ML perceptron (P1)

v0.1

Generated by Doxygen 1.13.2

Chapter 1

ML-perceptron

1.1 Student

Name: Stan Merlijn

Student nummer: 1863967

1.2 Introduction

In this repo we are going to implement and test perceptrons, perceptron layers and a perceptron networks(neural network). Theset are going to be tested by creating AND, OR, INVERT, NAND, XOR and half adder logic gates. the reader can be found here

1.3 Installing

Enter the test dir then

Generate build files:

```
cmake -S . -B build
```

Build the project:

```
cmake --build build
```

Run the executable:

```
./build/MLPerceptronTest
```

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

Class Documentation

4.1 halfAdder Class Reference

Public Member Functions

- halfAdderOutput **predict** (const std::vector< int > &x) const
- void **__str__** (int verbose) const

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

- /Users/stanislav/Github/MachineLearning/ML-Perceptron/src/header/halfAdder.hpp

4.2 halfAdderOutput Struct Reference

Public Attributes

- int **sum**
- int **carry**

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

- /Users/stanislav/Github/MachineLearning/ML-Perceptron/src/header/halfAdder.hpp

4.3 Perceptron Class Reference

A simple perceptron model for binary classification.

```
#include <perceptron.hpp>
```

Public Member Functions

- Perceptron (std::vector< double > weights, double bias, double learningRate)

  Constructs a Perceptron with given weights, bias, and learning rate.
- int predict (const std::vector< int > &inputs) const

  Predicts the output for a given input vector.
- void train (const std::vector< std::vector< int > > &inputs, const std::vector< int > &targets, int epochs)

  Trains the perceptron using the given dataset. Using th learning rule to update the weights.
- void __str__ (int verbose) const

  Prints perceptron details.

4.3.1 Detailed Description

A simple perceptron model for binary classification.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Perceptron()

```
Perceptron::Perceptron (
             std::vector< double > weights,
             double bias,
             double learningRate)
```

Constructs a Perceptron with given weights, bias, and learning rate.

Parameters

weights	Initial weights.
bias	Initial bias.
learningRate	Learning rate for training.

4.3.3 Member Function Documentation

4.3.3.1 __str__()

```
void Perceptron::__str__ (
             int verbose) const
```

Prints perceptron details.

Parameters

verbose	Verbosity level.

4.3.3.2 predict()

```
int Perceptron::predict (
             const std::vector< int > & inputs) const
```

Predicts the output for a given input vector.

Parameters

inputs	Input vector.

Returns

1 if activated, otherwise 0.

4.3.3.3 train()

```
void Perceptron::train (
             const std::vector< std::vector< int > > & inputs,
             const std::vector< int > & targets,
             int epochs)
```

Trains the perceptron using the given dataset. Using th learning rule to update the weights.

Parameters

inputs	Input samples.
targets	Target outputs.
epochs	Number of training iterations.

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

- /Users/stanislav/Github/MachineLearning/ML-Perceptron/src/header/perceptron.hpp
- /Users/stanislav/Github/MachineLearning/ML-Perceptron/src/perceptron.cpp

4.4 PerceptronLayer Class Reference

Represents a layer of perceptrons in a neural network.

```
#include <perceptronLayer.hpp>
```

Public Member Functions

- PerceptronLayer (const std::vector< Perceptron > &neurons)

  Constructs a perceptron layer.
- std::vector< int > feedForward (const std::vector< int > &input) const

  Feeds input forward through the layer.
- void __str__ (int verbose) const

  Prints layer details.

4.4.1 Detailed Description

Represents a layer of perceptrons in a neural network.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 PerceptronLayer()

```
PerceptronLayer::PerceptronLayer (
             const std::vector< Perceptron > & neurons)
```

Constructs a perceptron layer.

Parameters

neurons	List of perceptrons.

4.4.3 Member Function Documentation

4.4.3.1 __str__()

```
void PerceptronLayer::__str__ (
             int verbose) const
```

Prints layer details.

Parameters

verbose	Verbosity level.

4.4.3.2 feedForward()

```
std::vector< int > PerceptronLayer::feedForward (
             const std::vector< int > & input) const
```

Feeds input forward through the layer.

Parameters

input	Input vector.

Returns

Output vector after applying all perceptrons.

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

- /Users/stanislav/Github/MachineLearning/ML-Perceptron/src/header/perceptronLayer.hpp
- /Users/stanislav/Github/MachineLearning/ML-Perceptron/src/perceptronLayer.cpp

4.5 PerceptronNetwork Class Reference

Represents a multi-layer perceptron network.

```
#include <perceptronNetwork.hpp>
```

Public Member Functions

- PerceptronNetwork (std::vector< PerceptronLayer > layers)

  Constructs a perceptron network.
- std::vector< int > feedForward (const std::vector< int > &input) const

  Feeds input forward through the network.
- void __str__ (int verbose) const

  Prints network details.

