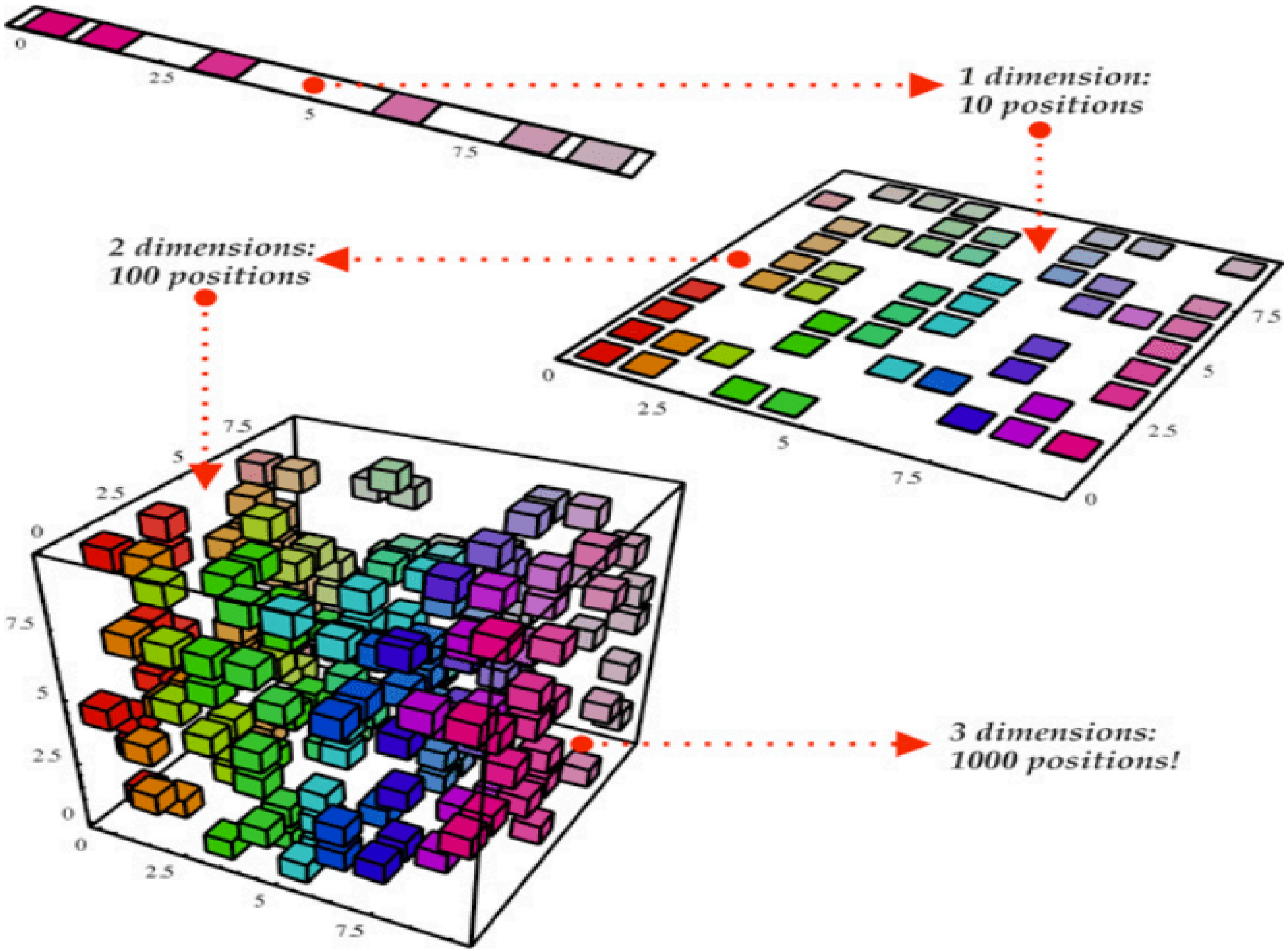
