Fisher's Linear Discriminant

**a) Model and Implementation Description:**

Fisher's discriminant analysis is a classification problem that uses the dimensionality reduction principle to solve. The algorithm's goal is to find a hyperplane (or a vector) that divides the target variable into two or more groups. In order for FLD to work, the data must be continuous. For each type of dependent variable, the key assumption when using this algorithm is that the independent variables are normally distributed. The algorithm works by generating discriminant functions, which are linear combinations of predictors. There are one less of these discriminant functions (or boundaries) than there are groups. Since there are two separable groups in our case, the number of discriminant boundaries is one.

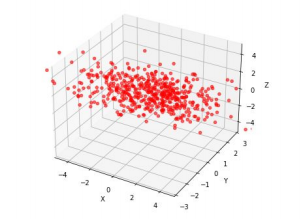
First, the data corresponding to the two groups is separated, and the mean for the two datasets thus produced, as well as the standard deviation, is determined. The between-class and within-class scatter matrices are created using this method. The discriminating vector is then obtained by taking the inverse of the dot product of these two matrices. Taking the dot product of this vector with the collection of independent variables yields the projection of the two groups. Eigenvalues and eigenvectors can be used to describe any mathematical transformation. To achieve the greatest possible discrimination between the necessary groups, these eigenvectors are generated and arranged in descending order of values.

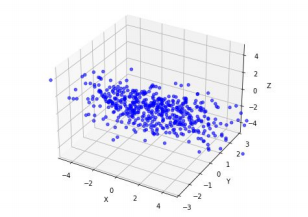
The required plots are generated with the matplotlib library. Different colours have been used to reflect the two different types of data. The class with the target variable value of 1 is shown in red, while the class with the target variable value of 0 is shown in blue. The normal vector and the point through which the plane moves, both of which are determined during the implementation of the algorithm, were used to create the hyperplane for the higher dimensional visualisation.

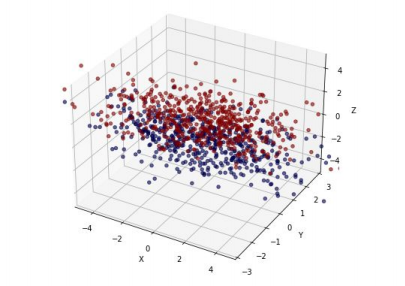
Since the classes are linearly separable, the algorithm produces a strong discriminant boundary capable of effectively linearly separating the two data classes.

**b) Matplotlib plotting of higher dimensional data:**

**Figure 1.a) C1 = val =1 Color = RED**

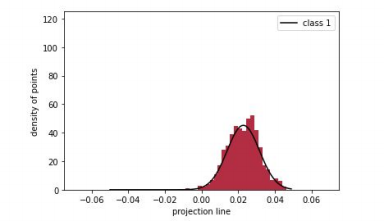
**Figure 1.b) C0 = val =0 Color = BLUE**

**Figure 1.c) In the higher dimension, the entire dataset is available.**

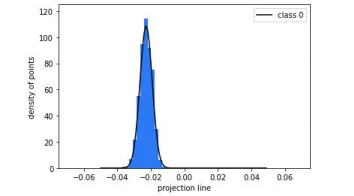
****

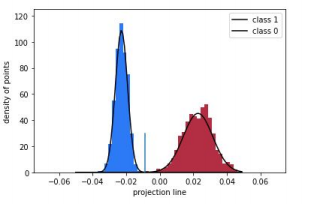
**c) Reduced clusters and the normal distribution that goes with them:**

**Figure 1.d) Normal distribution of reduced cluster C1.**

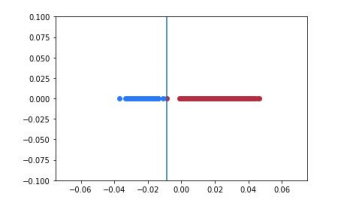
****

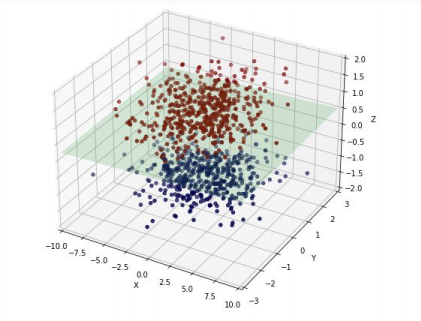
**Figure 1.e) Normal distribution of reduced cluster C0.**

**Figure 1.e) Normal distribution of the two clusters together, with the separating vector.**

****

**d) The intersection point of both the normal distributions and unit vector along the discriminant line**

**Figure 1.e) 1D projection of the two clusters together, with the separating vector.**

**Figure 1.f) 3D Scatter plot of the dataset, with the separating hyperplane. **

Since the dataset given to us was linearly separable, the accuracy was 100%, with no misclassifications