Politechnika Śląska
Wydział Informatyki, Elektroniki i Informatyki

Computer Programming

«Neural Network»

author	Olivier Halupczok
instructor	dr inż. Piotr Fabian
year	2021-2022
lab group	even Tuesday, 10:15 – 11:45
deadline	2022-06-30

1 Project's topic

Implement Simple Neural Network defined as a class that supports Neurons, configuring a user-defined connection structure, selected learning method.

Porgram input data are stored in the root folder in the input.csv file. Input file holds following CSV format:

```
Label-Header    Header1    Header2    Header-i
Label1   DataFeature1   DataFeature2   DataFeature-i
Label2   DataFeature1   DataFeature2   DataFeature-i
Label3   DataFeature1   DataFeature2   DataFeature-i
Label4   DataFeature1   DataFeature2   DataFeature-i
...
```

Then the program analyzes features and labels and on their basis calculates the loss funcion's value. Derivative of this function is used to adjust weights and biases of the Neurons in the object of Neural Network class.

2 Analysis of the task

The task focuses on the analyzing data given as an input dataset and predicting the label of the object. It requires implementation of the learning algorithm and minimizing loss of the entire neural network.

2.1 Data structures

The program uses data structures like vector and set. The target of using set is to store labels, loaded from input file, and store it in a structure, which stores only unique values. This simplifies counting the number of unique labels, which are then encoded and passed to the neural network as an argument. Program uses also various custom classes like NeuralNetwork, Neurons, Neuron, to organize data in convenient way, which is assumed by the concept of Neural Network and its common understanding.

2.2 Algorithms

To achieve the main goal of the function, Program uses algorithm called Stochastic Gradient Descent to adjust properties of the neurons: weights and biases. These properties are elements to calculate a value of the neuron using activation function which is passed to a constructor of the neuron object along with its derivative. The default activation function is sigmoid function:

$$sigmoid(x) = \frac{1}{1 + e^{-x}}$$

, where

$$x_{i,j} = \overrightarrow{inputs} \circ \overrightarrow{weights} + b_{i,j}$$

, `i` is the index of the layer, `j` is the index of the neuron in the `i-th` layer. Inputs for the neuron are values of the neurons in the previous layer. Input of the neuron is passed as the vector of double type variables. In the same way inputs of the whole neural network are passed. Weights are also stored as a vector. Storing both of those properties as vectors let the program easily calculate a dot product of them.

3 External specification

This is a command line program.You can execute a program by using compiled .exe or .o file or by using make in the e.g. Bash terminal. The program requires input datasets specified in the input file.

After execution you can find output file in the following direction:

```
/logs/output.csv
```

You have to ensure yourself '/logs' directory exists, otherwise output file won't be created.

4 Internal specification

The program is implemented with object-oriented, structural and functional paradigm. User interface is separated from program's logic by using relevant classes.

4.1 Program overview

The `main` function creates objects of several classes which, every of them has its own responsibility designated. In the begining we prepare an output file using an instance of CSV_Logger class. Then it creates instance of NeuralNetwork class and inputLoader. Function saves datased loaded from the loader and pass readed inputs and labels as arguments of the NeuralNetwork train method. At the end program informs about termination of the whole process.

4.2 Description of types and functions

Description of types and functions is moved to the appendix.

5 Testing

The program has been tested with various types of files. Incorrect files (with no numbers, numbers in incorrect format, strings with some invalid whitespaces, ...) are detected and an error message is printed. An empty input file does not cause failure – an empty output file is created. Maximal number value(**double**) in an input file is approximately 1.8e+308. Maximal input file size handled by the program is 1.57 GB. Larger files result in a bad allocation error. The program has no memory leaks.

6 Conclusions

The program implements a neural network. The most demanding task was to create a structure of the Neural Network and to preserve dataflow simple using features of objective and functional programming paradigm.

References

Appendix
Description of types and functions

knapsackProblem

0.1

Generated by Doxygen 1.9.1

Chapter 1

Neural Network

1.1 Olivier Halupczok

1.1.1 To build and execute the whole main, and compile all necesary libraries, you can simply execute make command in the terminal.