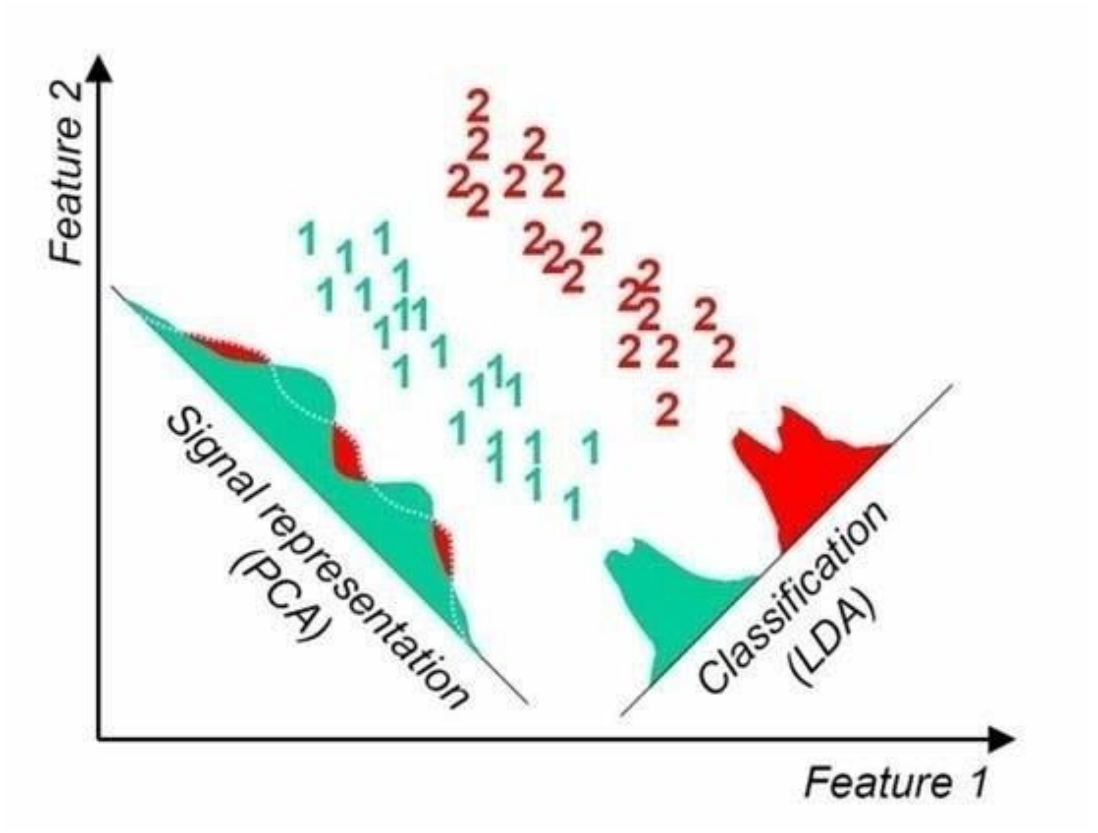
Basic


