()

get_isTransformed()

get_recordCount()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_InlineTable(*InlineTable*)

set_InstanceFields(*InstanceFields*)

set_TableLocator(*TableLocator*)

set_fieldCount(*fieldCount*)

set_isTransformed(*isTransformed*)

set_recordCount(*recordCount*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TrainingInstances'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**TrainingParameters**(*architectureName=None*, *dataset=None*, *framework=None*, *Extension=None*, *Losses=None*, *Metrics=None*, *Optimizers=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TrainingParameters'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TrainingParameters'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TrainingParameters'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TrainingParameters'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Losses()

get_Metrics()

get_Optimizers()

get_architectureName()

get_dataset()

get_framework()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Losses(*Losses*)

set_Metrics(*Metrics*)

set_Optimizers(*Optimizers*)

set_architectureName(*architectureName*)

set_dataset(*dataset*)

set_framework(*framework*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TrainingParameters'*, *mapping_=None*)

class PMML44Super.**TransferFunctionValues**(*Array=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TransferFunctionValues'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TransferFunctionValues'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TransferFunctionValues'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TransferFunctionValues'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

hasContent_()

set_Array(*Array*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TransferFunctionValues'*, *mapping_=None*)

class PMML44Super.**TransformationDictionary**(*Extension=None*, *DefineFunction=None*, *DerivedField=None*)

Bases: PMML44Super.GeneratedsSuper

add_DefineFunction(*value*)

add_DerivedField(*value*)

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TransformationDictionary'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TransformationDictionary'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TransformationDictionary'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TransformationDictionary'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_DefineFunction()

get_DerivedField()

get_Extension()

hasContent_()

insert_DefineFunction_at(*index*, *value*)

insert_DerivedField_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

replace_DefineFunction_at(*index*, *value*)

replace_DerivedField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_DefineFunction(*DefineFunction*)

set_DerivedField(*DerivedField*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TransformationDictionary'*, *mapping_=None*)

class PMML44Super.**TransitionMatrix**(*Extension=None*, *Matrix=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TransitionMatrix'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='*', *name_='TransitionMatrix'*)

exportChildren(*outfile*, *level*, *namespace_='*', *name_='TransitionMatrix'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TransitionMatrix'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Matrix()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Matrix(*Matrix*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TransitionMatrix'*, *mapping_=None*)

class PMML44Super.**TreeModel**(*modelName=None*, *functionName=None*, *algorithmName=None*, *missingValueStrategy='none'*, *missingValuePenalty='1.0'*, *noTrueChildStrategy='returnNullPrediction'*, *splitCharacteristic='multiSplit'*, *isScorable=True*, *MiningSchema=None*, *Output=None*, *ModelStats=None*, *ModelExplanation=None*, *Targets=None*, *LocalTransformations=None*, *Node=None*, *ModelVerification=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_='*', *name_='TreeModel'*, *namespacedef_='*', *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='*', *name_='TreeModel'*)

exportChildren(*outfile*, *level*, *namespace_='*', *name_='TreeModel'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TreeModel'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_LocalTransformations()

get_MiningSchema()

get_ModelExplanation()

get_ModelStats()

get_ModelVerification()

get_Node()

get_Output()

get_Targets()

get_algorithmName()

get_functionName()

get_isScorable()

get_missingValuePenalty()

get_missingValueStrategy()

get_modelName()

get_noTrueChildStrategy()

get_splitCharacteristic()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_LocalTransformations(*LocalTransformations*)

set_MiningSchema(*MiningSchema*)

set_ModelExplanation(*ModelExplanation*)

set_ModelStats(*ModelStats*)

set_ModelVerification(*ModelVerification*)

set_Node(*Node*)

set_Output(*Output*)

set_Targets(*Targets*)

set_algorithmName(*algorithmName*)

set_functionName(*functionName*)

set_isScorable(*isScorable*)

set_missingValuePenalty(*missingValuePenalty*)

set_missingValueStrategy(*missingValueStrategy*)

set_modelName(*modelName*)

set_noTrueChildStrategy(*noTrueChildStrategy*)

set_splitCharacteristic(*splitCharacteristic*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TreeModel'*, *mapping_=None*)

validate_MINING_FUNCTION(*value*)

validate_MISSING_VALUE_STRATEGY(*value*)

validate_NO_TRUE_CHILD_STRATEGY(*value*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**TrendCoefficients**(*Extension=None*, *REAL_SparseArray=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='TrendCoefficients'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='TrendCoefficients'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='TrendCoefficients'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TrendCoefficients'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_REAL_SparseArray()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_REAL_SparseArray(*REAL_SparseArray*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TrendCoefficients'*, *mapping_=None*)