4.5.1 Detailed Description

Represents a multi-layer perceptron network.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 PerceptronNetwork()

```
PerceptronNetwork::PerceptronNetwork (
             std::vector< PerceptronLayer > layers)
```

Constructs a perceptron network.

Parameters

layers	List of perceptron layers.

4.5.3 Member Function Documentation

4.5.3.1 __str__()

```
void PerceptronNetwork::__str__ (
             int verbose) const
```

Prints network details.

Parameters

verbose	Verbosity level.

4.5.3.2 feedForward()

```
std::vector< int > PerceptronNetwork::feedForward (
             const std::vector< int > & input) const
```

Feeds input forward through the network.

Parameters

input	Input vector.

Returns

Output vector after processing through all layers.

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

- /Users/stanislav/Github/MachineLearning/ML-Perceptron/src/header/perceptronNetwork.hpp
- /Users/stanislav/Github/MachineLearning/ML-Perceptron/src/perceptronNetwork.cpp

Chapter 5

File Documentation

5.1 halfAdder.hpp

```cpp
#include "perceptron.hpp"
#include <iostream>
#include <vector>

#define EPOCHS 100

struct halfAdderOutput
{
    int sum;
    int carry;
};

class halfAdder
{
private:
    // Perceptrons for the half adder
    Perceptron andGate;
    Perceptron orGate;
    Perceptron nanGate;

public:
    halfAdder()
        : andGate({0.1, 0.1}, 1, 0.1),
        orGate({0.1, 0.1}, 1, 0.1),
        nanGate({0.1, 0.1}, 1, 0.1)
    {
        // Training data for the half adder
        std::vector<std::vector<int» inputs = {{0, 0}, {0, 1}, {1, 0}, {1, 1}};
        std::vector<int> targetsAnd  = {0, 0, 0, 1};
        std::vector<int> targetsOr   = {0, 1, 1, 1};
        std::vector<int> targetsNand = {1, 1, 1, 0};

        // Train the perceptrons
        andGate.train(inputs, targetsAnd, EPOCHS);
        orGate.train(inputs, targetsOr, EPOCHS);
        nanGate.train(inputs, targetsNand, EPOCHS);
}


    halfAdderOutput predict(const std::vector<int>& x) const
    {
        halfAdderOutput output;
        output.sum = andGate.predict({orGate.predict(x), nanGate.predict(x)});
        output.carry = andGate.predict(x);
        return output;
    }

    void __str__(int verbose) const
    {
        std::cout « "Half Adder Structure:" « std::endl;
        std::cout « "AND Gate: ";
        andGate.__str__(verbose);
        std::cout « "OR Gate: ";
        orGate.__str__(verbose);
        std::cout « "NAND Gate: ";
        nanGate.__str__(verbose);
    }

};
```

5.2 /Users/stanislav/Github/MachineLearning/ML-↩ Perceptron/src/header/perceptron.hpp File Reference

In this file the Perceptron class is defined.

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

Classes

- class Perceptron

  A simple perceptron model for binary classification.

5.2.1 Detailed Description

In this file the Perceptron class is defined.

Author

Stan Merlijn

Version

0.1

Date

2025-02-12

Copyright

Copyright (c) 2025

5.3 perceptron.hpp

Go to the documentation of this file.

```


#pragma once
#include <iostream>
#include <vector>

class Perceptron
{
public:
    Perceptron(std::vector<double> weights, double bias, double learningRate);

    int predict(const std::vector<int>& inputs) const;

    void train(const std::vector<std::vector<int»& inputs, const std::vector<int>& targets, int epochs);

    void __str__(int verbose) const;

private:
        std::vector<double> weights;
        double bias;
        double learningRate;
};
```

5.4 /Users/stanislav/Github/MachineLearning/ML-Perceptron/src/header/perceptronLayer.hpp File Reference

In this file the PerceptronLayer class is defined.

```
#include "perceptron.hpp"
#include <iostream>
#include <vector>
```

Classes

- class PerceptronLayer

  Represents a layer of perceptrons in a neural network.

5.4.1 Detailed Description

In this file the PerceptronLayer class is defined.

Author

Stan Merlijn

Version

0.1

Date

2025-02-12

Copyright

Copyright (c) 2025

5.5 perceptronLayer.hpp

Go to the documentation of this file.

```

#pragma once
#include "perceptron.hpp"
#include <iostream>
#include <vector>

class PerceptronLayer
{
public:
    PerceptronLayer(const std::vector<Perceptron>& neurons);

    std::vector<int> feedForward(const std::vector<int>& input) const;

    void __str__(int verbose) const;

private:
    std::vector<Perceptron> neurons;
};
```

5.6 /Users/stanislav/Github/MachineLearning/ML-Perceptron/src/header/perceptronNetwork.hpp File Reference

In this file the PerceptronNetwork class is defined.

```
#include "perceptronLayer.hpp"
#include <iostream>
#include <vector>
```

Classes

- class PerceptronNetwork

  Represents a multi-layer perceptron network.