1 dimension:
10 positions
2 dimensions:
100 positions
3 dimensions:
1000 positions!
0
2.5
5
7.5

LDA

- Linear Discriminant Analysis or most commonly known as 'LDA' is one of the most interesting machine learning techniques till date.
- The idea was first coined by "Dr. Ronald Fisher" to classify binary classes using 'Fisher's linear discriminant' and later on it was generalized for multiple classes as well.
- In case of a binary class problem, LDA acts as a classifier like "logistic regression" and when it comes to multi classes it acts as a dimensional reduction technique like "PCA".
- In this article I will not be talking much about how to use LDA for a multi class problem, instead, let's focus on binary classes.
- However, let's take a peak into how "LDA" and "PCA" are different yet function the same.



Using Linear Discriminant Analysis on Binary Classification

- Step 1: Create a matrix containing the mean of each of the features (X) pertaining to a particular class say ‘Ci’.
- Step 2: Create a scatter matrix (Si) for each class using the following formula.

$$\mu_i = \frac{1}{N_i} \sum_{x \in c_i} x$$

Step 2: Create a scatter matrix (*Si*) for each class using the following formula.

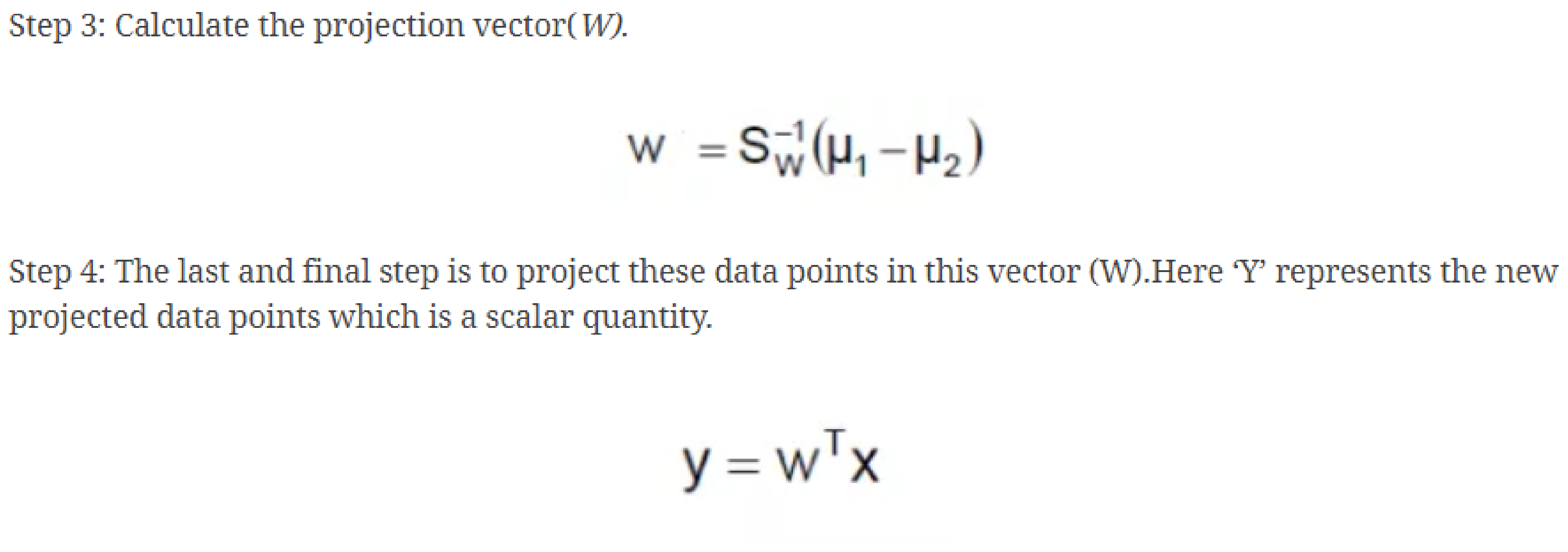
$$S_i = \sum (x - \mu_i)(x - \mu_i)^T$$

$$S_1 + S_2 = S_W$$

- Step 3: Calculate the projection vector(W).

Steps

- Step 4: The last and final step is to project these data points in this vector (W).Here ‘Y’ represents the new projected data points which is a scalar quantity.



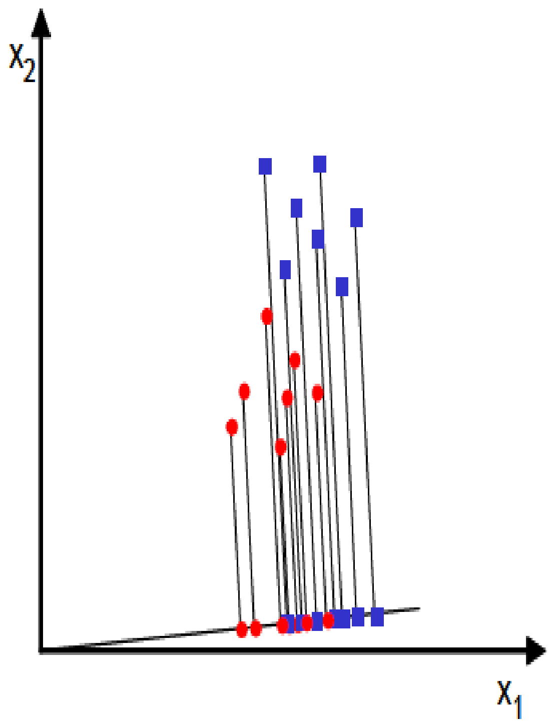
Step 3: Calculate the projection vector(*W*).