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

File Index

4.1 File List

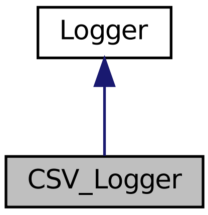
Here is a list of all documented files with brief descriptions:

Chapter 5

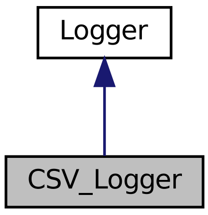
Class Documentation

5.1 CSV_Logger Class Reference

Inheritance diagram for CSV_Logger:



Collaboration diagram for CSV_Logger:



Public Member Functions

- CSV_Logger (std::string path)

    Construct a new csv logger object.
- ∼CSV_Logger ()

    Destroy the csv logger objectand close opened file.
- void setLabels (std::vector< std::string > &labels)

    prints labels into output csv file
- void setSeparator (const char separator)

    Set the Separator property.

5.1.1 Constructor & Destructor Documentation

5.1.1.1 CSV_Logger()

```
CSV_Logger::CSV_Logger (
             std::string path )
```

Construct a new csv logger object.

Parameters

path	to the file

5.1.2 Member Function Documentation

5.1.2.1 setLabels()

```
void CSV_Logger::setLabels (
             std::vector< std::string > & labels )
```

prints labels into output csv file

Parameters

labels	

5.1.2.2 setSeparator()

```
void CSV_Logger::setSeparator (
            const char separator )
```

Set the Separator property.

Parameters

separator	

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

- logger.h
- logger.cpp

5.2 Dataset Class Reference

Public Member Functions

- Dataset (size_t _dataFeatures)

  Construct a new Dataset object.
- void addDataRow (std::string label, std::vector< double > &_inputs)

  add data to inputs and labels vectors and insert label to the set if it wasn't done beforehand
- std::vector< std::vector< double > > getInputs ()

  Get the Inputs object.
- std::vector< std::string > getLabels ()

  Get the Labels vector.
- std::set< std::string > getSetOfLabels ()

  Get the Set Of Labels.
- void setHeaders (std::vector< std::string > &headers)

  Set the Headers.
- std::vector< std::string > getHeaders ()

  Get the Headers vector.
- std::vector< double > getLabelsEncoded ()

  Get the Labels Encoded object.

5.2.1 Constructor & Destructor Documentation

5.2.1.1 Dataset()

```
Dataset::Dataset (
            size_t _dataFeatures )
```

Construct a new Dataset object.

Parameters

_dataFeatures	

5.2.2 Member Function Documentation

5.2.2.1 addDataRow()

```
void Dataset::addDataRow (
             std::string label,
             std::vector< double > & _inputs )
```

add data to inputs and labels vectors and insert label to the set if it wasn't done beforehand

Parameters

label	
_inputs	

5.2.2.2 getHeaders()

```
std::vector< std::string > Dataset::getHeaders ( )
```

Get the Headers vector.

Returns

std::vector<std::string>

5.2.2.3 getInputs()

```
std::vector< std::vector< double > > Dataset::getInputs ( )
```

Get the Inputs object.

Returns

std::vector<std::vector<double>>

5.2.2.4 getLabels()

```
std::vector< std::string > Dataset::getLabels ( )
```

Get the Labels vector.

Returns

std::vector<std::string>

5.2.2.5 getLabelsEncoded()

```
std::vector< double > Dataset::getLabelsEncoded ( )
```

Get the Labels Encoded object.

Returns

std::vector<double> codes of the label for every entity presented in the input file

5.2.2.6 getSetOfLabels()

```
std::set< std::string > Dataset::getSetOfLabels ( )
```

Get the Set Of Labels.

Returns

std::set<std::string> set with unique labels

5.2.2.7 setHeaders()

```
void Dataset::setHeaders (
            std::vector< std::string > & headers )
```

Set the Headers.

Parameters

headers	from the CSV file

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

- dataset.h
- dataset.cpp

5.3 InputLoader Class Reference

Public Member Functions

- InputLoader (std::string _inputFilePath)

  Construct a new Input Loader object.
- ∼InputLoader ()

  Destroy the Input Loader object and its pointers.
- Dataset getData ()

  Get the Data object.

5.3.1 Constructor & Destructor Documentation

5.3.1.1 InputLoader()

```
InputLoader::InputLoader (
            std::string _inputFilePath )
```

Construct a new Input Loader object.

Parameters

_inputFilePath	path to the input file

5.3.2 Member Function Documentation

5.3.2.1 getData()

```
Dataset InputLoader::getData ( )
```

Get the Data object.

