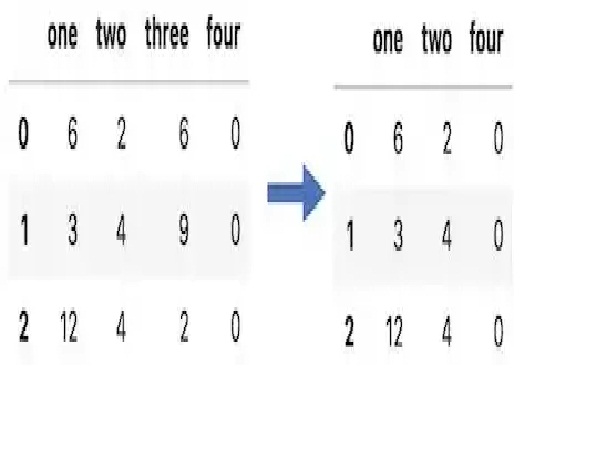
**1. Introduction to Dimensionality Reduction:**

First, let’s start with understanding the simple definition of ‘Dimensionality Reduction’ -

*“It is a process of reducing the number of random variables under the consideration by obtaining a set of principal values”*

Think as you have many columns in the dataset and many columns are just redundant or not useful in capturing the information that is required in our objective. We remove such features and by removing it we actually reduce the dimensions of our data.



Columns 3 is removed.

There are two main components in dimensionality reduction-

1. **Feature Selection:** In feature selection, we try to find the best subset of features that are required to fulfil our objective.
2. **Feature Extraction:**In feature extraction, we create new features based on transformation or combinations of the original feature set.

***Note:****Both feature selection and extraction leads to dimensionality reduction. No evidence that one of them is superior to other on all types of task.*

There are situations when we deal with high dimensional data (Think as thousands of rows and millions of columns) in machine learning. In such a situation, our machine learning algorithm may not work properly. This situation broadly referred to as **“ The Curse of Dimensionality.”**

Since we know that as dimensionality increases, the number of data points to perform good classification/regression/clustering etc. machine learning task increases exponentially. There are some cases where the number of rows (n) of the dataset is fixed and performance decreases as the dimensionality increases. This phenomenon is known as **‘Hughes Phenomena’.**

**How do we check whether data is high dimensional or not?**

There is no proper measure that can be deployed to check the high dimensionality of data. Since in machine learning a lot of things we decide based on some thumb rule. Here also we will use one thumb rule to decide whether data is high dimensional or not.

**Thumb Rule:** If data is greater than**2^(No of Columns)**will be considered as high dimensional data.

Now you may ask how do we get rid of this curse? It is very easy and that’s too objective — By reducing the dimensions. There are many methods/algorithms are available to reduce high dimensionality from a dataset. Some of them are mentioned below-

1. Principal Component Analysis (PCA)
2. Linear Discriminant Analysis (LDA)
3. Generalized Discriminant Analysis (GDA)

Dimensionality reduction may be linear or non-linear depending upon the method used. We will be discussing PCA in the series of lecture.

**2. Prerequisite for Principal Component Analysis (PCA) :**

Principal Component Analysis (PCA) is nothing but the dimension reduction technique used to reduce the dimensions by forming linear combinations of features which retain maximum information/variance of data.

This is slightly confusing right now for total beginners to understand since concept like retaining information (variance) is mentioned above. Don’t worry, if you do not get it at this moment we will revisit it once again once you learn all prerequisites.

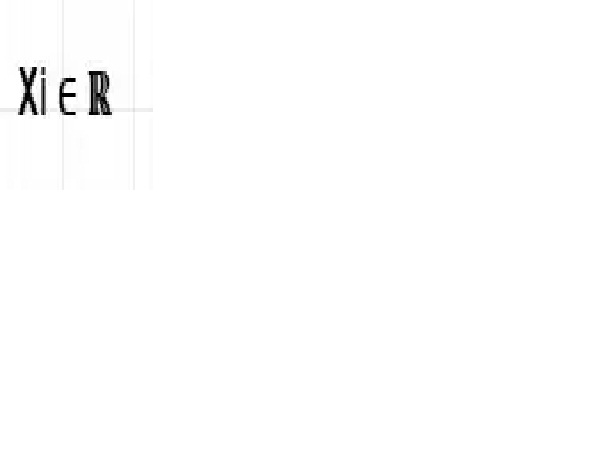
We will visit the below-mentioned concepts that required to understand PCA thoroughly —