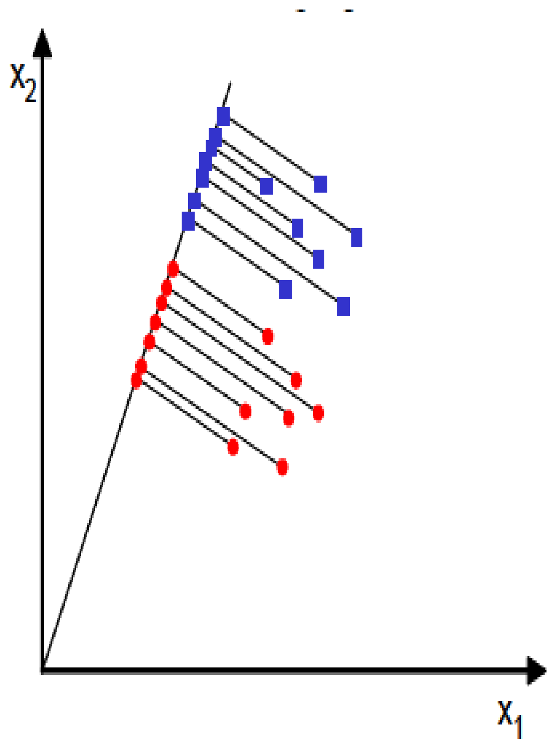
$$w = S_W^{-1}(\mu_1 - \mu_2)$$

Step 4: The last and final step is to project these data points in this vector (W).Here ‘Y’ represents the new projected data points which is a scalar quantity.

$$y = w^T x$$

Explanation-





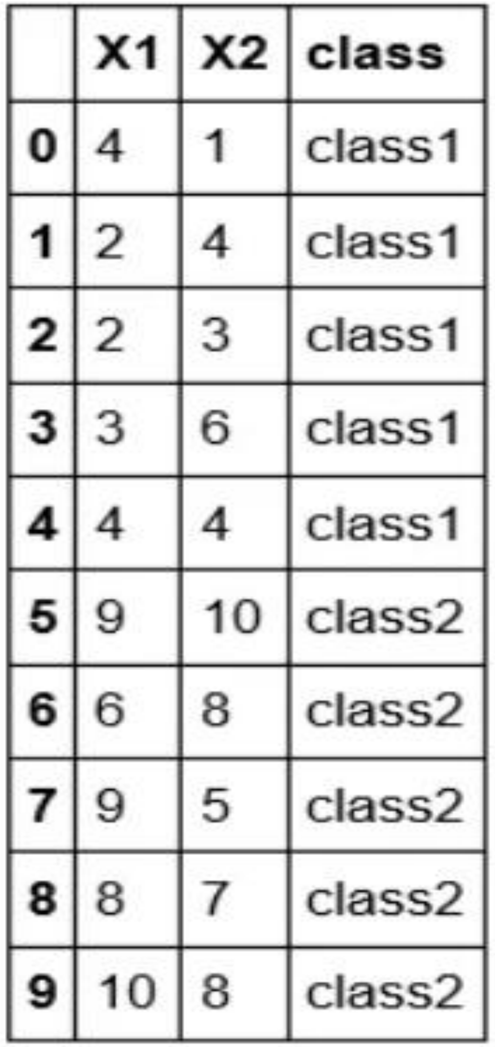
- Let’s look at the following figure of a two feature binary classification problem.You can clearly see how the data points are projected along the vector(the line represents W) after applying LDA.
- I have given you the basic difference between “LDA” and “PCA“, also I have given all the necessary steps for LDA but the question remains, How is LDA doing what it’s doing?
- I haven’t given you the intuition which led ‘Dr.Fisher‘ to formulate this beautiful technique. If you recall “logistic regression” what it does is that it takes a number of input features and represents in terms of probabilities.Well, LDA does the same except it takes the input feature vector(don’t get confused it means the set of features) and represents them in terms of scalar quantity.Yes, you read that right it converts vectors of features to scalars.Now, these scalars can be used to separate the two classes and very soon we will go through an example to get a better idea of the same.But first, let’s look at the following figure.

