BITS F464
MACHINE LEARNING

ASSIGNMENT 1C
Linear Perceptron Model

Team members:

Parth Krishna Sharma : 2017B3A70907H

Siddhi Burse : 2017B3A70972H

Prajjwal Vijaywargiya : 2017B3A70954H

I) Description of Linear Perceptron model

Linear perceptron model is a supervised machine learning algorithm for binary classification problems. As it is a supervised learning algorithm, the model trains itself using the training data and then the accuracy of the model is checked based on testing data examples.

The aim of the linear perceptron is to build a linear discriminant that can classify the given examples into their respective class.

The model implemented classifies all points into 2 classes:

Class1 : +1 , Class2: -1

X : feature vector

Y : target attribute

As the units of the feature vector variables are unknown, the values are standardized using the below formula:

$$\boxed{X' = \frac{X - \mu}{\sigma}}$$

Standardization also helps the gradient descent algorithm to converge to the minima faster.

An extra feature with constant value 1 is added to the feature vector to accommodate the bias term.

The entire dataset is split into 2 datasets – training data (70%) & testing data(30%).

The linear classifier to be found is of the form:

$$\boxed{w_0 + w_1 * x_1 + w_2 * x_2 + w_3 * x_3 + \ldots = 0}$$

The linear perceptron algorithm uses the stochastic gradient method to update the weights based on every misclassified training example.

Hyperparameters values chosen are:

Epochs = 1000

Learning rate = 0.01

The weights are initialized to 0 and are updated using the following formula:

weight = weight + (learning_rate)*(y_actual – y_predicted) * x

Prediction of target attribute is done by calculating the activation value and then applying the activation function on it.

The activation value for predicting the target attribute is calculated as:

activation = X.dot(weights)

Then the activation function is applied to obtain the predicted value as follows:

Y_prediction = 1.0 if activation >= 0.0 else -1.0

For every epoch, the loss value is calculated using the formula:

$$\text{Loss} = \sum y_i * (\text{weight})^T * x_i$$

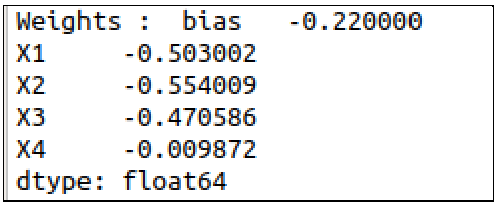
for all misclassified points.

The algorithm stops once all points have been correctly classified.

Results:

1. Weight values obtained for **1st dataset** are as follows:

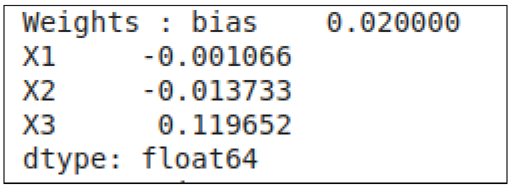
$w_0 = -0.22$, $w_1 = -0.50$, $w_2 = -0.55$, $w_3 = -0.47$, $w_4 = -0.0098$



```
Weights :  bias    -0.220000
X1      -0.503002
X2      -0.554009
X3      -0.470586
X4      -0.009872
dtype: float64
```

2. Weight values obtained for **2nd dataset** are as follows:

$w_0 = 0.02$, $w_1 = -0.001$, $w_2 = -0.013$, $w_3 = 0.1196$

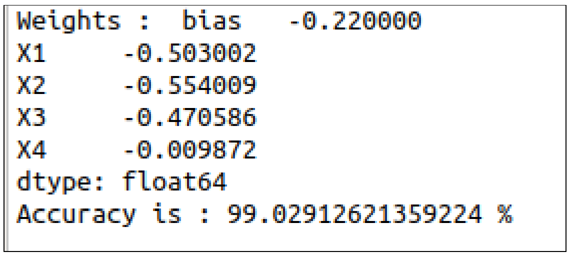


```
Weights : bias     0.020000
X1      -0.001066
X2      -0.013733
X3       0.119652
dtype: float64
```

II) Accuracy of models

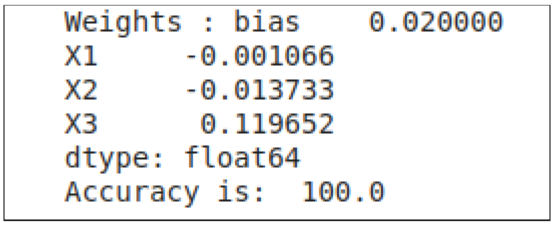
Accuracy of a model is calculated by dividing the total number of correctly classified examples by the total number of examples in the testing dataset.

Accuracy for dataset 1 = **99.029%**



```
Weights :  bias   -0.220000
X1     -0.503002
X2     -0.554009
X3     -0.470586
X4     -0.009872
dtype: float64
Accuracy is : 99.02912621359224 %
```

Accuracy for dataset 2 = **100%**



```
Weights : bias    0.020000
X1      -0.001066
X2      -0.013733
X3       0.119652
dtype: float64
Accuracy is:  100.0
```

III) Dataset which is more linearly separable

As the accuracy of the 2nd dataset is higher ,i.e. the model was able to correctly classify more number of examples for the 2nd dataset rather than the 1st dataset, we can say that the **2nd dataset is more linearly separable**.

IV) Limitations of Perceptron classifier

The main limitations of the perceptron classifier are:

1. The output value of a perceptron can only take binary values for classification.

2. It works on the assumption that all points are linearly separable which is an unrealistic example for many real-life problems. The algorithm keeps updating weight values until all points have not been classified into their appropriate classes, i.e. if the points are not linearly separable , the algorithm runs infinitely.