1. Vector (Row & Column)
2. How to represent dataset as a matrix?
3. Mean
4. Variance
5. Standardization of dataset
6. Covariance & Covariance Matrix
7. Eigen Value & Eigen Vectors

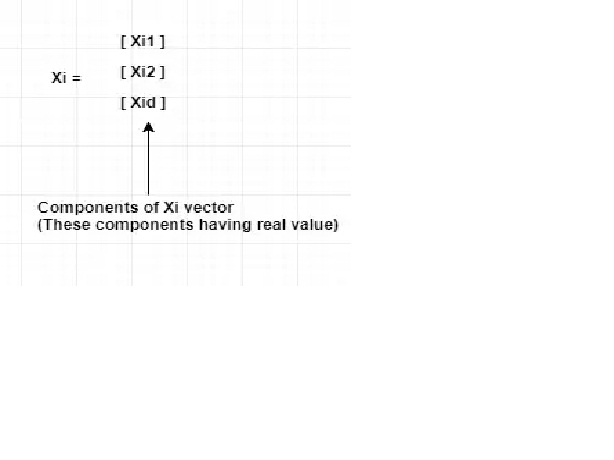
Let’s visit them one-by-one.

**1. Row Vector & Column Vector:**

In linear algebra the data points that are real-valued represented as



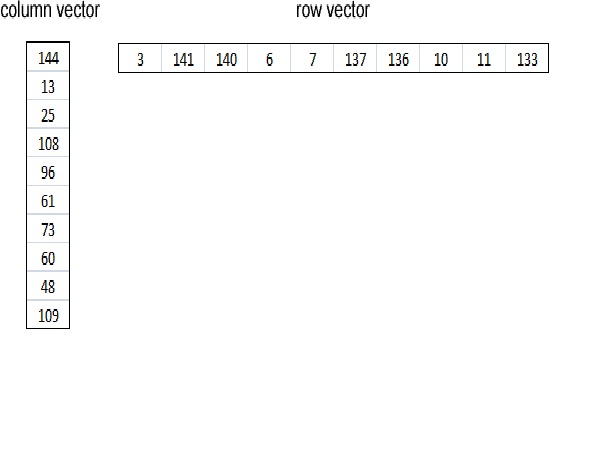
(ith data point)



Each column is considered a vector is known as a column vector. In the above figure, it is clear that **Xi**is the column vector. (No information is available by default is column vector)

The transpose of a column vector is known as a row vector. Each row in a dataset is considered as a row vector. In the above example, if you take the transpose then we will get a row vector.

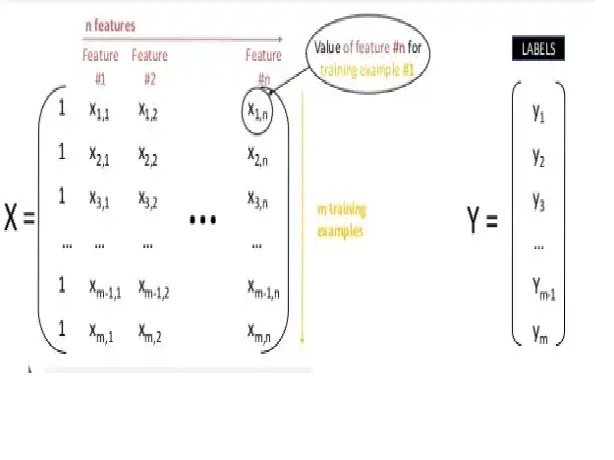
**Example:**



A column vector is on the left side, and a row vector is shown on the right side.

**2. How to represent dataset as a matrix:**

Each row in the dataset is considered as a data point and each column in the dataset is considered as features.