class PMML44Super.**Trend_ExpoSmooth**(*trend='additive'*, *gamma=None*, *phi='1'*, *smoothedValue=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Trend_ExpoSmooth'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='*', *name_='Trend_ExpoSmooth'*)

exportChildren(*outfile*, *level*, *namespace_='*', *name_='Trend_ExpoSmooth'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Trend_ExpoSmooth'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_gamma()

get_phi()

get_smoothedValue()

get_trend()

hasContent_()

set_Array(*Array*)

set_gamma(*gamma*)

set_phi(*phi*)

set_smoothedValue(*smoothedValue*)

set_trend(*trend*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Trend_ExpoSmooth'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**TriangularDistributionForBN**(*Extension=None*, *Mean=None*, *Lower=None*, *Upper=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_='*', *name_='TriangularDistributionForBN'*, *namespacedef_='*', *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_='*', *name_='TriangularDistributionForBN'*)

exportChildren(*outfile*, *level*, *namespace_='*', *name_='TriangularDistributionForBN'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='TriangularDistributionForBN'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

get_Lower()

get_Mean()

get_Upper()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Lower(*Lower*)

set_Mean(*Mean*)

set_Upper(*Upper*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='TriangularDistributionForBN'*, *mapping_=None*)

class PMML44Super.**True_**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='True'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='True'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='True'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='True'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='True'*, *mapping_=None*)

class PMML44Super.**UniformDistribution**(*lower=None*, *upper=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='UniformDistribution'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='UniformDistribution'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='UniformDistribution'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='UniformDistribution'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_lower()

get_upper()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_lower(*lower*)

set_upper(*upper*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='UniformDistribution'*, *mapping_=None*)

validate_REAL_NUMBER(*value*)

class PMML44Super.**UniformDistributionForBN**(*Extension=None*, *Lower=None*, *Upper=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='UniformDistributionForBN'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='UniformDistributionForBN'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='UniformDistributionForBN'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='UniformDistributionForBN'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_Lower()

get_Upper()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_Lower(*Lower*)

set_Upper(*Upper*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='UniformDistributionForBN'*, *mapping_=None*)

class PMML44Super.**UnivariateStats**(*field=None*, *weighted='0'*, *Extension=None*, *Counts=None*, *NumericInfo=None*, *DiscrStats=None*, *ContStats=None*, *Anova=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='UnivariateStats'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='UnivariateStats'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='UnivariateStats'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='UnivariateStats'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Anova()

get_ContStats()

get_Counts()

get_DiscrStats()

get_Extension()

get_NumericInfo()

get_field()

get_weighted()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Anova(*Anova*)

set_ContStats(*ContStats*)

set_Counts(*Counts*)

set_DiscrStats(*DiscrStats*)

set_Extension(*Extension*)

set_NumericInfo(*NumericInfo*)

set_field(*field*)

set_weighted(*weighted*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='UnivariateStats'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**Upper**(*Extension=None*, *FieldRef=None*, *Constant=None*, *NormContinuous=None*, *NormDiscrete=None*, *Discretize=None*, *MapValues=None*, *TextIndex=None*, *Apply=None*, *Aggregate=None*, *Lag=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Upper'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Upper'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Upper'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Upper'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Aggregate()

get_Apply()

get_Constant()

get_Discretize()

get_Extension()

get_FieldRef()

get_Lag()

get_MapValues()

get_NormContinuous()

get_NormDiscrete()

get_TextIndex()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Aggregate(*Aggregate*)

set_Apply(*Apply*)

set_Constant(*Constant*)

set_Discretize(*Discretize*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_Lag(*Lag*)

set_MapValues(*MapValues*)

set_NormContinuous(*NormContinuous*)

set_NormDiscrete(*NormDiscrete*)

set_TextIndex(*TextIndex*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Upper'*, *mapping_=None*)

class PMML44Super.**Value**(*value=None*, *displayValue=None*, *property='valid'*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Value'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Value'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Value'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Value'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_displayValue()

get_property()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_displayValue(*displayValue*)

set_property(*property*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Value'*, *mapping_=None*)

class PMML44Super.**ValueProbability**(*value=None*, *probability=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='ValueProbability'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='ValueProbability'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='ValueProbability'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='ValueProbability'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_probability()

get_value()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_probability(*probability*)

set_value(*value*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='ValueProbability'*, *mapping_=None*)

validate_PROB_NUMBER(*value*)

