

Gradient Boosting Machine

Generated by Doxygen 1.8.9.1

Mon Jun 22 2015 09:31:22

Contents

1	Class Index	1
1.1	Class List	1
2	Class Documentation	3
2.1	BinaryTree Class Reference	3
2.1.1	Constructor & Destructor Documentation	3
2.1.1.1	BinaryTree	3
2.1.1.2	~BinaryTree	3
2.1.2	Member Function Documentation	3
2.1.2.1	computeVariance	3
2.1.2.2	generateTree	4
2.1.2.3	getGamma	4
2.1.2.4	getNumberOfRegions	4
2.1.2.5	makeAPrediction	4
2.1.2.6	print	4
2.1.2.7	printToFile	5
2.1.2.8	readFromBinaryFile	5
2.1.2.9	readFromBinaryFile	5
2.1.2.10	saveToBinaryFile	5
2.1.2.11	saveToBinaryFile	5
2.2	Config Class Reference	5
2.2.1	Member Function Documentation	6
2.2.1.1	print	6
2.2.2	Member Data Documentation	6
2.2.2.1	applyWeight	6
2.2.2.2	classWeight	6
2.2.2.3	inputFile	6
2.2.2.4	learningRate	6
2.2.2.5	MAXBUFFER	6
2.2.2.6	MAXIMUMDEPTH	6
2.2.2.7	MAXIMUMNUMBEROFLEAVES	6

2.2.2.8	MINIMNUMBEROFINSTANCESPERLEAF	6
2.2.2.9	MINVARIANCE	7
2.2.2.10	modelFile	7
2.2.2.11	multipleExperiments	7
2.2.2.12	NOTVALID	7
2.2.2.13	numberOfTrees	7
2.2.2.14	UNKNOWNVALUE	7
2.2.2.15	verbose	7
2.3	DataSet Class Reference	7
2.3.1	Constructor & Destructor Documentation	8
2.3.1.1	DataSet	8
2.3.1.2	DataSet	8
2.3.1.3	DataSet	8
2.3.1.4	~DataSet	8
2.3.2	Member Function Documentation	8
2.3.2.1	computeResidual	8
2.3.2.2	getExampleAt	9
2.3.2.3	getNumberOfInstancesWithThresholdAt	9
2.3.2.4	getOutputVariableAt	9
2.3.2.5	getResidualAt	9
2.3.2.6	getWeight	9
2.3.2.7	print	9
2.3.2.8	setResidualAt	10
2.3.2.9	sort	11
2.4	GradientBoosting Class Reference	11
2.4.1	Constructor & Destructor Documentation	11
2.4.1.1	GradientBoosting	11
2.4.1.2	~GradientBoosting	11
2.4.2	Member Function Documentation	11
2.4.2.1	test	11
2.4.2.2	train	11
2.5	NominalFeature Class Reference	12
2.5.1	Constructor & Destructor Documentation	12
2.5.1.1	NominalFeature	12
2.5.1.2	NominalFeature	12
2.5.1.3	~NominalFeature	12
2.5.2	Member Function Documentation	12
2.5.2.1	add	12
2.5.2.2	getIntValue	12
2.5.2.3	getName	12

CONTENTS	v
2.5.2.4 getNumberOfValues	13
2.5.2.5 getStringValue	13
2.5.2.6 setName	13
2.6 SAUXMap Struct Reference	13
Index	15

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

BinaryTree	3
Config	5
DataSet	7
GradientBoosting	11
NominalFeature	12
SAUXMap	13

Chapter 2

Class Documentation

2.1 BinaryTree Class Reference

Public Member Functions

- [BinaryTree](#) ()
- void [generateTree](#) (unsigned int classLabel, [DataSet](#) *data)
- [~BinaryTree](#) ()
- float [computeVariance](#) (float value, float sumSQValue, float weight)
- int [getNumberOfRegions](#) ()
- float [makeAPrediction](#) (const float *example) const
- void [saveToBinaryFile](#) () const
- void [readFromBinaryFile](#) ()
- void [readFromBinaryFile](#) (std::ifstream &mFile)
- void [saveToBinaryFile](#) (std::ofstream &oFile) const
- void [print](#) (const [DataSet](#) *data)
- void [printToFile](#) (const [DataSet](#) *data)
- float [getGamma](#) (const float *example) const

2.1.1 Constructor & Destructor Documentation

2.1.1.1 [BinaryTree::BinaryTree](#) ()

The object constructor.