**3. Mean:**

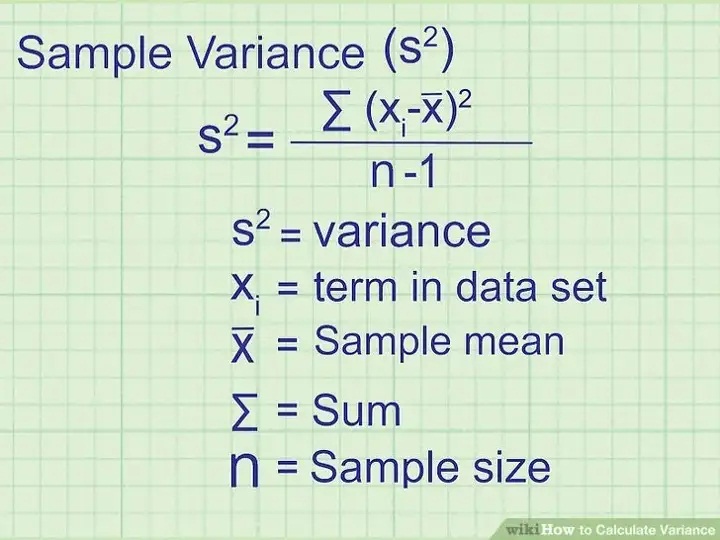
The mean is the average of the numbers. It is very easy to calculate: add up all the numbers, then divide how many numbers there are.

**4. Variance & Standard Deviation:**

The variance is defined as-

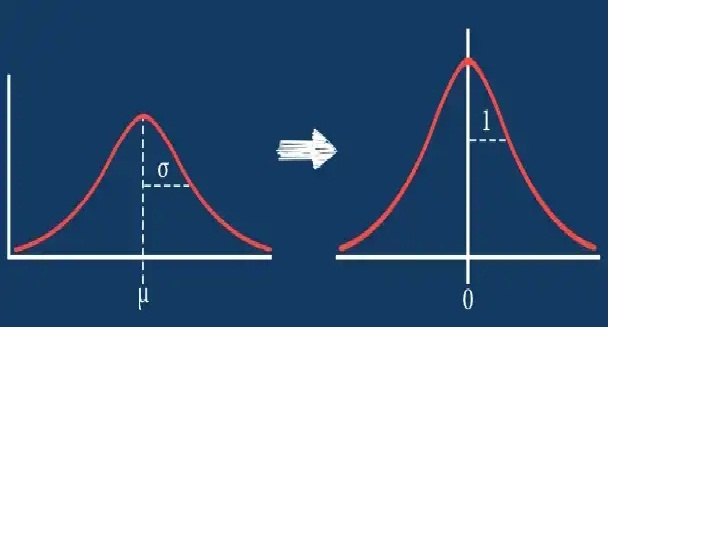
*“The average of the squared differences from the mean”*

The variance tells us how much your data has spread out from the mean of the dataset.

Once we have variance we can easily calculate a standard deviation once we have variance. Standard deviation is nothing but the square root of the variance.

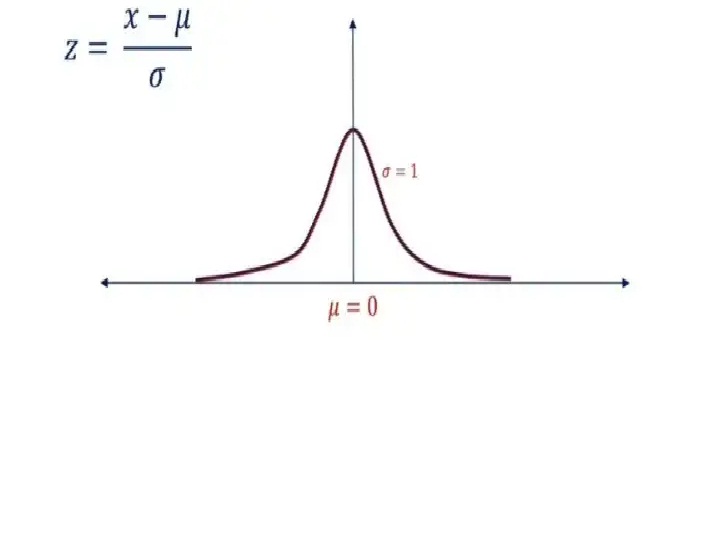
**5. Standardization of Dataset:**

Sometimes many features come with different scales. It means these variables or feature do not give an equal contribution to the analysis. Let’s say you’ve two variables height(cm) of the student and weight (KG). Since both the variable are having different scales cause a serious issue in our analysis. When we standardize variable we make sure that the mean of the feature is 0 and the standard deviation is equal to 1. By doing this we can get rid of scales.