Linear Discriminant Analysis for Binary Classification

- *import numpy as np*
- *import pandas as pd*
- *import matplotlib.pyplot as plt*
- *from pandas import Series,DataFrame*
- *%matplotlib inline*

- First of all lets prepare a data set

- *data = DataFrame()*
- *X1 = Series(np.array([4,2,2,3,4,9,6,9,8,10]))*
- *X2 = Series(np.array([1,4,3,6,4,10,8,5,7,8]))*
- *data['X1']=X1*
- *data['X2']=X2*
- *data['class']=np.array(['class1','class1','class1','class1','class1','class2','class2','class2','class2','class2'])*

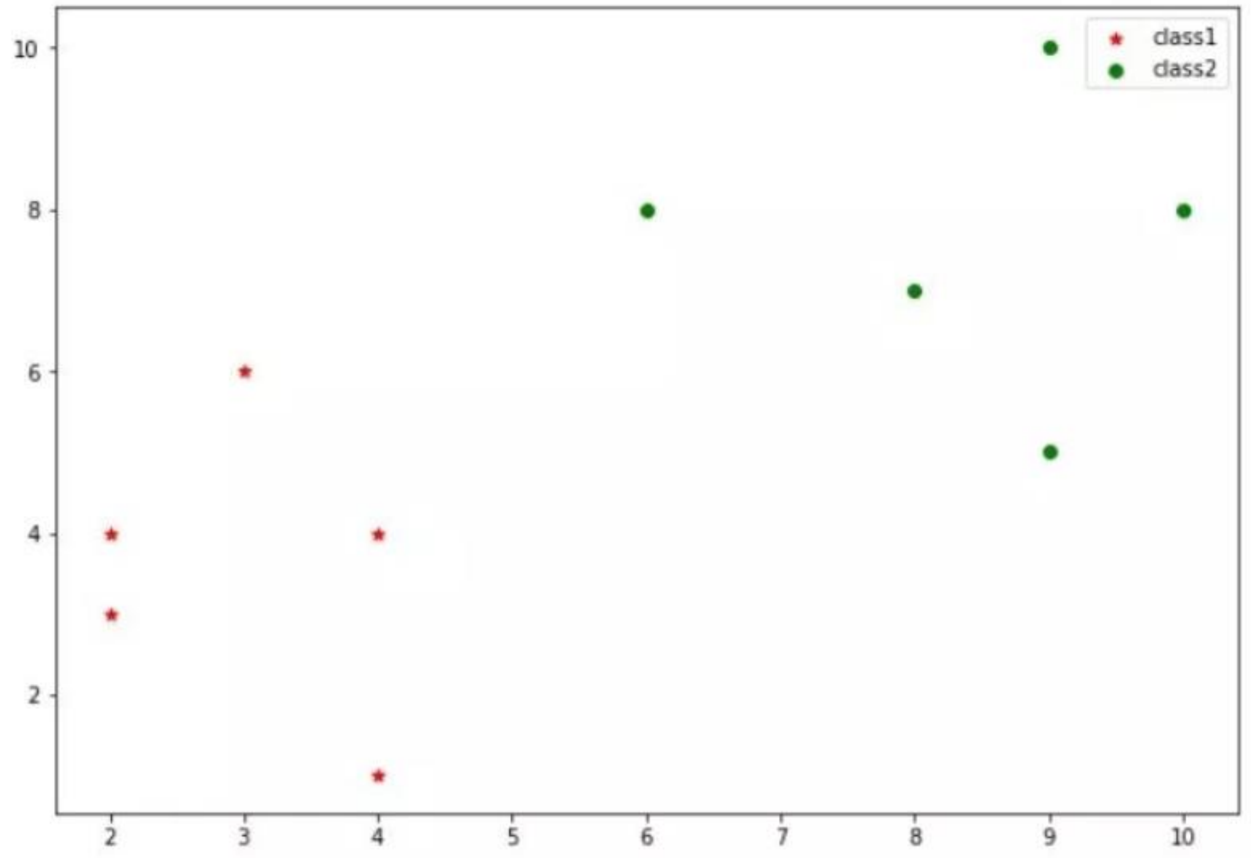


	X1	X2	class
0	4	1	class1
1	2	4	class1
2	2	3	class1
3	3	6	class1
4	4	4	class1
5	9	10	class2
6	6	8	class2
7	9	5	class2
8	8	7	class2
9	10	8	class2

Lets plot the above data and see how it looks

Code-

- *plt.figure(figsize=(10,7))*
- *plt.scatter(data.ix[data['class']=='class1',0],data.ix[data['class']=='class1',1],color='r',marker='*',label='class1')*
- *plt.scatter(data.ix[data['class']=='class2',0],data.ix[data['class']=='class2',1],color='g',marker='o',label='class2')*
