UTM

UNIVERSITI TEKNOLOGI MALAYSIA

Computer Intelligence

Assignment 1

Name	**Matric No**
Adam Iskandar Bin Sharudin	A21MJ5041

```cpp
//Adam Iskandar
//A21MJ5041
//Computer Intelligence
//Assignment 1

#include <iostream>
#include <vector>
#include <cstdlib>
#include <ctime>
#include <iomanip>

using namespace std;

class Perceptron {
private:
    vector<double> weights;
    double threshold;
    double learningRate;

public:
    // Constructor
    Perceptron(int numInputs, double threshold = 0.5, double
learningRate = 0.1) {
        // Initialize weights randomly
        srand(time(NULL));
        for (int i = 0; i < numInputs; ++i) {
            weights.push_back((double)rand() / RAND_MAX);
        }
        this->threshold = threshold;
        this->learningRate = learningRate;
    }

    // Activation function (Step function)
    int activation(double sum) {
        return (sum >= threshold) ? 1 : 0;
    }

    // Predict method
    int predict(vector<int> inputs) {
        double sum = 0;
        for (size_t i = 0; i < inputs.size(); ++i) {
            sum += inputs[i] * weights[i];
        }
        return activation(sum);
    }
```

```
    // Train method
    void train(vector<vector<int>> trainingData, vector<int> labels, int
epochs = 100) {
        for (int epoch = 0; epoch < epochs; ++epoch) {
            for (size_t i = 0; i < trainingData.size(); ++i) {
                int prediction = predict(trainingData[i]);
                int error = labels[i] - prediction;
                for (size_t j = 0; j < weights.size(); ++j) {
                    weights[j] += learningRate * error *
trainingData[i][j];
                }
            }
        }
    }

    // Getter method for weights
    vector<double> getWeights() {
        return weights;
    }
};

// Function to print truth table
void printTruthTable(const vector<vector<int>>& data, const vector<int>&
labels, const string& gate_name) {
    cout << "Truth Table for " << gate_name << " Gate:" << endl;
    cout << "-------------------------" << endl;
    cout << "Input(s) | Output" << endl;
    cout << "-------------------------" << endl;
    for (size_t i = 0; i < data.size(); ++i) {
        for (size_t j = 0; j < data[i].size(); ++j) {
            cout << setw(4) << data[i][j] << " ";
        }
        cout << "| " << setw(6) << labels[i] << endl;
    }
    cout << "-------------------------" << endl;
}

int main() {
    // OR problem
    vector<vector<int>> orData = {{0, 0}, {0, 1}, {1, 0}, {1, 1}};
    vector<int> orLabels = {0, 1, 1, 1};

    // Create and train OR Perceptron
    Perceptron orPerceptron(2, 0.2, 0.1); // 2 input nodes, threshold =
0.2, learning rate = 0.1
    orPerceptron.train(orData, orLabels);
```

```
    // NOT problem
    vector<vector<int>> notData = {{0}, {1}};
    vector<int> notLabels = {1, 0};

    // Create and train NOT Perceptron
    Perceptron notPerceptron(1, 0.2, 0.1); // 1 input node, threshold =
0.2, learning rate = 0.1
    notPerceptron.train(notData, notLabels);

    // Print truth tables
    printTruthTable(orData, orLabels, "OR");
    cout << endl;
    printTruthTable(notData, notLabels, "NOT");
    cout << endl;

    // Print final weights
    cout << "Final Weights for OR Gate: ";
    for (auto weight: orPerceptron.getWeights()) {
        cout << weight << " ";
    }
    cout << endl;

    cout << "Final Weights for NOT Gate: ";
    for (auto weight: notPerceptron.getWeights()) {
        cout << weight << " ";
    }
    cout << endl;

    return 0;
}
```

Output:

```
Truth Table for OR Gate:
-------------------------
Input(s) | Output
-------------------------
   0    0 |      0
   0    1 |      1
   1    0 |      1
   1    1 |      1
-------------------------

Truth Table for NOT Gate:
-------------------------
Input(s) | Output
-------------------------
   0 |      1
   1 |      0
-------------------------

Final Weights for OR Gate: 0.258461 0.617115
Final Weights for NOT Gate: 0.158461

--------------------------------
Process exited after 0.1597 seconds with return value 0
Press any key to continue . . .
```