Computer Intelligence Lab - CS354N

Weekly Assignment Report

Question 1:

```C/C++
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

float randomFloat() {
    return static_cast<float>(rand()) / static_cast<float>(RAND_MAX);
}

int main() {
    srand(static_cast<unsigned int>(time(0)));

    float w1, w2, theta;
    w1 = randomFloat();
    w2 = randomFloat();
    theta = 0.5;
```

```
    cout << "The weights are " << w1 << " and " << w2 << "\n";
    cout << "Theta value is " << theta << "\n\n";

    float input1, input2;
    cout << "Enter the 2 inputs: ";
    cin >> input1 >> input2;
    cout << "\n\n";

    float res = w1 * input1 + w2 * input2;
    int ans = (res >= theta) ? 1 : 0;

    cout << "The output is: " << ans << "\n";
    return 0;
}
```

Output:

```
┌──(thenetherwatcher㉿kali)-[~/Documents/CS354-Lab/Lab-4]
└─$ ./a.out
The weights are 0.78393 and 0.0990984
Theta value is 0.5

Enter the 2 inputs: 1 2


The output is: 1

┌──(thenetherwatcher㉿kali)-[~/Documents/CS354-Lab/Lab-4]
└─$ ./a.out
The weights are 0.569075 and 0.658523
Theta value is 0.5

Enter the 2 inputs: 1 1


The output is: 1
```

Question 2,3,4,5:

```
C/C++

#include <iostream>

#include <vector>

#include <fstream>

#include <sstream>


using namespace std;


struct Perceptron {

    vector<double> weights;

    double eta;  // Learning rate

    int theta;   // Threshold

    int bias_ip; // Bias input


    Perceptron(double eta, int theta, int input_size) {

        this->eta = eta;

        this->theta = theta;

        this->bias_ip = 1;

        weights = vector<double>(input_size + 1, 0); // Initializing weights
including bias weight

    }


    void train(const vector<pair<vector<int>, int>> &data) {
```

```
    bool flag = true;
    int epoch = 0;

    while (flag) {
        flag = false;
        for (const auto &sample : data) {
            const vector<int> &inputs = sample.first;
            int target = sample.second;
            double net = weights[0] * bias_ip;
            for (size_t i = 0; i < inputs.size(); ++i) {
                net += weights[i + 1] * inputs[i];
            }
            int output = (net >= theta) ? 1 : 0;

            int error = target - output;
            if (error != 0) {
                flag = true;
                weights[0] += eta * error * bias_ip;
                for (size_t i = 0; i < inputs.size(); ++i) {
                    weights[i + 1] += eta * error * inputs[i];
                }
            }
        }
```

```
            epoch++;
        }
    }

    int predict(const vector<int> &inputs) {
        double net = weights[0] * bias_ip;
        for (size_t i = 0; i < inputs.size(); ++i) {
            net += weights[i + 1] * inputs[i];
        }
        return (net >= theta) ? 1 : 0;
    }
};

vector<pair<vector<int>, int>> load_data(const string &filename) {
    vector<pair<vector<int>, int>> data;
    ifstream file(filename);
    string line;

    while (getline(file, line)) {
        stringstream ss(line);
        vector<int> inputs;
        int input, target;
```

```
        while (ss >> input) {
            inputs.push_back(input);
        }
        target = inputs.back();
        inputs.pop_back();
        data.push_back({inputs, target});
    }

    return data;
}

int main() {
    int choice;
    cout << "Select the gate to train the perceptron model:\n";
    cout << "1. AND\n";
    cout << "2. NAND\n";
    cout << "3. OR\n";
    cout << "4. NOR\n";
    cout << "Enter your choice: ";
    cin >> choice;

    string filename;
    switch (choice) {
```

```
        case 1:
            filename = "example_and.txt";
            break;
        case 2:
            filename = "example_nand.txt";
            break;
        case 3:
            filename = "example_or.txt";
            break;
        case 4:
            filename = "example_nor.txt";
            break;
        default:
            cout << "Invalid choice!" << endl;
            return 1;
    }


    vector<pair<vector<int>, int>> data = load_data(filename);
    if (data.empty()) {
        cout << "Failed to load data from " << filename << endl;
        return 1;
    }
```

```
    int input_size = data[0].first.size();
    Perceptron perceptron(0.1, 1, input_size);
    perceptron.train(data);

    cout << "Trained weights: ";
    for (double weight : perceptron.weights) {
        cout << weight << " ";
    }
    cout << endl;

    cout << "Predictions:" << endl;
    for (const auto &sample : data) {
        const vector<int> &inputs = sample.first;
        cout << "Inputs: ";
        for (int input : inputs) {
            cout << input << " ";
        }
        cout << "Output: " << perceptron.predict(inputs) << endl;
    }

    return 0;
}
```

