Machine Learning C++

Generated by Doxygen 1.8.6

Sat Aug 19 2017 23:34:13

Contents

Chapter 1

MLcpp

A Machine Learning Library written in C++ Practice the state-of-art machine learning algorithms in C++ (Growing weekly by 1 or 2 algorithms).

Algorithms included:

- Artificial neural network
- Support vector machines

Step one: Set up the third party libraries

- Boost C++ libraries

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

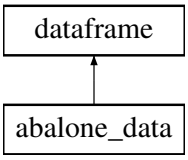
Class Documentation

4.1 abalone_data Struct Reference

The UCI data abalone data set.

`#include <abalone.h>`

Inheritance diagram for abalone_data:



Public Member Functions

- void load (std::string &path)
- void data_info ()

Additional Inherited Members

4.1.1 Detailed Description

The UCI data abalone data set.

4.1.2 Member Function Documentation

4.1.2.1 void abalone_data::data_info ( ) [virtual]

print data info

Reimplemented from dataframe.

4.1.2.2 void abalone_data::load (std::string & *path*) [virtual]

load data from file

Parameters

path	string path to file

Reimplemented from dataframe.

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

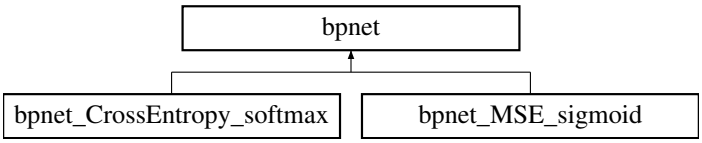
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/examples/abalone.h
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/examples/abalone.cpp

4.2 bpnet Class Reference

A backpropagation neural network class The back-propagation network has the architecture of three components propagate, update and train.

`#include <nnet.h>`

Inheritance diagram for bpnet:



Public Member Functions

- bpnet (int _n_input, int _n_neurons_in, int _n_output, std::vector< int > _hidden_layers, int _n_hidden_layers)
- virtual ~bpnet ()
- virtual void create ()
- virtual int get_n_hidden_layers ()
- void propagate (const std::vector< double > &input)
- void update (int layer_index)
- virtual double train (const std::vector< double > &train_data, const std::vector< double > &train_class, double learning_rate, double momentum)
- virtual void get_output (std::vector< double > &input, std::vector< double > &output)

Protected Attributes

- std::unique_ptr< layer > input_layer
- std::unique_ptr< layer > output_layer
- std::vector< std::unique_ptr < layer > > hidden_layers
- int **n_hidden_layers**
- int **n_input**
- int **n_neurons_in**
- int **n_output**
- std::vector< int > **hidden_layer_layout**

4.2.1 Detailed Description

A backpropagation neural network class The back-propagation network has the architecture of three components propagate, update and train.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 bpnet::bpnet (int *_n_input,* int *_n_neurons_in,* int *_n_output,* std::vector< int > *_hidden_layers,* int *_n_hidden_layers*)

bpnet constructor

Parameters

_n_input	number of input variables.
_n_neurons_in	number of neurons in input layer.
_n_ouput	number of output
_hidden_layers	hidden_layers size vector
_n_hidden_layers	number of hidden layers

4.2.2.2 virtual bpnet::~bpnet ( ) [inline], [virtual]

bpnet destructor

4.2.3 Member Function Documentation

4.2.3.1 virtual void bpnet::create ( ) [inline], [virtual]

a virtual function to create neural network

Reimplemented in bpnet_CrossEntropy_softmax, and bpnet_MSE_sigmoid.

4.2.3.2 virtual int bpnet::get_n_hidden_layers ( ) [inline], [virtual]

a function to get number of hidden layers

Returns

number of hidden layers

Reimplemented in bpnet_CrossEntropy_softmax, and bpnet_MSE_sigmoid.

4.2.3.3 void bpnet::get_output (std::vector< double > & *input,* std::vector< double > & *output*) [virtual]

a virtual function to get output class labels

Parameters

input	input data
output	output class label

Reimplemented in bpnet_CrossEntropy_softmax, and bpnet_MSE_sigmoid.

4.2.3.4 void bpnet::propagate (const std::vector< double > & *input*)

a function to perform forward feeding of data

Parameters

input	the input data vector

4.2.3.5 virtual double bpnet::train (const std::vector< double > & *train_data*, const std::vector< double > & *train_class*, double *learning_rate*, double *momentum*) [inline], [virtual]

a virtual function to train data

Parameters

train_data	training data
train_class	training data class label
learning_rate	learning rate
momentum	momentum or damping factor

Returns

loss value

Reimplemented in bpnet_CrossEntropy_softmax, and bpnet_MSE_sigmoid.

