Assignment 1

Fischer Linear Discriminant Analysis

The algorithm is designed based on Fischer’s discriminant method by using Sw and m1,m2 means of different classes to find the separating vector.

* m1 = sum(data.iloc[points\_pos].values[:, :-1]) / l1  
  m2 = sum(data.iloc[points\_neg].values[:, :-1]) / l2  
  m1 = m1.reshape(len(m1), 1)   
  m2 = m2.reshape(len(m2), 1)

The means of the two classes(class 0 and class 1) are calculated by taking simple average of the data

* Sw is calculated using the following formula

SW=∑n∈C1(xn−m1)(xn−m1)^t +∑n∈C2(xn−m2)(xn−m2)^t

Where c1,c2 are two classes

w is calculated by the dot product of (Sw)^-1 and m1-m2

S\_pos = np.zeros((dim, dim), dtype=float) # initializing the matrix of dimension d\*d with zeroes  
S\_neg = np.zeros((dim, dim), dtype=float)  
for i in range(len(data)):  
 if data.iloc[i, -1:].values == 1:  
 S\_pos += np.dot(np.transpose(data.iloc[i, :-1].values - m1), np.asarray(data.iloc[i, :-1].values - m1))  
 else:  
 S\_neg += np.dot(np.transpose(data.iloc[i, :-1].values - m2), np.asarray(data.iloc[i, :-1].values - m2))  
S\_pos /= l1  
S\_neg /= l2  
Sw = S\_neg + S\_pos  
w = np.dot(np.linalg.inv(Sw), m1 - m2)

* The data is converted into 1D by dot product of w transpose and dataset values

and their values are stored in res\_vec.

res\_vec = np.zeros(len(data), dtype=float)  
for j in range(len(data)):  
 res\_vec[j] = np.dot(np.transpose(w), data.iloc[j, :-1].values) # wTX

* Normal distribution is plotted for both the classes.Normal distribution can be plotted by calculating mean and standard deviation for each of the classes.   
   mean\_p = np.mean(points\_p)  
   mean\_n = np.mean(points\_n)  
   std\_d\_p = np.std(points\_p)  
   std\_d\_n = np.std(points\_n)
* The intersection point of the normal distribution is the threshold based on which the points are classified. Threshold lies in between means of the curves.

The intersection point is the solution of a quadratic equation ax2+bx+c

Where a= 1/2\*(1/σ21 - 1/σ22)

B= (μ2/σ22 - μ1/σ21)

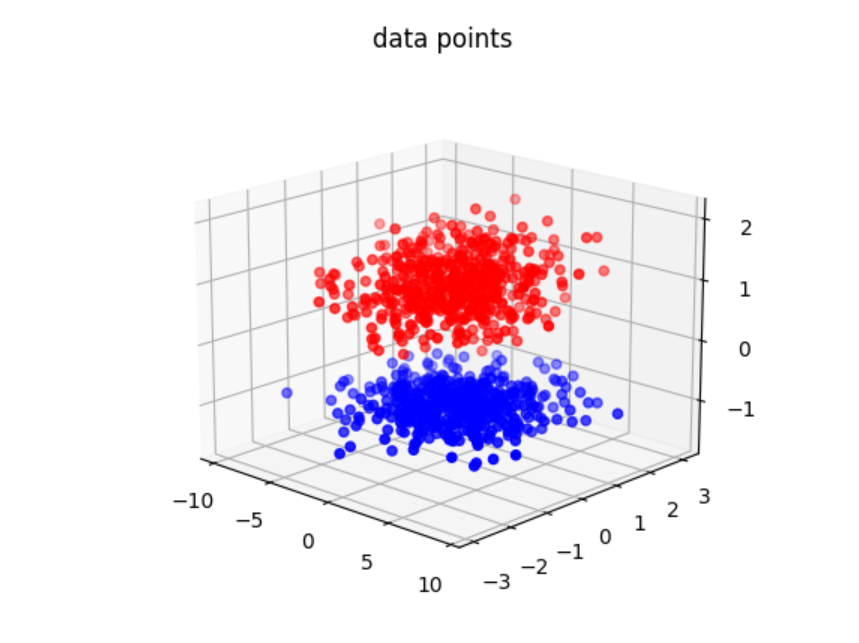
C= 1/2\*(μ21/σ21− μ22/σ22)-log(σ2/σ1).

a0 = 1 / 2 \* ((1 / std\_d\_p \*\* 2) - (1 / std\_d\_n \*\* 2))  
a1 = (mean\_n / std\_d\_n \*\* 2) - (mean\_p / std\_d\_p \*\* 2)  
a2 = 1 / 2 \* (mean\_p \*\* 2 / std\_d\_p \*\* 2 - mean\_n \*\* 2 / std\_d\_n \*\* 2) - np.log(std\_d\_n / std\_d\_p)

* The points are classified based on the value wTx >=threshold\_point.

