Aim of Project / Problem Statement

This project teaches us how actually the multiple Fisher’s Linear Discriminant Analysis (FLDA) (also called simply as linear discriminant analysis (LDA)) algorithm is used to solve the classification problem and how it is written in the sklearn library and the indepth mathematical intuition behind this algorithm because i have implemented this algorithm from scratch

The entire theory for the implementation can be understood by referring to my notes at following link (recommended as must read) :
https://drive.google.com/drive/folders/1Q0gmJ3FPJTrRQclG2XM2-Anx2N0sdlQK?usp=sharing

The dataset used can be found at this link :
https://drive.google.com/file/d/1R8ZJqLYUztj_IpeaRVxCdCMUi4CqDLih/view?usp=sharing

The code for the implementation from scratch can be found at following link :
https://colab.research.google.com/drive/1jEM7odMQ_ha-_86VomrImx1KPczyb_Ln?usp=sharing(open with google colab)

Code Design

The code can be divided into following subsections for clear understanding:

A. Importing Required Libraries:

You need to import some essential basic manipulation libraries to do some math operations and hence we are importing these libraries here.Import numpy, pandas and matplotlib libraries to successfully run this project

B. Data Preprocessing:

1. First we use pandas library to read the dataset from the location where it has been uploaded.

   The dataset used in this project consists of three independent features i.e. X1,X2 and X3 and a dependent feature Y. Our goal is to use these independent features to predict the values of dependent feature for which we will use FLDA algorithm.

2. The given dataset contains features with almost the same scales hence we don’t need to perform any feature scaling i.e. we need not do any normalization / standardization on the dataset.

3. Using the entire dataset as a train dataset. We know that the validation dataset is used for hyperparameter tuning and since there is no hyperparameter in this algorithm we don't need the validation dataset.Since we have only 1000 data points we are not even using test dataset.

C. Data Learning Without Using SKLearn Library

1. Defining Our LDA Class:

To understand the math and derivation of formulas used in this class the reader must go through the notes linked above.Once the reader has understood the mathematical derivation for the formulas provided in the notes the reader will take no time to understand the code since the coding is done using similar notations for easy understanding. Moreover additional comments have been added for external help. This class has following functions defined within it

The fit_transform() method is used to calculate the mean vector and covariance matrix of the respective classes in the original feature space .

The weight_vector() method is used to calculate the line onto which the original feature should be projected to convert it into 1 feature space using information from the fit_transform() method.

The projections1D() method is used to project the original feature space onto the line that we got in the weight_vector() method to get the 1 feature space.

The threshold_point() method is used to find the discriminant point using the 1 feature space data that we get using the projections1D() method.

The predict() method is used to predict the values using the threshold point that we got using the threshold_point() method.

2. Defining Our Evaluation Metric Class:

This class is used to calculate the performance of our model. Now we know that there are various performance measures but here we have implemented one of them namely accuracy.The formula of accuracy is given by the following mathematical equation.

$$\text{Accuracy} = \frac{True\ Positives + Ture\ Negatives}{True\ Positives + Ture\ Negatives + False\ Positives + False\ Negatives}$$

Values of true positives , true negatives , false positives and false negatives can be calculated using the confusion matrix.

For obvious reasons the higher the accuracy i.e. closer to 1 the better is our model and vice versa.

3. Data Learning using the user defined classes :

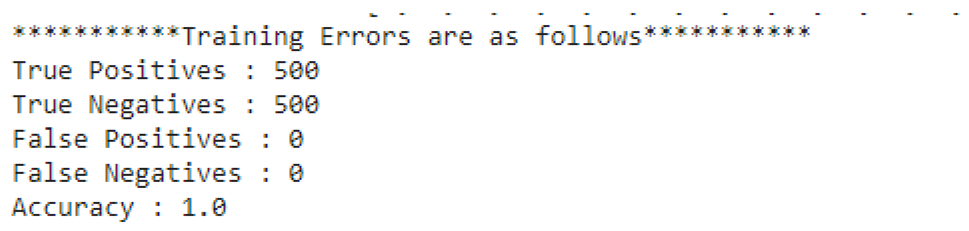
<u>The unit weight vector after applying the algorithm is</u>

```
Unit(Normalized) Weight vector is : [[ 0.00655686  0.01823739
-0.99981218]]
```

<u>Equations of Discriminating line and plane and threshold point</u>

```
Threshold Point is: 0.3893028020993766
Unit 1D Vector (Discriminating Line) is : X = 0.3893028020993766
Unit 3D Vector (Discriminating Plane) is :
[0.00655686]X1+[0.01823739]X2+[-0.99981218]X3 = 0.3893028020993766
```