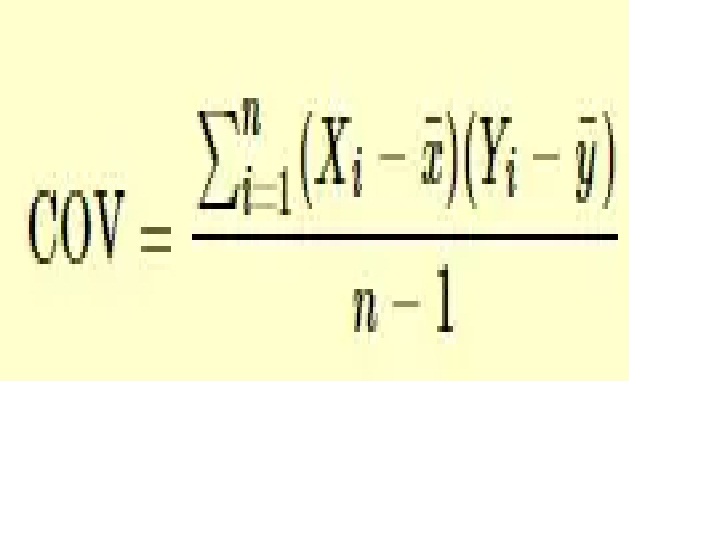
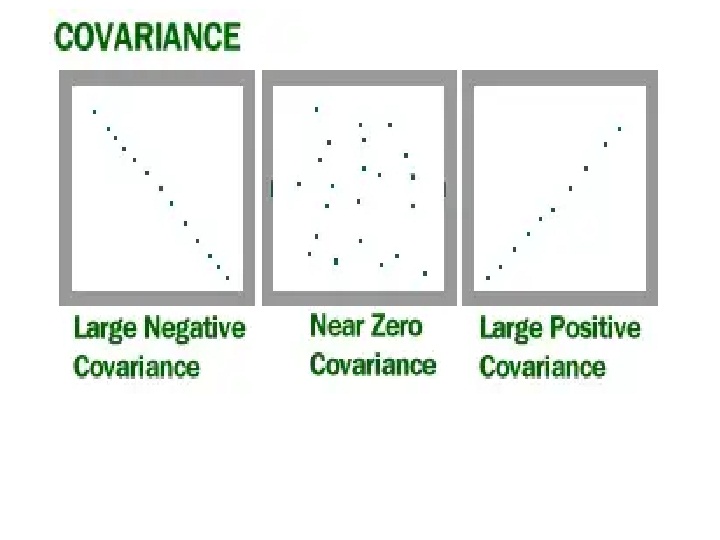
**How do we standardize our data?**

Z-score standardization is one of the most popular methods to normalize data. In this case, we rescale an original variable to have a **mean of zero** and a **standard deviation of one.**



**6. Covariance & Covariance Matrix:**

Covariance is one of a family statistical measures used to analyse the linear relationship between two variables. **How do two variables behave as a pair?**Covariance can take any value between **-Inf to +Inf.**A positive value indicates an increasing linear relationship and a negative value indicates a decreasing relationship. Covariance near to **ZERO**indicates no relationship between the variables.

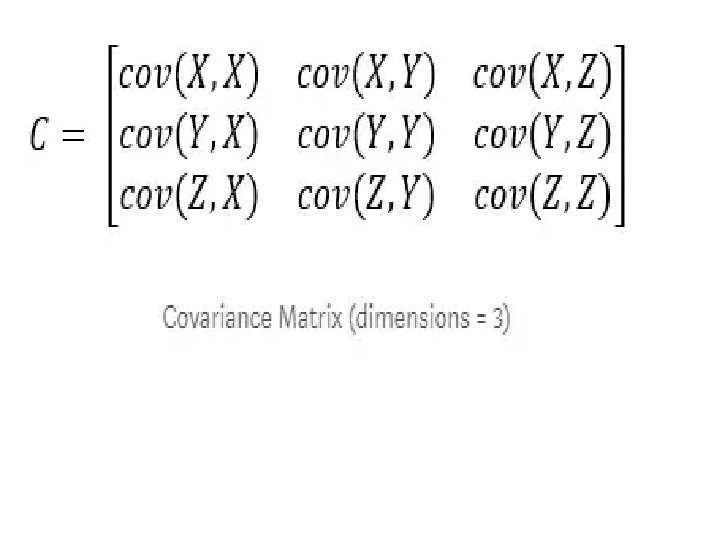


How to calculate COV(X, Y)

***Note:****Covariance does not tell you about ‘Strength of relationship’. It just tells you whether there is a positive or negative linear relationship exists or not.*