4.2.3.6 void bpnet::update (int *layer_index*)

a function to update

Parameters

layer_index	the layer index

4.2.4 Member Data Documentation

4.2.4.1 std::vector<std::unique_ptr<layer> > bpnet::hidden_layers [protected]

hidden layers holder

4.2.4.2 std::unique_ptr<layer> bpnet::input_layer [protected]

input layer

4.2.4.3 std::unique_ptr<layer> bpnet::output_layer [protected]

output layer

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

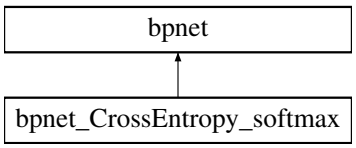
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/nnet.h
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/nnet.cpp

4.3 bpnet_CrossEntropy_softmax Class Reference

child class, backpropagation foward feed nerual net using croos entropy loss and softmax activation. Usually a good choice for multi-classification problems.

`#include <nnet.h>`

Inheritance diagram for bpnet_CrossEntropy_softmax:



Public Member Functions

- bpnet_CrossEntropy_softmax (int n_input, int n_neurons_in, int n_output, std::vector< int > _hidden_layers, int _n_hidden_layers)
- ~bpnet_CrossEntropy_softmax ()
- void create ()
- void propagate (const std::vector< double > &input)
- void update (int layer_index)
- double train (const std::vector< double > &train_data, const std::vector< double > &train_class, double learning_rate, double momentum)
- int get_n_hidden_layers ()
- void get_output (std::vector< double > &input, std::vector< double > &output)

Additional Inherited Members

4.3.1 Detailed Description

child class, backpropagation foward feed nerual net using croos entropy loss and softmax activation. Usually a good choice for multi-classification problems.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 bpnet_CrossEntropy_softmax::bpnet_CrossEntropy_softmax (int *n_input*, int *n_neurons_in*, int *n_output*, std::vector< int > *_hidden_layers*, int *_n_hidden_layers*) [inline]

constructor initialize from base class

Parameters

_n_input	number of input variables.
_n_neurons_in	number of neurons in input layer.
_n_ouput	number of output
_hidden_layers	hidden_layers size vector
_n_hidden_layers	number of hidden layers

4.3.2.2 bpnet_CrossEntropy_softmax::~bpnet_CrossEntropy_softmax ( ) [inline]

default destructor

4.3.3 Member Function Documentation

4.3.3.1 void bpnet_CrossEntropy_softmax::create ( ) [virtual]

create network

Reimplemented from bpnet.

4.3.3.2 int bpnet_CrossEntropy_softmax::get_n_hidden_layers ( ) [inline], [virtual]

get number of hidden layers

Returns

number of hidden layers

Reimplemented from bpnet.

4.3.3.3 void bpnet_CrossEntropy_softmax::get_output (std::vector< double > & *input*, std::vector< double > & *output*) [virtual]

get output class labels

Parameters

input	input data
output	output class label

Reimplemented from bpnet.

4.3.3.4 void bpnet_CrossEntropy_softmax::propagate (const std::vector< double > & *input*)

forward feeding of data

Parameters

input	the input data vector

4.3.3.5 double bpnet_CrossEntropy_softmax::train (const std::vector< double > & *train_data*, const std::vector< double > & *train_class*, double *learning_rate*, double *momentum*) [virtual]

training function

Parameters

train_data	training data
train_class	training data class label
learning_rate	learning rate
momentum	momentum or damping factor

Returns

loss value

Reimplemented from bpnet.

4.3.3.6 void bpnet_CrossEntropy_softmax::update (int *layer_index*)

update a layer

Parameters

layer_index	the layer index

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

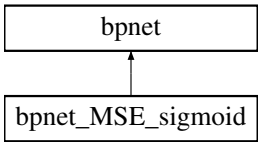
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/nnet.h
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/nnet.cpp

4.4 bpnet_MSE_sigmoid Class Reference

child class, backpropagation foward feed nerual net using mean squared error loss and sigmoid activation. Usually a good choice for binary classification problems.

`#include <nnet.h>`

Inheritance diagram for bpnet_MSE_sigmoid:



Public Member Functions

- bpnet_MSE_sigmoid (int n_input, int n_neurons_in, int n_output, std::vector< int > _hidden_layers, int _n_hidden_layers)
- ~bpnet_MSE_sigmoid ()
- void create ()
- void propagate (const std::vector< double > &input)
- void update (int layer_index)
- double train (const std::vector< double > &train_data, const std::vector< double > &train_class, double learning_rate, double momentum)
- int get_n_hidden_layers ()
- void get_output (std::vector< double > &input, std::vector< double > &output)

Additional Inherited Members

4.4.1 Detailed Description

child class, backpropagation foward feed nerual net using mean squared error loss and sigmoid activation. Usually a good choice for binary classification problems.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 bpnet_MSE_sigmoid::bpnet_MSE_sigmoid (int *n_input*, int *n_neurons_in*, int *n_output*, std::vector< int > *_hidden_layers*, int *_n_hidden_layers*) [inline]

constructor initialize from base class

Parameters

_n_input	number of input variables.
_n_neurons_in	number of neurons in input layer.
_n_ouput	number of output
_hidden_layers	hidden_layers size vector
_n_hidden_layers	number of hidden layers

4.4.2.2 bpnet_MSE_sigmoid::~bpnet_MSE_sigmoid ( ) [inline]

destructor

4.4.3 Member Function Documentation

4.4.3.1 void bpnet_MSE_sigmoid::create ( ) [virtual]

create network

Reimplemented from bpnet.