2.1.1.2 [BinaryTree::~~BinaryTree](#) ()

The object destructor.

2.1.2 Member Function Documentation

2.1.2.1 float [BinaryTree::computeVariance](#) (float *value*, float *sumSQValue*, float *weight*)

It computes the variance of a single instance.

Parameters

<i>value</i>	is the value for computing a sample mean.
<i>sumSQValue</i>	is the sum of squared values.
<i>weight</i>	is the normalization term.

Returns

the sample variance.

2.1.2.2 void BinaryTree::generateTree (unsigned int *classLabel*, DataSet * *data*)

It builds the tree.

Parameters

<i>classLabel</i>	is the class for the tree.
<i>data</i>	is the data to fit with a tree.

2.1.2.3 float BinaryTree::getGamma (const float * *example*) const

It gets the gamma value for the given example.

Parameters

<i>example</i>	is the given example.
----------------	-----------------------

2.1.2.4 int BinaryTree::getNumberOfRegions ()

It returns the number of regions Ri in the tree.

Returns

the number of regions Ri of the tree.

2.1.2.5 float BinaryTree::makeAPrediction (const float * *example*) const

It makes a prediction of the expected value.

Parameters

<i>example</i>	is the given example to predict.
----------------	----------------------------------

Returns

the expected value.

2.1.2.6 void BinaryTree::print (const DataSet * *data*)

It prints by command line the model.

Parameters

<i>data</i>	is the data set.
-------------	------------------

2.1.2.7 void BinaryTree::printToFile (const DataSet * *data*)

It copies to a string the model.

Parameters

<i>data</i>	is the data set.
-------------	------------------

2.1.2.8 void BinaryTree::readFromBinaryFile ()

It reads the model to a file.

2.1.2.9 void BinaryTree::readFromBinaryFile (std::ifstream & *mFile*)

It reads the file.

Parameters

<i>mFile</i>	is the model file to read.
--------------	----------------------------

2.1.2.10 void BinaryTree::saveToBinaryFile () const

It writes the model to a file.

2.1.2.11 void BinaryTree::saveToBinaryFile (std::ofstream & *oFile*) const

It writes the model to a file.

Parameters

<i>oFile</i>	is the output file.
--------------	---------------------

The documentation for this class was generated from the following files:

- BinaryTree.h
- BinaryTree.cpp

2.2 Config Class Reference

Static Public Member Functions

- static void [print](#) ()

Static Public Attributes

- static const unsigned int [MAXBUFFER](#) = 3000
- static const int [NOTVALID](#) = std::numeric_limits<int>::min()
- static const float [UNKNOWNVALUE](#) = std::numeric_limits<float>::max()
- static float [MINVARIANCE](#) = 0.000001f

- static int `MINIMNUMBEROFINSTANCESPERLEAF` = 4
- static int `MAXIMUMDEPTH` = 1
- static int `MAXIMUMNUMBEROFLEAVES` = 0
- static std::string `inputFile` = "../GradientBoosting/DataSet/notas/notas-10-1tst.dat"
- static std::string `modelFile` = "model.bin"
- static float `learningRate` = 0.1f
- static int `numberOfTrees` = 10
- static bool `verbose` = false
- static bool `multipleExperiments` = false
- static float * `classWeight` = NULL
- static bool `applyWeight` = false

2.2.1 Member Function Documentation

2.2.1.1 void Config::print () [static]

It prints the configuration.

2.2.2 Member Data Documentation

2.2.2.1 bool Config::applyWeight = false [static]

Apply weight.

2.2.2.2 float * Config::classWeight = NULL [static]

It weights.

2.2.2.3 std::string Config::inputFile = "../GradientBoosting/DataSet/notas/notas-10-1tst.dat" [static]

Data set file name.

2.2.2.4 float Config::learningRate = 0.1f [static]

The learning rate of the GBM.

2.2.2.5 const unsigned int Config::MAXBUFFER = 3000 [static]

The maximum amount of characters in a char* string.