class PMML44Super.**VariableWeight**(*field=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='VariableWeight'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='VariableWeight'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='VariableWeight'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='VariableWeight'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_field()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_field(*field*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='VariableWeight'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**Variance**(*Extension=None*, *FieldRef=None*, *Constant=None*, *NormContinuous=None*, *NormDiscrete=None*, *Discretize=None*, *MapValues=None*, *TextIndex=None*, *Apply=None*, *Aggregate=None*, *Lag=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='Variance'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='Variance'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='Variance'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='Variance'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Aggregate()

get_Apply()

get_Constant()

get_Discretize()

get_Extension()

get_FieldRef()

get_Lag()

get_MapValues()

get_NormContinuous()

get_NormDiscrete()

get_TextIndex()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Aggregate(*Aggregate*)

set_Apply(*Apply*)

set_Constant(*Constant*)

set_Discretize(*Discretize*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_Lag(*Lag*)

set_MapValues(*MapValues*)

set_NormContinuous(*NormContinuous*)

set_NormDiscrete(*NormDiscrete*)

set_TextIndex(*TextIndex*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='Variance'*, *mapping_=None*)

class PMML44Super.**VarianceCoefficients**(*Extension=None*, *PastVariances=None*, *MACoefficients=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='VarianceCoefficients'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='VarianceCoefficients'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='VarianceCoefficients'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='VarianceCoefficients'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_MACoefficients()

get_PastVariances()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_MACoefficients(*MACoefficients*)

set_PastVariances(*PastVariances*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='VarianceCoefficients'*, *mapping_=None*)

class PMML44Super.**VectorDictionary**(*numberOfVectors=None*, *Extension=None*, *VectorFields=None*, *VectorInstance=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_VectorInstance(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='VectorDictionary'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='VectorDictionary'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='VectorDictionary'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='VectorDictionary'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_VectorFields()

get_VectorInstance()

get_numberOfVectors()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_VectorInstance_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_VectorInstance_at(*index*, *value*)

set_Extension(*Extension*)

set_VectorFields(*VectorFields*)

set_VectorInstance(*VectorInstance*)

set_numberOfVectors(*numberOfVectors*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='VectorDictionary'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**VectorFields**(*numberOfFields=None*, *Extension=None*, *FieldRef=None*, *CategoricalPredictor=None*)

Bases: PMML44Super.GeneratedsSuper

add_CategoricalPredictor(*value*)

add_Extension(*value*)

add_FieldRef(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='VectorFields'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='VectorFields'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='VectorFields'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='VectorFields'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_CategoricalPredictor()

get_Extension()

get_FieldRef()

get_numberOfFields()

hasContent_()

insert_CategoricalPredictor_at(*index*, *value*)

insert_Extension_at(*index*, *value*)

insert_FieldRef_at(*index*, *value*)

replace_CategoricalPredictor_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_FieldRef_at(*index*, *value*)

set_CategoricalPredictor(*CategoricalPredictor*)

set_Extension(*Extension*)

set_FieldRef(*FieldRef*)

set_numberOfFields(*numberOfFields*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='VectorFields'*, *mapping_=None*)

validate_INT_NUMBER(*value*)

class PMML44Super.**VectorInstance**(*id=None*, *Extension=None*, *REAL_SparseArray=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='VectorInstance'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='VectorInstance'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='VectorInstance'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='VectorInstance'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

get_REAL_SparseArray()

get_id()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

set_REAL_SparseArray(*REAL_SparseArray*)

set_id(*id*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='VectorInstance'*, *mapping_=None*)

validate_VECTOR_ID(*value*)

class PMML44Super.**VerificationField**(*field=None*, *column=None*, *precision=1e-06*, *zeroThreshold=1e-16*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='VerificationField'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='VerificationField'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='VerificationField'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='VerificationField'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_column()

get_field()

get_precision()

get_zeroThreshold()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_column(*column*)

set_field(*field*)

set_precision(*precision*)

set_zeroThreshold(*zeroThreshold*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='VerificationField'*, *mapping_=None*)

validate_FIELD_NAME(*value*)

class PMML44Super.**VerificationFields**(*Extension=None*, *VerificationField=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

add_VerificationField(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='VerificationFields'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='VerificationFields'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='VerificationFields'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='VerificationFields'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_VerificationField()

hasContent_()

insert_Extension_at(*index*, *value*)

insert_VerificationField_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

replace_VerificationField_at(*index*, *value*)

set_Extension(*Extension*)

set_VerificationField(*VerificationField*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='VerificationFields'*, *mapping_=None*)

class PMML44Super.**XCoordinates**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='XCoordinates'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='XCoordinates'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='XCoordinates'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='XCoordinates'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='XCoordinates'*, *mapping_=None*)

class PMML44Super.**YCoordinates**(*Extension=None*, *Array=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='YCoordinates'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='YCoordinates'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='YCoordinates'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='YCoordinates'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Array()