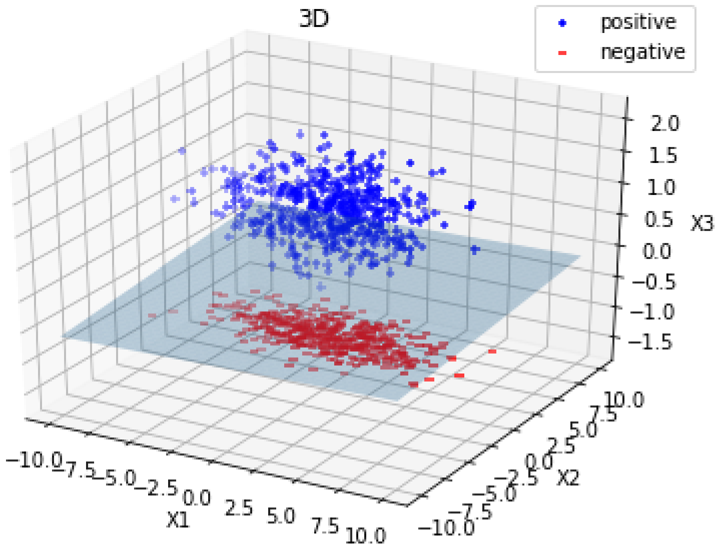
<u>Final Accuracy Achieved</u>



```
***********Training Errors are as follows***********
True Positives : 500
True Negatives : 500
False Positives : 0
False Negatives : 0
Accuracy : 1.0
```

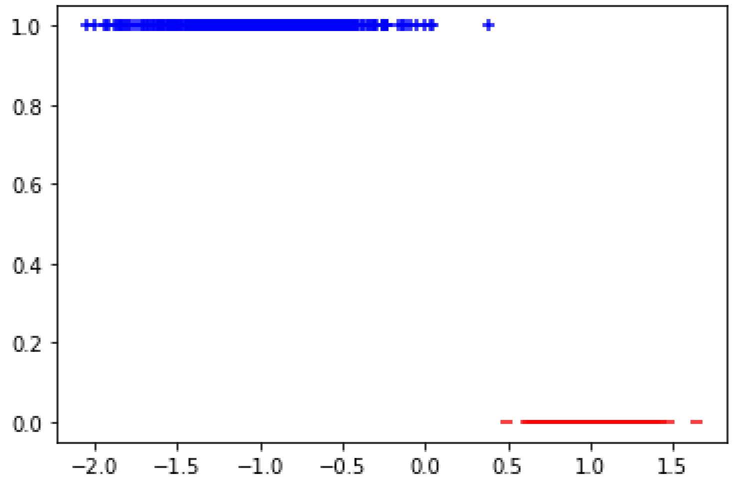
Plot for Higher Dimensional Data with the Discriminating Plane

```
Text(0.5, 0.92, '3D')Text(0.5, 0, 'X1')Text(0.5, 0, 'X2')Text(0.5, 0, 'X3')
<mpl_toolkits.mplot3d.art3d.Poly3DCollection at 0x7f4842a99d10><matplotlib.
```



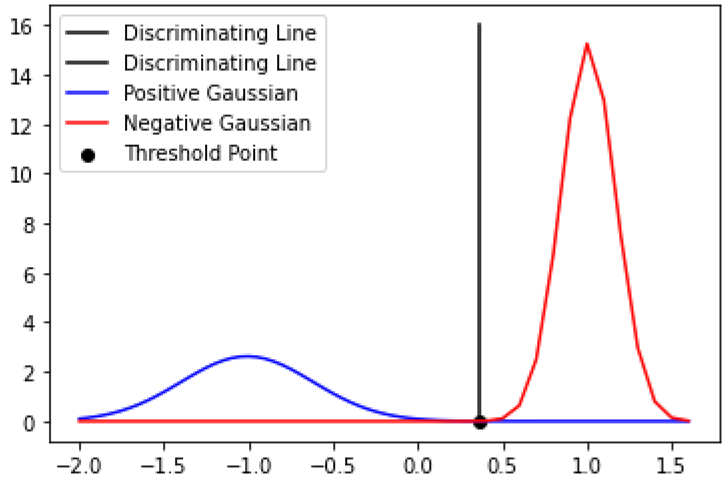
Plot for Projected 1 Feature Data (Reduced Clusters)

```
<matplotlib.collections.PathCollection at 0x7f4841c5ba10><matplotli
```



Plot for Gaussians with Discriminating Line and Discriminating Point

```
<matplotlib.collections.PathCollection at 0x7f4841dedb50>[<mat
 <matplotlib.lines.Line2D at 0x7f4841eb6b50>][<matplotlib.line
```


Discriminating Line
Discriminating Line
Positive Gaussian
Negative Gaussian
Threshold Point