**Fisher’s Linear Discriminant Analysis**

**Overview**

Fisher’s LDA method is one of the popular methods used in binary and multi-classification problem. This involves reduction of parameters, so as to reduce the number of dimensions of a point from ‘n’ to 2. This way, we would have a linear distribution of points belonging to all the classes. By finding the mean of the set of points belonging to each class, we can find the discriminating point between any two classes. This method of finding the discriminating point works under the assumption that the set of points belonging to each class follows a Gaussian Distribution. So, finding the discriminating point would essentially mean that we would be finding the point of intersection of Normal distribution of both set of points. So, when a new testing point is given, we first project it onto our line, and then see if the point is above or below the discriminating point(in the case of Binary classification). To test the training and validation accuracy, there would be no necessity of a test-train split in this algorithm.

**Implementation**

The code for our implementation of the LDA can be found at the following repository:

<https://github.com/JuiP/ML/tree/LDA>

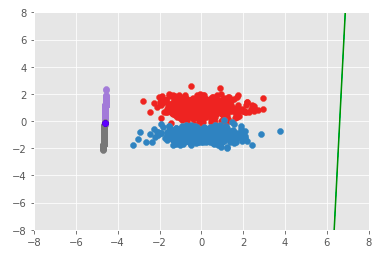
**1. For two parameters problem:**

In the beginning, we extracted the training examples provided to us, in the form of Pandas Dataframe. We were then able to obtain the parameters and final result in two different arrays. We then found out the mean and variance of both the sets of points, and used the inverse of sum of variances, and the difference of means to predict the Linear Discriminant. This would be the line onto which our points would be projected.

The next step was to project all of the training points onto the obtained line. This gave a whole new set of transformed datapoints. Then, we found the mean value of both the classes of points, and the mean of those two points turned out to be the discriminating point.

We have assumed that the points in each class follows a Normal Distribution, so finding the mean of means would essentially mean finding out the intersection point of the normal distribution graphs of both the classes.

The following graph depicts the projection of datapoints, and the discriminant point:



The F-score was calculated using the following formula:

F-score = (2\*Precision\*Recall) / (Precision + Recall), where

Precision = Actual no. of positive points / Total no. of positive points

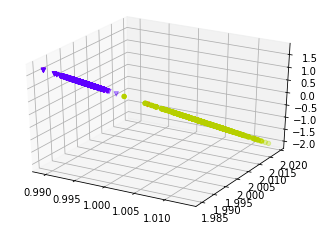
Recall = Actual no. of positve points / Total no. of points

We obtained an F-score of 0.99 and an accuracy of 98.9% for the given data.

**2. For three parameters problem:**

The approach here was similar to the one we did for the other problem. After extracting data from the Pandas Dataframe, we found out the inverse of sum of variances and differences of mean. Once the plane was obtained, we derived a line perpendicular to the plane, which would be the one onto which all the points would be projected. Later, we find the mean of means of both the distributions. This resulted in a discriminating point lying along the previously obtained line. We have assumed that the datapoints belonging to each class which are lying on the obtained line, follows a Normal distribution.

The following graph depicts the datapoints projected onto the line(discriminating point visible faintly in red color):

****

The F-score was calculated using the following formula:

F-score = (2\*Precision\*Recall) / (Precision + Recall), where

Precision = Actual no. of positive points / Total no. of positive points

Recall = Actual no. of positve points / Total no. of points

We obtained an F-score of 1.00 and an accuracy of 99.7% for the given data.