4.4.3.2 int bpnet_MSE_sigmoid::get_n_hidden_layers ( ) [inline], [virtual]

get number of hidden layers

Returns

number of hidden layers

Reimplemented from bpnet.

4.4.3.3 void bpnet_MSE_sigmoid::get_output (std::vector< double > & *input*, std::vector< double > & *output*) [virtual]

get output class labels

Parameters

input	input data
output	output class label

Reimplemented from bpnet.

4.4.3.4 void bpnet_MSE_sigmoid::propagate (const std::vector< double > & *input*)

forward feeding of data

Parameters

input	the input data vector

4.4.3.5 double bpnet_MSE_sigmoid::train (const std::vector< double > & *train_data*, const std::vector< double > & *train_class*, double *learning_rate*, double *momentum*) [virtual]

training function

Parameters

train_data	training data
train_class	training data class label
learning_rate	learning rate
momentum	momentum or damping factor

Returns

loss value

Reimplemented from bpnet.

4.4.3.6 void bpnet_MSE_sigmoid::update (int *layer_index*)

update a layer

Parameters

layer_index	the layer index

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

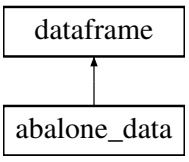
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/nnet.h
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/nnet.cpp

4.5 dataframe Struct Reference

The data framework for classification problems. It defines the way to load data and handle categorical class variables and display data info.

`#include <dataframe.h>`

Inheritance diagram for dataframe:



Public Member Functions

- dataframe ()
- virtual ~dataframe ()
- virtual void load (std::string &path)
- virtual void data_info ()

Public Attributes

- int **n_instance**
- int **n_attributes**
- std::vector< std::vector < double > > all_label
- std::vector< std::vector < double > > all_feature

4.5.1 Detailed Description

The data framework for classification problems. It defines the way to load data and handle categorical class variables and display data info.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 dataframe::dataframe ( ) [inline]

constructor

4.5.2.2 virtual dataframe::~dataframe ( ) [inline], [virtual]

destructor

4.5.3 Member Function Documentation

4.5.3.1 virtual void dataframe::data_info ( ) [inline], [virtual]

print data info

Reimplemented in abalone_data.

4.5.3.2 virtual void dataframe::load (std::string & *path*) [inline], [virtual]

load data from file

Parameters

path	string path to file

Reimplemented in abalone_data.

4.5.4 Member Data Documentation

4.5.4.1 std::vector<std::vector<double> > dataframe::all_feature

feature sets

4.5.4.2 std::vector<std::vector<double> > dataframe::all_label

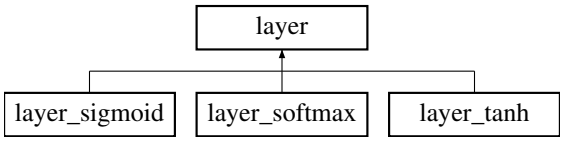
all class label which is transformed into numerical values

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

- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/examples/dataframe.h

4.6 layer Struct Reference

Inheritance diagram for layer:



Public Member Functions

- void **create** (int _n_input, int _n_neuron)
- virtual void **calculate** ()=0

Public Attributes

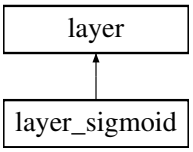
- std::vector< neuron > **neurons**
- std::vector< double > **layerinput**
- int **n_neuron**
- int **n_input**

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

- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/layer.h
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/layer.cpp

4.7 layer_sigmoid Struct Reference

Inheritance diagram for layer_sigmoid:



Public Member Functions

- void **calculate** ()

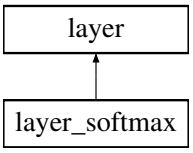
Additional Inherited Members

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

- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/layer.h
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/layer.cpp

4.8 layer_softmax Struct Reference

Inheritance diagram for layer_softmax:



Public Member Functions

- void **calculate** ()

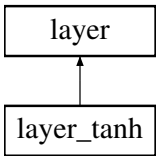
Additional Inherited Members

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

- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/layer.h
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/layer.cpp

4.9 layer_tanh Struct Reference

Inheritance diagram for layer_tanh:



Public Member Functions

- void **calculate** ()

Additional Inherited Members

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

- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/layer.h
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/layer.cpp

4.10 neuron Struct Reference

Public Member Functions

- void **create** (int n_input)
- void **activate** ()
- void **deactivate** ()

Public Attributes

- std::vector< double > **weights**
- std::vector< double > **deltas**
- double **output**
- double **bias**
- double **w_bias**
- bool **active**

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

- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/neuron.h
- /home/jiguangshen/HPC_MachineLearning/MLcpp/src/neural_network/neuron.cpp