5.6.1 Detailed Description

In this file the PerceptronNetwork class is defined.

Author

Stan Merlijn

Version

0.1

Date

2025-02-12

Copyright

Copyright (c) 2025

5.7 perceptronNetwork.hpp

Go to the documentation of this file.

```

#pragma once
#include "perceptronLayer.hpp"
#include <iostream>
#include <vector>

class PerceptronNetwork
{
public:
    PerceptronNetwork(std::vector<PerceptronLayer> layers);

    std::vector<int> feedForward(const std::vector<int>& input) const;

    void __str__(int verbose) const;

private:
    std::vector<PerceptronLayer> layers;

};
```

5.8 /Users/stanislav/Github/MachineLearning/ML-↩ Perceptron/src/test/test.cpp File Reference

In this file the test cases for the Perceptron, PerceptronLayer and PerceptronNetwork classes are defined.

```
#include "catch.hpp"
#include "../header/perceptron.hpp"
#include "../header/perceptronLayer.hpp"
#include "../header/perceptronNetwork.hpp"
#include "../header/halfAdder.hpp"
#include <iostream>
```

Macros

- #define **CATCH_CONFIG_MAIN**
- #define **EPOCHS** 100

Functions

- **TEST_CASE** ("Perceptron for INVERT Gate", "[perceptron]")

  Perceptron for INVERT Gate: Tests the perceptron's ability to learn the INVERT gate.
- **TEST_CASE** ("Perceptron for AND Gate", "[perceptron]")

  Perceptron for AND Gate: Tests the perceptron's ability to learn the AND gate.
- **TEST_CASE** ("Perceptron for OR Gate", "[perceptron]")

  Perceptron for OR Gate: Tests the perceptron's ability to learn the OR gate.
- TEST_CASE ("Perceptron for NOR Gate (3 inputs)", "[perceptron]")

  Perceptron for NOR Gate (3 inputs): Tests the perceptron's ability to learn the NOR gate with 3 inputs. The NOR gate is a digital logic gate that implements logical NOR
- **TEST_CASE** ("Perceptron for 3-input Majority Gate", "[perceptron]")

  Perceptron for 3-input Majority Gate: Tests the perceptron's ability to learn the 3-input Majority gate.
- **TEST_CASE** ("PerceptronLayer for AND and OR Gates", "[perceptronLayer]")

  PerceptronLayer for AND and OR Gates: Tests the PerceptronLayer's ability to learn the AND and OR gates. It contains two perceptrons: one for the AND gate and one for the OR gate.
- **TEST_CASE** ("PerceptronNetwork for the XOR gate with 2 inputs", "[perceptronNetwork]")

  PerceptronNetwork for the XOR gate with 2 inputs. This network contains two layers: inputLayer for the AND gate and one for the OR gate. outputLayer for the AND gate.
- **TEST_CASE** ("PerceptronNetwork for half adder", "[perceptronNetwork]")

  PerceptronNetwork for a half adder. This network contains two layers: hiddenLayer for the OR and AND gates. outputLayer for the XOR gate(sum) and the carry.

Variables

- std::vector< std::vector< int > > **inputs** = {{0, 0}, {0, 1}, {1, 0}, {1, 1}}

5.8.1 Detailed Description

In this file the test cases for the Perceptron, PerceptronLayer and PerceptronNetwork classes are defined.

Unit tests for the Perceptron, PerceptronLayer and PerceptronNetwork classes.

Author

Stan Merlijn

Version

0.1

Date

2025-02-12

Copyright

Copyright (c) 2025

This file contains a series of test cases to verify the functionality of the Perceptron and PerceptronLayer classes. The tests include training and prediction for various logic gates.

Test Cases:

- Perceptron for INVERT Gate: Tests the perceptron's ability to learn the INVERT gate.
- Perceptron for AND Gate: Tests the perceptron's ability to learn the AND gate.
- Perceptron for OR Gate: Tests the perceptron's ability to learn the OR gate.
- Perceptron for NOR Gate (3 inputs): Tests the perceptron's ability to learn the NOR gate with 3 inputs.
- Perceptron for 3-input Majority Gate: Tests the perceptron's ability to learn the 3-input Majority gate.
- PerceptronLayer for AND and OR Gates: Tests the PerceptronLayer's ability to learn the AND and OR gates.
- PerceptronNetwork for the XOR gate with 2 inputs.
- PerceptronNetwork for a half adder.

Note

The tests use the Catch2 framework for unit testing.

5.8.2 Function Documentation

5.8.2.1 TEST_CASE()

```
TEST_CASE (
             "Perceptron for NOR Gate (3 inputs)" ,
             "" [perceptron])
```

Perceptron for NOR Gate (3 inputs): Tests the perceptron's ability to learn the NOR gate with 3 inputs. The NOR gate is a digital logic gate that implements logical NOR

0, 0, 0, 0, 0, 0, 0, 1

Index