- *plt.legend(loc='best')*



- Scatter Plot

Lets plot the above data and see how it looks

Using Step 1 to calculate the mean matrix

- *mu1 = np.array([np.mean(data.ix[data['class']=='class1', 0]),np.mean(data.ix[data['class']=='class1',1])])*
- *mu2 = np.array([np.mean(data.ix[data['class']=='class2', 0]),np.mean(data.ix[data['class']=='class2',1])])*
- *print(mu1,mu2)*
- *[3.   3.6] [8.4 7.6]*

Using Step 2 to find the scatter mattrices of each class and also the within class scatter matrix

- *s1 = np.dot((np.array(data.ix[data['class']=='class1',:-1])-mu1).T,np.array(data.ix[data['class']=='class1',:-1])-mu1)*
- *s1#scatter matrix for class 1*
- *array([[4. , -2.],*
- *[-2. , 13.2]])*
- *s2 = np.dot((np.array(data.ix[data['class']=='class2',:-1])-mu2).T,np.array(data.ix[data['class']=='class2',:-1])-mu2)*
- *s2#scatter matrix for class 2*
- *array([[9.2, -0.2],*
- *[-0.2, 13.2]])*
- *sw = s1 + s2*
- *sw#within class scatter matrix*
- *array([[13.2, -2.2],*
- *[-2.2, 26.4]])*