Returns

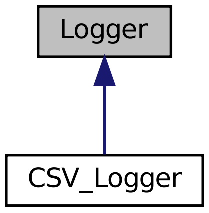
Dataset

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

- inputLoader.h
- inputLoader.cpp

5.4 Logger Class Reference

Inheritance diagram for Logger:



Public Member Functions

- Logger ()

  Construct a new Logger object.
- void setOutputStream (std::ostream &stream)

  Set the Output Stream property.
- void isOutputSet ()

  check if output is set, if not then throw an exception
- template<class T >
  Logger & operator<< (T &&dataToLog)

  Log data with << operator.
- Logger & operator<< (std::ostream &(∗manip)(std::ostream &))

  operator with manip definition to let Logger handle io manipulators

5.4.1 Member Function Documentation

5.4.1.1 operator<<() [1/2]

```
Logger & Logger::operator<< (
            std::ostream &(∗)(std::ostream &) manip )
```

operator with manip definition to let Logger handle io manipulators

Parameters

manip	io manipulators like 'std::endl'

Returns

Logger& It returns instance of currently using object

5.4.1.2 operator<<() [2/2]

```
template<class T >
Logger & Logger::operator<< (
             T && dataToLog )
```

Log data with << operator.

Template Parameters

T	template to let data of many types to be logged

Parameters

dataToLog	Data to be printed with the logger

Returns

Logger& It returns the whole instance of the object

5.4.1.3 setOutputStream()

```
void Logger::setOutputStream (
             std::ostream & stream )
```

Set the Output Stream property.

Parameters

stream	which the logs will be forwarded to

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

- logger.h
- logger.cpp

5.5 NetworkResult Class Reference

Public Member Functions

- NetworkResult (double _loss, std::vector< double > &_predictions)

Construct a new Network Result object.

- double getLoss ()

  Get the Loss property.
- std::vector< double > getPredictions ()

  Get the Predictions property.

Friends

- std::ostream & operator<< (std::ostream &stream, NetworkResult &result)

  print loss to the stream

5.5.1 Constructor & Destructor Documentation

5.5.1.1 NetworkResult()

```
NetworkResult::NetworkResult (
            double _loss,
            std::vector< double > & _predictions )
```

Construct a new Network Result object.

Parameters

_loss	loss of the given epoch
_predictions	predictions of the given epoch

5.5.2 Member Function Documentation

5.5.2.1 getLoss()

```
double NetworkResult::getLoss ( )
```

Get the Loss property.

Returns

double

5.5.2.2 getPredictions()

```
std::vector< double > NetworkResult::getPredictions ( )
```

Get the Predictions property.

Returns

std::vector<double>

5.5.3 Friends And Related Function Documentation

5.5.3.1 operator<<

```
std::ostream& operator<< (
             std::ostream & stream,
             NetworkResult & result )  [friend]
```

print loss to the stream

Parameters

stream	stream which the data is printed into
result	result of the given epoch

Returns

std::ostream&

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

- networkResult.h
- networkResult.cpp

5.6 NeuralNetwork Class Reference

Public Member Functions

- NeuralNetwork (size_t numOfNeurons, std::function< double(double)> activationFunc, std::function< double(double)> activationFuncDeriv, double learningRate)

    Construct a new Neural Network object.
- double feedforward (std::vector< double > inputs)

    feedforward every neuron from hidden layer
- void train (long long int epochs, std::vector< std::vector< double >> inputData, std::vector< double > labels, std::function< void(NetworkResult)> callback)

    train neural net

5.6.1 Constructor & Destructor Documentation

5.6.1.1 NeuralNetwork()

```
NeuralNetwork::NeuralNetwork (
            size_t numOfNeurons,
            std::function< double(double)> activationFunc,
            std::function< double(double)> activationFuncDeriv,
            double learningRate )
```

Construct a new Neural Network object.

random number engine

Parameters

numOfNeurons	num of Neurons to create
activationFunc	function to activate neuron
activationFuncDeriv	derivative of activation function
learningRate	learning rate of the neurons

5.6.2 Member Function Documentation

5.6.2.1 feedforward()

```
double NeuralNetwork::feedforward (
            std::vector< double > inputs )
```

feedforward every neuron from hidden layer

Parameters

inputs	from input layer

Returns

double output of the output neuron

5.6.2.2 train()

```
void NeuralNetwork::train (
            long long int epochs,
            std::vector< std::vector< double >> inputData,
            std::vector< double > labels,
            std::function< void(NetworkResult)> callback )
```

train neural net

Parameters

epochs	determines the number of iterations through the whole dataset
inputData	dataset to train on
labels	evualuate training process
callback	callback after training