2.2.2.6 int Config::MAXIMUMDEPTH = 1 [static]

It indicates the maximum depth of the tree.

2.2.2.7 int Config::MAXIMUMNUMBEROFLEAVES = 0 [static]

It indicates the maximum number of leaves in the tree.

2.2.2.8 int Config::MINIMNUMBEROFINSTANCESPERLEAF = 4 [static]

It indicates the minimum number of instances that a leaf node can have.

2.2.2.9 `float Config::MINVARIANCE = 0.000001f` `[static]`

It indicates the minimum variance required for split.

2.2.2.10 `std::string Config::modelFile = "model.bin"` `[static]`

The model file.

2.2.2.11 `bool Config::multipleExperiments = false` `[static]`

Prepares te algorithm to launch multiple experiments.

2.2.2.12 `const int Config::NOTVALID = std::numeric_limits<int>::min()` `[static]`

It indicates a non valid feature value.

2.2.2.13 `int Config::numberOfTrees = 10` `[static]`

The number of trees (rounds).

2.2.2.14 `const float Config::UNKNOWNVALUE = std::numeric_limits<float>::max()` `[static]`

An unknown value.

2.2.2.15 `bool Config::verbose = false` `[static]`

It prints the tree.

The documentation for this class was generated from the following files:

- Config.h
- Config.cpp

2.3 DataSet Class Reference

Public Member Functions

- [DataSet](#) (const std::string &fileName)
- [DataSet](#) (const [DataSet](#) *dataset)
- [DataSet](#) (unsigned int feature, const std::string &symbol, float threshold, const [DataSet](#) *dataset)
- [~DataSet](#) ()
- void [sort](#) (unsigned int feature)
- void [print](#) () const
- unsigned int [getNumberOfInstances](#) () const
- unsigned int [getNumberOfClasses](#) () const
- unsigned int [getNumberOfFeatures](#) () const
- unsigned int [getClassPosition](#) () const
- float [getValueAt](#) (unsigned int index, unsigned int feature) const
- float [getExampleAt](#) (unsigned int index, float *example) const
- float [getWeight](#) () const
- float [getResidualAt](#) (unsigned int index, unsigned int label) const

- void [setResidualAt](#) (unsigned int index, unsigned int label, float value)
- float [getOutputVariableAt](#) (unsigned int index) const
- int [getNumberOfInstancesWithThresholdAt](#) (unsigned int feature, const std::string &symbol, float threshold) const
- std::string [getFeatureName](#) (unsigned int index) const
- void [computeResidual](#) (unsigned int exampleIndex, unsigned int classIndex, float value)
- int [getNumberOfInstancesOfClass](#) (int classLabel)

2.3.1 Constructor & Destructor Documentation

2.3.1.1 DataSet::DataSet (const std::string & *fileName*)

The data set constructor.

Parameters

<i>fileName</i>	is the name of the file to open.
-----------------	----------------------------------

2.3.1.2 DataSet::DataSet (const DataSet * *dataset*)

Copy constructor.

Parameters

<i>dataset</i>	is the data set to clone.
----------------	---------------------------

2.3.1.3 DataSet::DataSet (unsigned int *feature*, const std::string & *symbol*, float *threshold*, const DataSet * *dataset*)

It copies the desired number of instances with threshold.

Parameters

<i>feature</i>	is the desired feature to copy data from.
<i>symbol</i>	is the desired symbol [\leq $>$].
<i>threshold</i>	is the threshold value.
<i>dataset</i>	is the data set to clone.

2.3.1.4 DataSet::~~DataSet ()

The data set destructor.

2.3.2 Member Function Documentation

2.3.2.1 void DataSet::computeResidual (unsigned int *exampleIndex*, unsigned int *classIndex*, float *value*)

It computes the residual for the given class.

Parameters

<i>exampleIndex</i>	is the index of the desired example.
<i>classIndex</i>	is the index of the desired class.

<i>value</i>	is the value to subtract to the residual.
--------------	---

2.3.2.2 float DataSet::getExampleAt (unsigned int *index*, float * *example*) const

It gets the example at the given position and example.

2.3.2.3 int DataSet::getNumberOfInstancesWithThresholdAt (unsigned int *feature*, const std::string & *symbol*, float *threshold*) const

It counts the number of instances which follow a given threshold.