Lets plot the above data and see how it looks

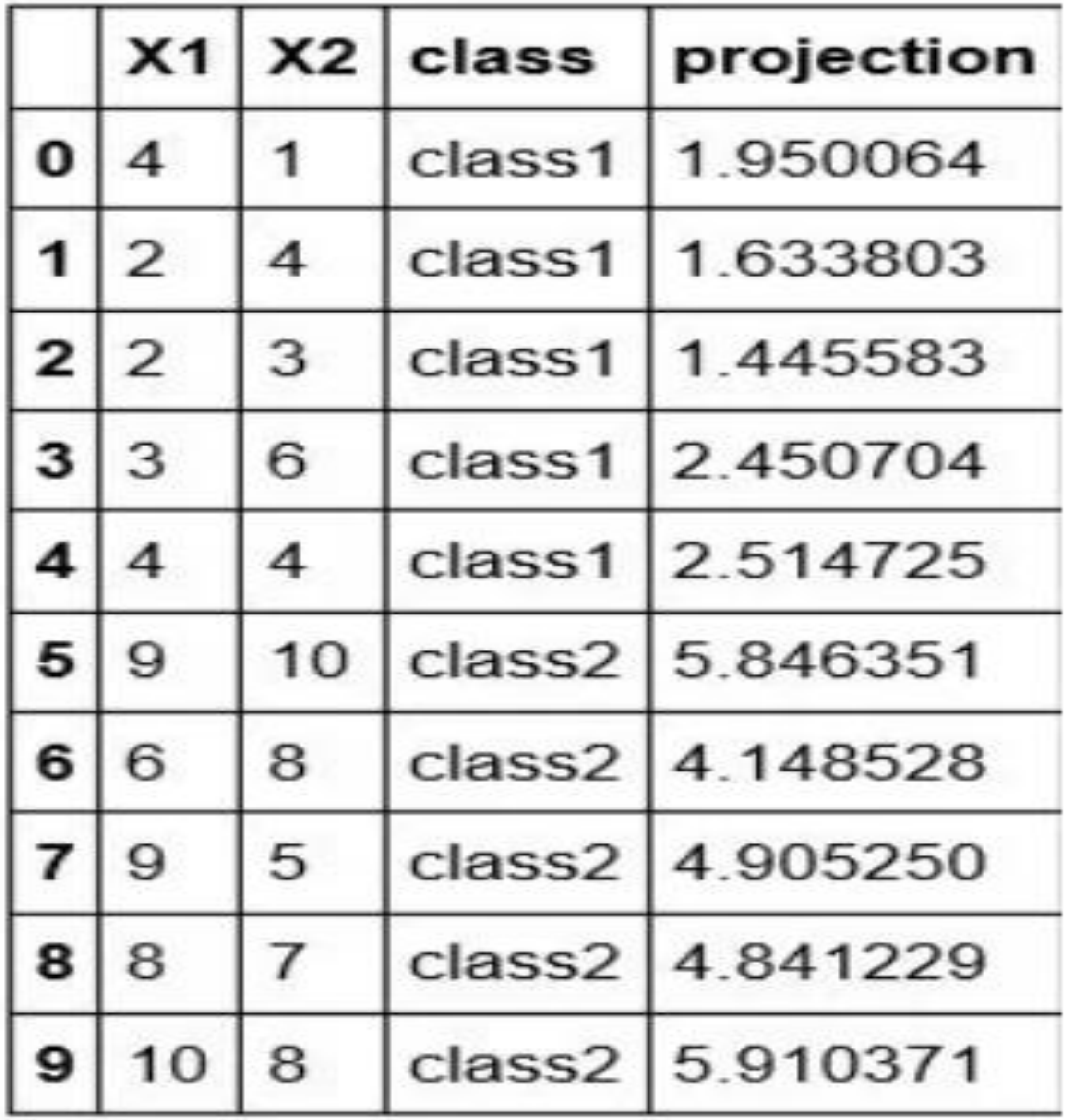
Using Step 3 to find the projection vector W

- *W = np.dot(np.linalg.inv(sw),(mu1-mu2).reshape(2,1))*
- *W*
- *array([[-0.44046095],*
- *[-0.18822023]])*
- *W = -W.T[0] #converting it to positive as it is not going to create much of a difference*
- *W*
- *array([0.44046095, 0.18822023])*

Using Step 4 to find the scalar quantities for each set of input features

- *f=[]*
- *for i in range(len(data.ix[:,:-1])):*
- *f.append(np.dot(W,np.array(data.ix[i,:-1],dtype='float64').reshape(2,1)))*
- *data['projection']=np.array(f)*

