**QUESTION 1**

The binary perceptron algorithms is a machine learning algorithm that classifies data into two groups. It assumes that the data points are linearly separable (a line can separate all the data into it’s two classes with no training error), therefore it’s accuracy is highly dependent on this.

Each of the inputs (i.e. features) of the data is assigned a weight (for our algorithm this is initially assigned as 0) and one bias is assigned to the data as a whole (for our algorithm this is initially assigned as 0). And activation score is then calculated by summing the product of the weights and the feature inputs, and adding the bias. To check if the classification is correct, the target (either 1 or -1) is multiplied by the activation score. If the result of this multiplication is positive, the classification is correct. If it is negative, the classification is wrong because:

|  |  |  |
| --- | --- | --- |
| **Actual** | **Algorithm Prediction** | **Product Result** |
| **1** | **1** | **1** |
| **-1** | **-1** | **1** |
| **1** | **-1** | **-1** |
| **-1** | **1** | **-1** |

So, all wrong predictions will always result in a negative product as will need adjustment. The weights are made bigger by adding them to the product of the prediction value and the feature input value to bring it closer to being positive. The bias is also adjusted by adding the prediction value to it and the loop begins again.

When the maximum number of iterations are reached, an adusted bias and weights are returned which should be more accurate. These will then be used when attempting to predict the outputs.

**PSEUDOCODE**

Import necessary libraries

Load the files with the training and test data OR split the data into a training and test set

Create a function to train the Perceptron algorithm with inputs as Training Data & Maximum iterations i.e. PerceptronTrain(Training Data: td, MaxIterations)

Weights initialised as 0 for all elements of td i.e. **(wi = 0 for all i = 1, ….td;)**

The initial bias is 0 i.e **b = 0**

**Begin algorithm FOR iteration 1 …. MaxIterations do** performs algorithm for MaxIerations times

**For all (x̄, y) ∈ td do:** for all features x̄ and predictions y perform the following

**A = W̅T \* X + b** Activation score is the weights times the features plus the bias

**If y\*A <= 0 then** if the activation is below 0 W̅and b need updating

**wi = wi + y ⋅ xi for all i = 1,…, td** weights updated by adding y times features

**b = b + y** new bias by adding y to the bias

**return b, w1, w2 …, wd** This returns the updated weights and biases

**QUESTION 2**

Refer to python document

**QUESTION 3**

Using unshuffled data, the accuracy when testing the perceptron against the train and test data is as follows:

**Class-1 vs Class-2:**

Test data: - 100%

Train data: - 100%

**Class-1 vs Class-3:**

Test data: - 100%

Train data: - 100%

**Class-2 vs Class-3:**

Test data: - 50%

Train data: - 50%

The pair of classes that is most difficult to separate is class 2 and class 3.

**QUESTION 4:**

For 1 vs rest the accuracies:

Test data: - 66.67%

Train data: - 66.67%

**QUESTION 5:**

For L2 the accuracies:

|  |  |  |
| --- | --- | --- |
| **L2 coefficient** | **Training Accuracy(%)** | **Test Accuracy (%)** |
| 0.01 | 66.67 | 66.67 |
| 0.1 | 66.67 | 66.67 |
| 1 | 33.33 | 33.33 |
| 10 | 33.33 | 33.33 |
| 100 | 33.33 | 33.33 |