Parameters

<i>feature</i>	is the feature index.
<i>symbol</i>	is the symbol [<i><=</i> <i>></i>].
<i>threshold</i>	is the threshold value.

2.3.2.4 float DataSet::getOutputVariableAt (unsigned int *index*) const [inline]

It gets the output variable (class) at the given position.

Parameters

<i>index</i>	is the index of the data matrix.
--------------	----------------------------------

Returns

the desired output variable.

It gets the output variable.

2.3.2.5 float DataSet::getResidualAt (unsigned int *index*, unsigned int *label*) const

It gets the residual at the given position and label.

Parameters

<i>index</i>	is the index of the data matrix.
<i>label</i>	is the index of the desired label.

Returns

the desired residual.

2.3.2.6 float DataSet::getWeight () const [inline]

Weight per example.

It gets the weight of each example.

2.3.2.7 void DataSet::print () const

It prints the data set.

2.3.2.8 void DataSet::setResidualAt (unsigned int *index*, unsigned int *label*, float *value*)

It sets the residual at the given position and label.

Parameters

<i>index</i>	is the index of the data matrix.
<i>label</i>	is the index of the desired label.
<i>value</i>	is the new value for the residual.

2.3.2.9 void DataSet::sort (unsigned int *feature*)

It sorts the data according to the given feature index by using iterative quicksort algorithm.

Parameters

<i>feature</i>	is the feature to sort.
----------------	-------------------------

The documentation for this class was generated from the following files:

- DataSet.h
- DataSet.cpp

2.4 GradientBoosting Class Reference

Public Member Functions

- [GradientBoosting](#) ()
- [~GradientBoosting](#) ()
- void [train](#) ()
- void [test](#) ()

2.4.1 Constructor & Destructor Documentation

2.4.1.1 GradientBoosting::GradientBoosting ()

It constructs the gradient boosting object.

2.4.1.2 GradientBoosting::~~GradientBoosting ()

It destructs the gradient boosting object.

2.4.2 Member Function Documentation

2.4.2.1 void GradientBoosting::test ()

It predicts.

2.4.2.2 void GradientBoosting::train ()

It trains.

The documentation for this class was generated from the following files:

- GradientBoosting.h
- GradientBoosting.cpp

2.5 NominalFeature Class Reference

Public Member Functions

- [NominalFeature](#) ()
- [NominalFeature](#) (const std::string &featureName)
- [~NominalFeature](#) ()
- void [setName](#) (std::string name)
- std::string [getName](#) () const
- void [add](#) (const std::string &featureValue)
- int [getIntValue](#) (std::string value) const
- std::string [getStringValue](#) (int index) const
- unsigned int [getNumberOfValues](#) () const

2.5.1 Constructor & Destructor Documentation

2.5.1.1 `NominalFeature::NominalFeature ()` `[inline]`

It sets the object to an empty state.

2.5.1.2 `NominalFeature::NominalFeature (const std::string & featureName)` `[inline]`

It constructs the nominal feature object.

Parameters

<i>name</i>	is the name of the feature.
-------------	-----------------------------

2.5.1.3 `NominalFeature::~~NominalFeature ()` `[inline]`

The nominal feature destructor.

2.5.2 Member Function Documentation

2.5.2.1 `void NominalFeature::add (const std::string & featureValue)` `[inline]`

It adds a new value to the object.

Parameters

<i>featureValue</i>	is the string containing the feature value.
---------------------	---

2.5.2.2 `int NominalFeature::getIntValue (std::string value) const` `[inline]`

It returns the index value of the feature.

Returns

the index value.

2.5.2.3 `std::string NominalFeature::getName () const` `[inline]`

It gets the name.

2.5.2.4 `unsigned int NominalFeature::getNumberOfValues () const [inline]`

It returns the number of values in the feature.

Returns

the number of values.

2.5.2.5 `std::string NominalFeature::getStringValue (int index) const [inline]`

It returns the string value of the feature.

Returns

the value.

2.5.2.6 `void NominalFeature::setName (std::string name) [inline]`

It sets the name.