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Array(*Array*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='YCoordinates'*, *mapping_=None*)

class PMML44Super.**binarySimilarity**(*c00_parameter=None*, *c01_parameter=None*, *c10_parameter=None*, *c11_parameter=None*, *d00_parameter=None*, *d01_parameter=None*, *d10_parameter=None*, *d11_parameter=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='binarySimilarity'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='binarySimilarity'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='binarySimilarity'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='binarySimilarity'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_c00_parameter()

get_c01_parameter()

get_c10_parameter()

get_c11_parameter()

get_d00_parameter()

get_d01_parameter()

get_d10_parameter()

get_d11_parameter()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_c00_parameter(*c00_parameter*)

set_c01_parameter(*c01_parameter*)

set_c10_parameter(*c10_parameter*)

set_c11_parameter(*c11_parameter*)

set_d00_parameter(*d00_parameter*)

set_d01_parameter(*d01_parameter*)

set_d10_parameter(*d10_parameter*)

set_d11_parameter(*d11_parameter*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='binarySimilarity'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**chebychev**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='chebychev'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='chebychev'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='chebychev'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='chebychev'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='chebychev'*, *mapping_=None*)

class PMML44Super.**cityBlock**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='cityBlock'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='cityBlock'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='cityBlock'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='cityBlock'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='cityBlock'*, *mapping_=None*)

class PMML44Super.**euclidean**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='euclidean'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='euclidean'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='euclidean'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='euclidean'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='euclidean'*, *mapping_=None*)

class PMML44Super.**jaccard**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='jaccard'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='jaccard'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='jaccard'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='jaccard'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_, **kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='jaccard'*, *mapping_=None*)

class PMML44Super.**minkowski**(*p_parameter=None*, *Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='minkowski'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='minkowski'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='minkowski'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='minkowski'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

get_p_parameter()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

set_p_parameter(*p_parameter*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='minkowski'*, *mapping_=None*)

validate_NUMBER(*value*)

class PMML44Super.**row**(*anytypeobjs_=None*)

Bases: PMML44Super.GeneratedsSuper

add_anytypeobjs_(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='row'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='row'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='row'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='row'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_anytypeobjs_()

hasContent_()

insert_anytypeobjs_(*index*, *value*)

set_anytypeobjs_(*anytypeobjs_*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='row'*, *mapping_=None*)

class PMML44Super.**simpleMatching**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='simpleMatching'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='simpleMatching'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='simpleMatching'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='simpleMatching'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='simpleMatching'*, *mapping_=None*)

class PMML44Super.**squaredEuclidean**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='squaredEuclidean'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='squaredEuclidean'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='squaredEuclidean'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='squaredEuclidean'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='squaredEuclidean'*, *mapping_=None*)

class PMML44Super.**tanimoto**(*Extension=None*)

Bases: PMML44Super.GeneratedsSuper

add_Extension(*value*)

build(*node*)

buildAttributes(*node*, *attrs*, *already_processed*)

buildChildren(*child_*, *node*, *nodeName_*, *fromsubclass_=False*)

export(*outfile*, *level*, *namespace_=''*, *name_='tanimoto'*, *namespacedef_=''*, *pretty_print=True*)

exportAttributes(*outfile*, *level*, *already_processed*, *namespace_=''*, *name_='tanimoto'*)

exportChildren(*outfile*, *level*, *namespace_=''*, *name_='tanimoto'*, *fromsubclass_=False*, *pretty_print=True*)

exportLiteral(*outfile*, *level*, *name_='tanimoto'*)

exportLiteralAttributes(*outfile*, *level*, *already_processed*, *name_*)

exportLiteralChildren(*outfile*, *level*, *name_*)

static factory(**args_*, ***kwargs_*)

get_Extension()

hasContent_()

insert_Extension_at(*index*, *value*)

replace_Extension_at(*index*, *value*)

set_Extension(*Extension*)

subclass = None

superclass = None

to_etree(*parent_element=None*, *name_='tanimoto'*, *mapping_=None*)

CHAPTER 3

arima module

CHAPTER 4

exponential_smoothing module

CHAPTER 5

keras_model_to_pmml module

CHAPTER 6

lgb_to_pmml module

CHAPTER 7

metadata module

CHAPTER 8

pre_process module

CHAPTER 9

skl_to_pmml module

CHAPTER 10

xgboost_to_pmml module

Python Module Index

m

p

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

X

Y