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

- neuralNetwork.h
- neuralNetwork.cpp

5.7 Neuron Class Reference

Public Member Functions

- Neuron (std::vector< double > weightsToInit, double biasToInit, std::function< double(double)> activation↵ FuncToInit, std::function< double(double)> activationFuncDeriv, double learningRateToInit)

  Construct a new Neuron object.
- double feedforward (std::vector< double > inputsToFeed)

  Feedforward with neurons from previous layers.
- double getTotal ()

  Get the Total property.
- std::vector< double > getWeights ()

  Get the Weights property.
- double getBias ()

  Get the Bias property.
- void adjustWeight (size_t index, Neuron outputNeuron, double lossDeriv_outDeriv_calced, size_t iterator)

  use backprop to adjust weight of specified index and to train network by doing so
- void adjustBias (Neuron outputNeuron, double lossDeriv_outDeriv_calced, size_t iterator)

  use backprop to adjust bias and to train network
- double getOutput ()

  Get the Output calculated during feedforward's execution.

5.7.1 Constructor & Destructor Documentation

5.7.1.1 Neuron()

```
Neuron::Neuron (
            std::vector< double > weightsToInit,
            double biasToInit,
            std::function< double(double)> activationFuncToInit,
            std::function< double(double)> activationFuncDeriv,
            double learningRateToInit = LEARNING_DEFAULT_RATE )
```

Construct a new Neuron object.

Parameters

weightsToInit	Weigths of inputs
biasToInit	Bias to calculate feedforward's total
activationFuncToInit	

5.7.2 Member Function Documentation

5.7.2.1 adjustBias()

```
void Neuron::adjustBias (
            Neuron outputNeuron,
            double lossDeriv_outDeriv_calced,
            size_t iterator )
```

use backprop to adjust bias and to train network

Parameters

outputNeuron	
lossDeriv_outDeriv	
iterator	

5.7.2.2 adjustWeight()

```
void Neuron::adjustWeight (
            size_t index,
            Neuron outputNeuron,
            double lossDeriv_outDeriv_calced,
            size_t iterator )
```

use backprop to adjust weight of specified index and to train network by doing so

Parameters

index	
outputNeuron	
lossDeriv_outDeriv	
iterator	

5.7.2.3 feedforward()

```
double Neuron::feedforward (
            std::vector< double > inputsToFeed )
```

Feedforward with neurons from previous layers.

Parameters

inputs	values of previous neurons

Returns

double total value of neuron

5.7.2.4 getBias()

```
double Neuron::getBias ( )
```

Get the Bias property.

Returns

double bias of the neuron

5.7.2.5 getOutput()

```
double Neuron::getOutput ( )
```

Get the Output calculated during feedforward's execution.

Returns

double - output value

5.7.2.6 getTotal()

```
double Neuron::getTotal ( )
```

Get the Total property.

Returns

double - sum of: dot product of inputs and weights, and bias

5.7.2.7 getWeights()

```
std::vector< double > Neuron::getWeights ( )
```

Get the Weights property.

Returns

std::vector<double> of weights

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

- neuron.h
- neuron.cpp

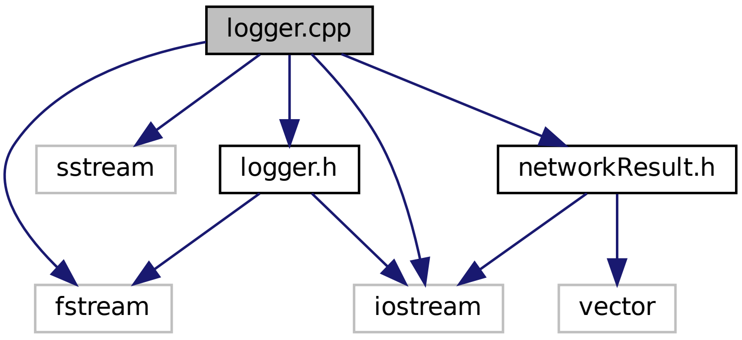
Chapter 6

File Documentation

6.1 logger.cpp File Reference

cpp file with the definitions of the logger library

```
#include <iostream>
#include <sstream>
#include <fstream>
#include "networkResult.h"
#include "logger.h"
```
Include dependency graph for logger.cpp:



Functions

- template Logger & **Logger::operator**<<< **std::string** > (std::string &&dataToLog)

6.1.1 Detailed Description

cpp file with the definitions of the logger library

Author

Olivier Halupczok

Version

0.1

Date

2022-06-18

Copyright

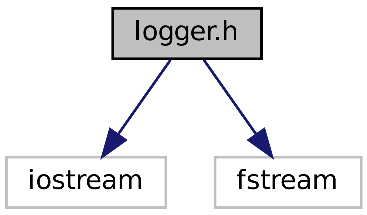
Copyright (c) 2022

6.2 logger.h File Reference

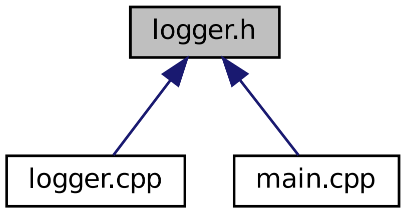
declarations of the logger library

```
#include <iostream>
#include <fstream>
```

Include dependency graph for logger.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Logger
- class CSV_Logger

6.2.1 Detailed Description

declarations of the logger library

Author

Olivier Halupczok

Version

0.1

Date

2022-06-18

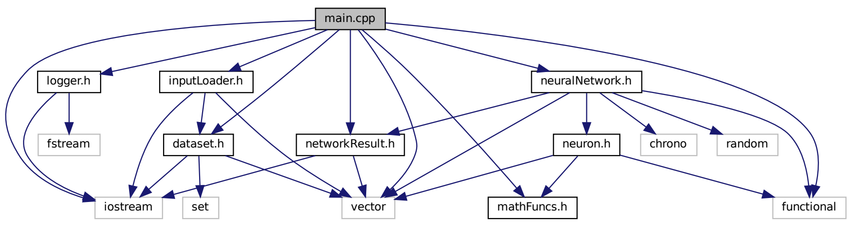
Copyright

Copyright (c) 2022

6.3 main.cpp File Reference

```
#include <iostream>
#include <vector>
#include <functional>
#include "inputLoader.h"
#include "mathFuncs.h"
#include "networkResult.h"
#include "neuralNetwork.h"
#include "logger.h"
#include "dataset.h"
```

Include dependency graph for main.cpp:



Functions

- void **logResults** (NetworkResult res)
- int **main** (int argc, char const ∗argv[ ])

Variables

- size_t **epochCount** = 1
- CSV_Logger **logger** ("logs/output.csv")

6.3.1 Detailed Description

Author

Olivier Halupczok

Version

0.1

Date

2022-06-11

Copyright

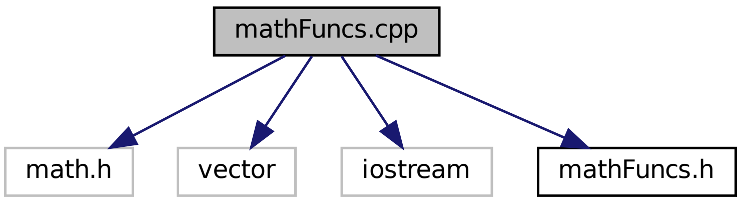
Copyright (c) 2022

6.4 mathFuncs.cpp File Reference

Library with math functions used in the program.

```
#include <math.h>
#include <vector>
#include <iostream>
#include "mathFuncs.h"
```

Include dependency graph for mathFuncs.cpp:



Functions

- std::string **invArgVectorsMsg** (std::string nameOfFunc)
- double sigmoid (double arg)

  return value of sigmoid function (1/(1 + exp(-arg)))
- double deriv_sigmoid (double arg)

  return value of derivative of sigmoid
- double dotProductOf2Vectors (std::vector< double > vector1, std::vector< double > vector2)

  return dot product of two two components vectors
- double calc_mse_loss (std::vector< double > outputTrue, std::vector< double > outputPredicted)

  calculate mean squarred error of the neural network
- double lossDeriv_outDeriv (double label, double output)

  it calculates value of the derivative of the Loss' function of the whole network divided by the derivative of output value's(of the entire network) function

Variables

- const std::string **INV_ARG_MSG** = "Invalid argument: "
- const std::string **INV_VECTORS_MSG** = " has to be executed with 2 vectors of the same length"
- const std::string **DOT_PRODUCT** = "dot product"
- const std::string **MSE** = "MSE loss calculation"

6.4.1 Detailed Description

Library with math functions used in the program.