The documentation for this class was generated from the following file:

- NominalFeature.h

2.6 SAUXMap Struct Reference

Public Attributes

- int **nivell**
- int **repetits**

The documentation for this struct was generated from the following file:

- BinaryTree.h

Index

- ~BinaryTree
 - BinaryTree, 3
- ~DataSet
 - DataSet, 8
- ~GradientBoosting
 - GradientBoosting, 11
- ~NominalFeature
 - NominalFeature, 12
- add
 - NominalFeature, 12
- applyWeight
 - Config, 6
- BinaryTree, 3
 - ~BinaryTree, 3
 - BinaryTree, 3
 - computeVariance, 3
 - generateTree, 4
 - getGamma, 4
 - getNumberOfRegions, 4
 - makeAPrediction, 4
 - print, 4
 - printToFile, 5
 - readFromBinaryFile, 5
 - saveToBinaryFile, 5
- classWeight
 - Config, 6
- computeResidual
 - DataSet, 8
- computeVariance
 - BinaryTree, 3
- Config, 5
 - applyWeight, 6
 - classWeight, 6
 - inputFile, 6
 - learningRate, 6
 - MAXBUFFER, 6
 - MAXIMUMDEPTH, 6
 - MAXIMUMNUMBEROFLEAVES, 6
 - MINIMUMNUMBEROFINSTANCESPERLEAF, 6
 - MINVARIANCE, 6
 - modelFile, 7
 - multipleExperiments, 7
 - NOTVALID, 7
 - numberOfTrees, 7
 - print, 6
 - UNKNOWNVALUE, 7
 - verbose, 7
- DataSet, 7
 - ~DataSet, 8
 - computeResidual, 8
 - DataSet, 8
 - getExampleAt, 9
 - getNumberOfInstancesWithThresholdAt, 9
 - getOutputVariableAt, 9
 - getResidualAt, 9
 - getWeight, 9
 - print, 9
 - setResidualAt, 9
 - sort, 11
- generateTree
 - BinaryTree, 4
- getExampleAt
 - DataSet, 9
- getGamma
 - BinaryTree, 4
- getIntValue
 - NominalFeature, 12
- getName
 - NominalFeature, 12
- getNumberOfInstancesWithThresholdAt
 - DataSet, 9
- getNumberOfRegions
 - BinaryTree, 4
- getNumberOfValues
 - NominalFeature, 12
- getOutputVariableAt
 - DataSet, 9
- getResidualAt
 - DataSet, 9
- getStringValue
 - NominalFeature, 13
- getWeight
 - DataSet, 9
- GradientBoosting, 11
 - ~GradientBoosting, 11
 - GradientBoosting, 11
 - test, 11
 - train, 11
- inputFile
 - Config, 6
- learningRate
 - Config, 6
- MAXBUFFER

- Config, [6](#)
- MAXIMUMDEPTH
 - Config, [6](#)
- MAXIMUMNUMBEROFLEAVES
 - Config, [6](#)
- MINIMUMNUMBEROFINSTANCESPERLEAF
 - Config, [6](#)
- MINVARIANCE
 - Config, [6](#)
- makeAPrediction
 - BinaryTree, [4](#)
- modelFile
 - Config, [7](#)
- multipleExperiments
 - Config, [7](#)
- NOTVALID
 - Config, [7](#)
- NominalFeature, [12](#)
 - ~NominalFeature, [12](#)
 - add, [12](#)
 - getIntValue, [12](#)
 - getName, [12](#)
 - getNumberOfValues, [12](#)
 - getStringValue, [13](#)
 - NominalFeature, [12](#)
 - setName, [13](#)
- numberOfTrees
 - Config, [7](#)
- print
 - BinaryTree, [4](#)
 - Config, [6](#)
 - DataSet, [9](#)
- printToFile
 - BinaryTree, [5](#)
- readFromBinaryFile
 - BinaryTree, [5](#)
- SAUXMap, [13](#)
- saveToBinaryFile
 - BinaryTree, [5](#)
- setName
 - NominalFeature, [13](#)
- setResidualAt
 - DataSet, [9](#)
- sort
 - DataSet, [11](#)
- test
 - GradientBoosting, [11](#)
- train
 - GradientBoosting, [11](#)
- UNKNOWNVALUE
 - Config, [7](#)
- verbose
 - Config, [7](#)