Output:

```
┌──(thenetherwatcher㉿kali)-[~/Documents/CS354-Lab/Lab-4]
└─$ ./a.out
Select the gate to train the perceptron model:
1. AND
2. NAND
3. OR
4. NOR
Enter your choice: 1
Trained weights: 0.2 0.2 0.2 0.2 0.2
Predictions:
Inputs: 0 0 0 0 Output: 0
Inputs: 0 0 0 1 Output: 0
Inputs: 0 0 1 0 Output: 0
Inputs: 0 0 1 1 Output: 0
Inputs: 0 1 0 0 Output: 0
Inputs: 0 1 0 1 Output: 0
Inputs: 0 1 1 0 Output: 0
Inputs: 0 1 1 1 Output: 0
Inputs: 1 0 0 0 Output: 0
Inputs: 1 0 0 1 Output: 0
Inputs: 1 0 1 0 Output: 0
Inputs: 1 0 1 1 Output: 0
Inputs: 1 1 0 0 Output: 0
Inputs: 1 1 0 1 Output: 0
Inputs: 1 1 1 0 Output: 0
Inputs: 1 1 1 1 Output: 1
```

```
┌──(thenetherwatcher㉿kali)-[~/Documents/CS354-Lab/Lab-4]
└─$ ./a.out
Select the gate to train the perceptron model:
1. AND
2. NAND
3. OR
4. NOR
Enter your choice: 2
Trained weights: 1.7 -0.4 -0.2 -0.1 -0.1
Predictions:
Inputs: 0 0 0 0 Output: 1
Inputs: 0 0 0 1 Output: 1
Inputs: 0 0 1 0 Output: 1
Inputs: 0 0 1 1 Output: 1
Inputs: 0 1 0 0 Output: 1
Inputs: 0 1 0 1 Output: 1
Inputs: 0 1 1 0 Output: 1
Inputs: 0 1 1 1 Output: 1
Inputs: 1 0 0 0 Output: 1
Inputs: 1 0 0 1 Output: 1
Inputs: 1 0 1 0 Output: 1
Inputs: 1 0 1 1 Output: 1
Inputs: 1 1 0 0 Output: 1
Inputs: 1 1 0 1 Output: 1
Inputs: 1 1 1 0 Output: 1
Inputs: 1 1 1 1 Output: 0
```

```
┌──(thenetherwatcher㉿kali)-[~/Documents/CS354-Lab/Lab-4]
└─$ ./a.out
Select the gate to train the perceptron model:
1. AND
2. NAND
3. OR
4. NOR
Enter your choice: 3
Trained weights: 0.8 0.2 0.3 0.3 0.3
Predictions:
Inputs: 0 0 0 0 Output: 0
Inputs: 0 0 0 1 Output: 1
Inputs: 0 0 1 0 Output: 1
Inputs: 0 0 1 1 Output: 1
Inputs: 0 1 0 0 Output: 1
Inputs: 0 1 0 1 Output: 1
Inputs: 0 1 1 0 Output: 1
Inputs: 0 1 1 1 Output: 1
Inputs: 1 0 0 0 Output: 1
Inputs: 1 0 0 1 Output: 1
Inputs: 1 0 1 0 Output: 1
Inputs: 1 0 1 1 Output: 1
Inputs: 1 1 0 0 Output: 1
Inputs: 1 1 0 1 Output: 1
Inputs: 1 1 1 0 Output: 1
Inputs: 1 1 1 1 Output: 1
```

```
┌──(thenetherwatcher㉿kali)-[~/Documents/CS354-Lab/Lab-4]
└─$ ./a.out
Select the gate to train the perceptron model:
1. AND
2. NAND
3. OR
4. NOR
Enter your choice: 4
Trained weights: 1.1 -0.1 -0.1 -0.1 -0.1
Predictions:
Inputs: 0 0 0 0 Output: 1
Inputs: 0 0 0 1 Output: 0
Inputs: 0 0 1 0 Output: 0
Inputs: 0 0 1 1 Output: 0
Inputs: 0 1 0 0 Output: 0
Inputs: 0 1 0 1 Output: 0
Inputs: 0 1 1 0 Output: 0
Inputs: 0 1 1 1 Output: 0
Inputs: 1 0 0 0 Output: 0
Inputs: 1 0 0 1 Output: 0
Inputs: 1 0 1 0 Output: 0
Inputs: 1 0 1 1 Output: 0
Inputs: 1 1 0 0 Output: 0
Inputs: 1 1 0 1 Output: 0
Inputs: 1 1 1 0 Output: 0
Inputs: 1 1 1 1 Output: 0
```