Author

Olivier Halupczok

Version

0.1

Date

2022-06-18

Copyright

Copyright (c) 2022

6.4.2 Function Documentation

6.4.2.1 calc_mse_loss()

```
double calc_mse_loss (
            std::vector< double > outputTrue,
            std::vector< double > outputPredicted )
```

calculate mean squarred error of the neural network

Parameters

outputTrue	labels of dataargumentsHandler
outputPredicted	guesses of network

Returns

double return mean squarred error

6.4.2.2 deriv_sigmoid()

```
double deriv_sigmoid (
            double arg )
```

return value of derivative of sigmoid

Parameters

arg	

Returns

double

6.4.2.3 dotProductOf2Vectors()

```
double dotProductOf2Vectors (
            std::vector< double > vector1,
            std::vector< double > vector2 )
```

return dot product of two two components vectors

Parameters

vector1	
vector2	

Returns

double dot product

6.4.2.4 lossDeriv_outDeriv()

```
double lossDeriv_outDeriv (
             double label,
             double output )
```

it calculates value of the derivative of the Loss' function of the whole network divided by the derivative of output value's(of the entire network) function

Parameters

label	labels of the learning dataset
output	output value of net

Returns

double calculated derivative

6.4.2.5 sigmoid()

```
double sigmoid (
             double arg )
```

return value of sigmoid function (1/(1 + exp(-arg)))

Parameters

arg	

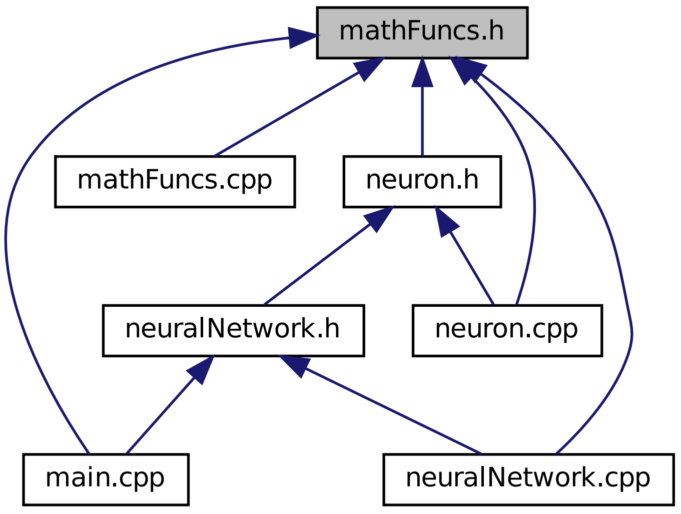
Returns

double

6.5 mathFuncs.h File Reference

library with math functions used in the program

This graph shows which files directly or indirectly include this file:



Functions

- double sigmoid (double arg)

  return value of sigmoid function (1/(1 + exp(-arg)))
- double deriv_sigmoid (double arg)

  return value of derivative of sigmoid
- double dotProductOf2Vectors (std::vector< double > vector1, std::vector< double > vector2)

  return dot product of two two components vectors
- double calc_mse_loss (std::vector< double > outputTrue, std::vector< double > outputPredicted)

  calculate mean squarred error of the neural network
- double lossDeriv_outDeriv (double label, double output)

  it calculates value of the derivative of the Loss' function of the whole network divided by the derivative of output value's(of the entire network) function

6.5.1 Detailed Description

library with math functions used in the program

Author

Olivier Halupczok

Version

0.1

Date

2022-06-12

Copyright

Copyright (c) 2022

6.5.2 Function Documentation

6.5.2.1 calc_mse_loss()

```
double calc_mse_loss (
            std::vector< double > outputTrue,
            std::vector< double > outputPredicted )
```

calculate mean squared error of the neural network

Parameters

outputTrue	labels of dataargumentsHandler
outputPredicted	guesses of network

Returns

double return mean squarred error

6.5.2.2 deriv_sigmoid()

```
double deriv_sigmoid (
            double arg )
```

return value of derivative of sigmoid

Parameters

arg	

Returns

double

6.5.2.3 dotProductOf2Vectors()

```
double dotProductOf2Vectors (
            std::vector< double > vector1,
            std::vector< double > vector2 )
```

return dot product of two two components vectors

Parameters

vector1	
vector2	

Returns

double dot product

6.5.2.4 lossDeriv_outDeriv()

```
double lossDeriv_outDeriv (
            double label,
            double output )
```

it calculates value of the derivative of the Loss' function of the whole network divided by the derivative of output value's(of the entire network) function