The final data along with the projected scalar quantities looks like this

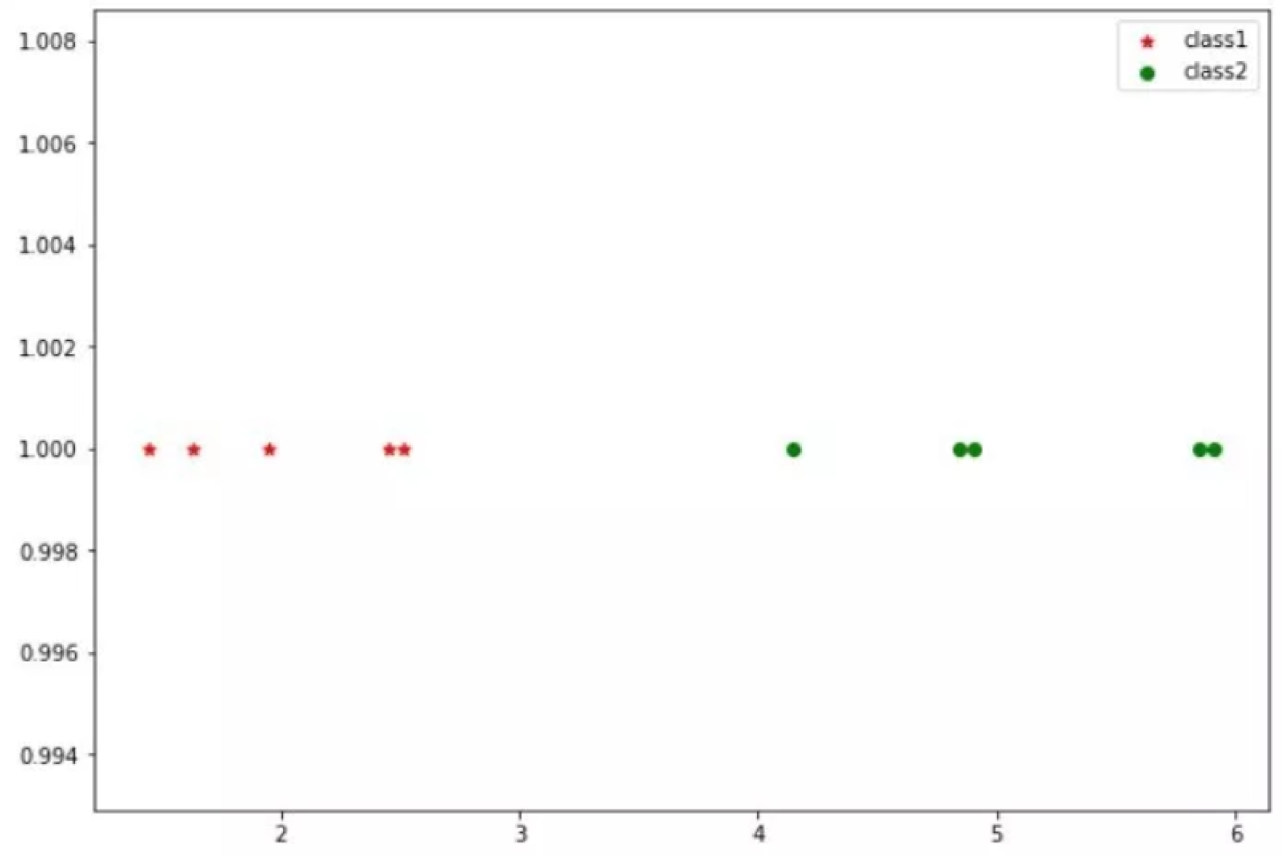


	X1	X2	class	projection
0	4	1	class1	1.950064
1	2	4	class1	1.633803
2	2	3	class1	1.445583
3	3	6	class1	2.450704
4	4	4	class1	2.514725
5	9	10	class2	5.846351
6	6	8	class2	4.148528
7	9	5	class2	4.905250
8	8	7	class2	4.841229
9	10	8	class2	5.910371

Lets plot and see how well they are separated

Code-

- *plt.scatter(data.ix[data['class']=='class1',3],np.array([1,1,1,1,1]),color='r',marker = '*',label='class1')*
- *plt.scatter(data.ix[data['class']=='class2',3],np.array([1,1,1,1,1]),color='g',marker = 'o',label='class2')*
- *plt.legend(loc='best')*
- *<matplotlib.legend.Legend at 0x2090caa7588>*



- Scatter Plot

Questions-

- Using the above plot of the projected scalar quantities we can easily separate the two classes i.e any class with a scalar value less than, say ‘3’ belongs to class 1 and values greater than ‘4’ belongs to class 2.How easily LDA solved our problem of binary classification, but I wish life was that simple.
- There might be some cases that even after applying LDA there is overlapping of the two classes.What to do then? Our model’s accuracy will degrade.Is there a solution?