**Plots:**

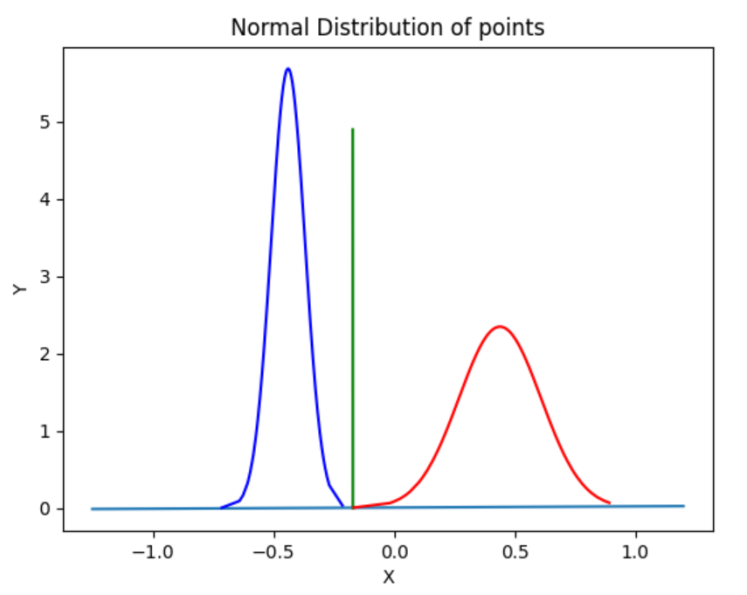
* Data points in 3d



Red-class 1 positive points

Blue-class 0 negative points

* Normal distribution of points:

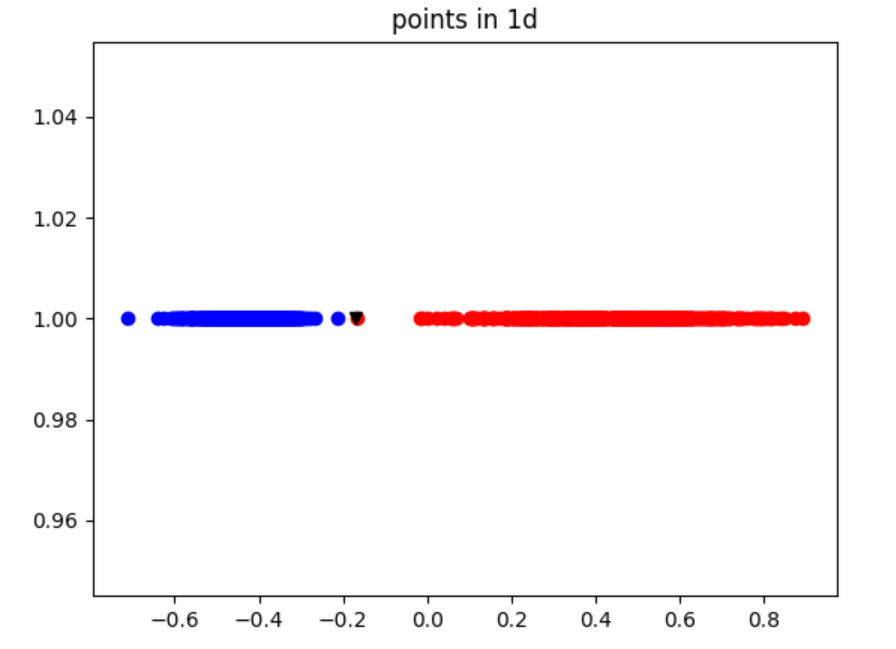


Blue line - w , Green line - discriminant line

Red - class 1 positive points

Blue - class 0 negative points

* Projected points in 1d:

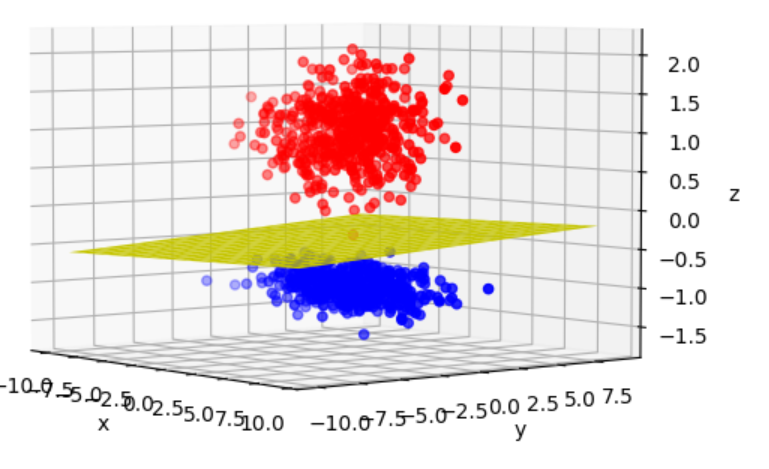


Red class 1 positive points

Blue class 0 negative points

Black point - threshold point

* 3d Points plotted along with the separator plane:



* Testing accuracy for the dataset = 100%
* Unit vector w: [[-0.00536519]

[-0.01538095]

[ 0.99986731]]

* Threshold value = -0.1703454405971936
* unit vector along discriminant line

In 1D: [[-0.16786669]

[-0.98569309]

[-0.01516291]]

In 3D: [[ 0.36415483]

[-0.0049968 ]

[-0.01432487]

[ 0.93121484]]

By:

Sharvani

Radhika Sigtia

Bathini Akash