Parameters

label	labels of the learning dataset
output	output value of net

Returns

double calculated derivative

6.5.2.5 sigmoid()

```
double sigmoid (
            double arg )
```

return value of sigmoid function (1/(1 + exp(-arg)))

Parameters

arg	

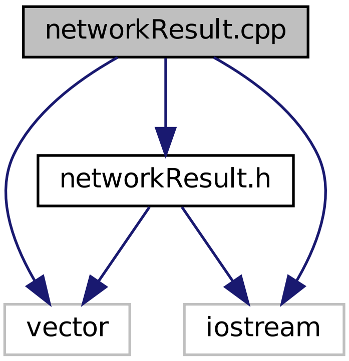
Returns

double

6.6 networkResult.cpp File Reference

```
#include <vector>
#include <iostream>
#include "networkResult.h"
```

Include dependency graph for networkResult.cpp:



Functions

- std::ostream & operator<< (std::ostream &stream, NetworkResult &result)

6.6.1 Detailed Description

Author

Olivier Halupczok

Version

0.1

Date

2022-06-18

Copyright

Copyright (c) 2022

6.6.2 Function Documentation

6.6.2.1 operator<<()

```
std::ostream& operator<< (
             std::ostream & stream,
             NetworkResult & result )
```

Parameters

stream	stream which the data is printed into
result	result of the given epoch

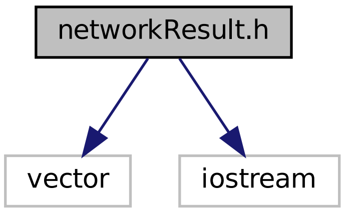
Returns

std::ostream&

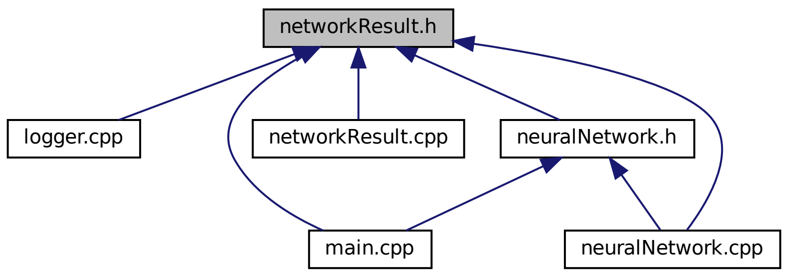
6.7 networkResult.h File Reference

```
#include <vector>
#include <iostream>
```

Include dependency graph for networkResult.h:



This graph shows which files directly or indirectly include this file:



Classes

- class NetworkResult

6.7.1 Detailed Description

Author

Olivier Halupczok

Version

0.1

Date

2022-06-18

Copyright

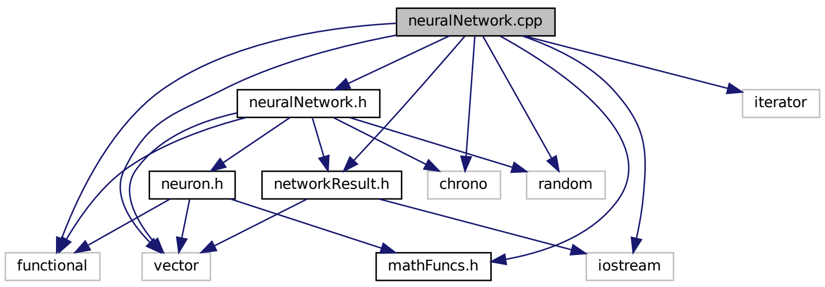
Copyright (c) 2022

6.8 neuralNetwork.cpp File Reference

```
#include <functional>
#include <vector>
#include <iterator>
#include <iostream>
#include <chrono>
#include <random>
#include "mathFuncs.h"
#include "networkResult.h"
```

```
#include "neuralNetwork.h"
```

Include dependency graph for neuralNetwork.cpp:



6.8.1 Detailed Description

Author

Olivier Halupczok

Version

0.1

Date

2022-06-13

Copyright

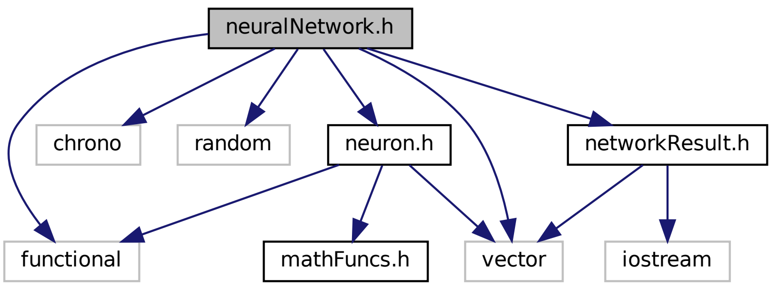
Copyright (c) 2022

6.9 neuralNetwork.h File Reference

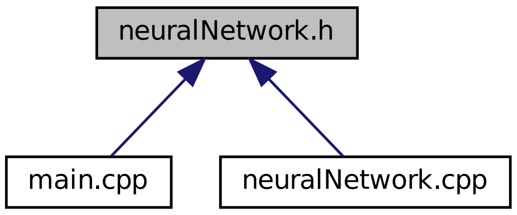
```
#include <vector>
#include <chrono>
#include <random>
#include <functional>
#include "neuron.h"
```

```
#include "networkResult.h"
```

Include dependency graph for neuralNetwork.h:



This graph shows which files directly or indirectly include this file:



Classes

- class NeuralNetwork

6.9.1 Detailed Description

Author

Olivier Halupczok

Version

0.1

Date

2022-06-13

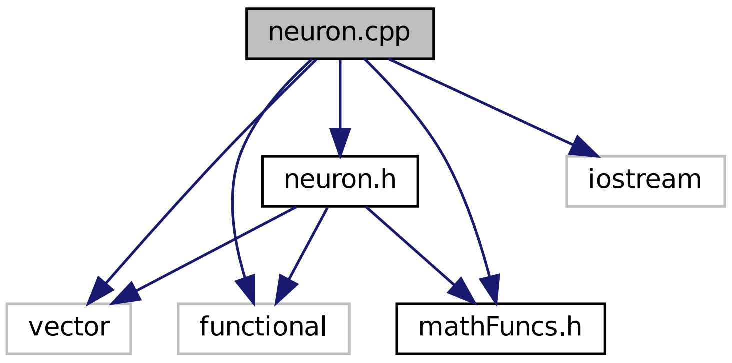
Copyright

Copyright (c) 2022

6.10 neuron.cpp File Reference

```
#include <vector>
#include <functional>
#include <iostream>
#include "mathFuncs.h"
#include "neuron.h"
```

Include dependency graph for neuron.cpp:



Functions

- std::string **exceptionMsg** (std::string propertyName)

Variables

- constexpr double **LEARNING_DEFAULT_RATE** = 0.01
- const std::string **OUTPUT** = "output"
- const std::string **TOTAL** = "total"

6.10.1 Detailed Description

Author

Olivier Halupczok

Version

0.1

Date

2022-06-12

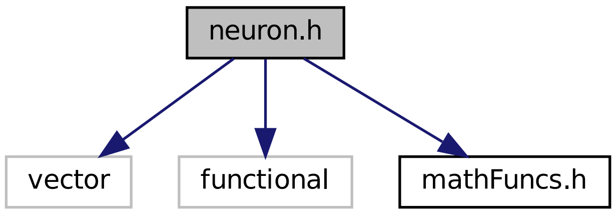
Copyright

Copyright (c) 2022

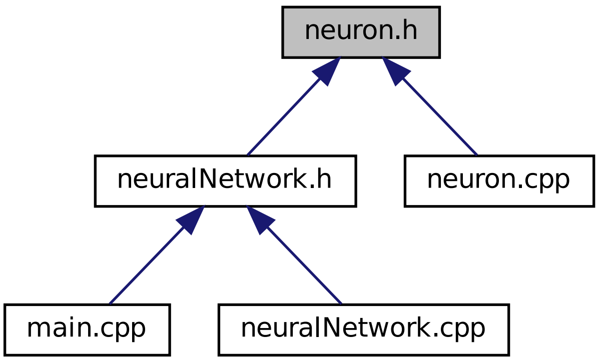
6.11 neuron.h File Reference

```
#include <vector>
#include <functional>
#include "mathFuncs.h"
```

Include dependency graph for neuron.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Neuron

6.11.1 Detailed Description

Author

Olivier Halupczok

Version

0.1

Date

2022-06-12

Copyright

Copyright (c) 2022

Index