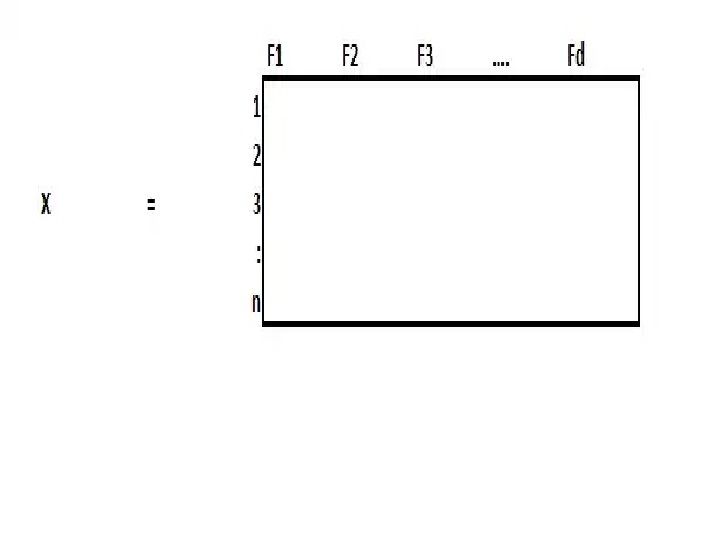
These all the information that you require to understand Covariance. Now let’s move towards Covariance Matrix and see what is it?

A covariance matrix is nothing but the information about auto-covariance (Covariance with itself) and Covariance between two different variables presented in matrix format as shown below-

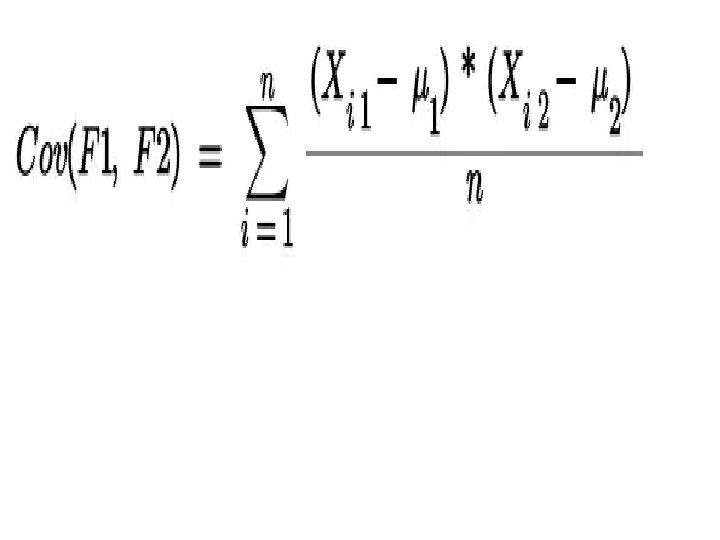


If you observe the above matrix carefully you can see all diagonal elements are nothing but the covariance with itself. These are also called variance of the variable. Thus we can say,

**Cov(X,X) = Var(X) | Cov(Y,Y) = Var(Y) | Cov(Z,Z) = Var(Z)**



Let’s calculate the covariance of two columns COV(F1, F2) to understand the concept that we discussed above in a better way.

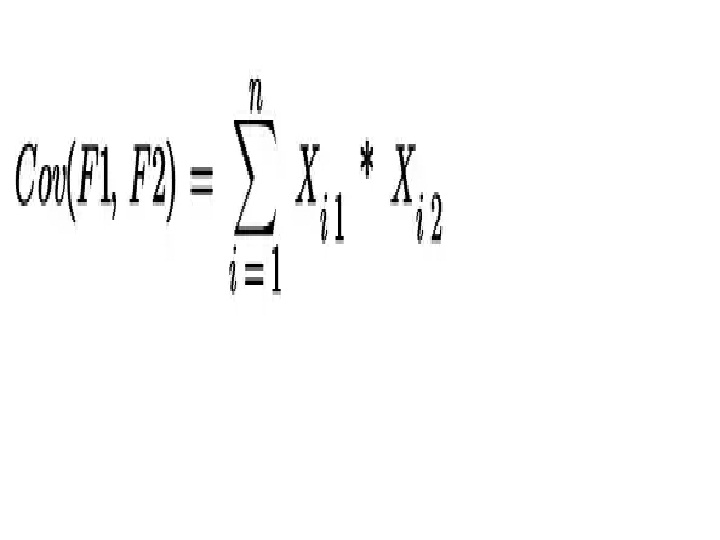


Covariance Between 2 Features

where,

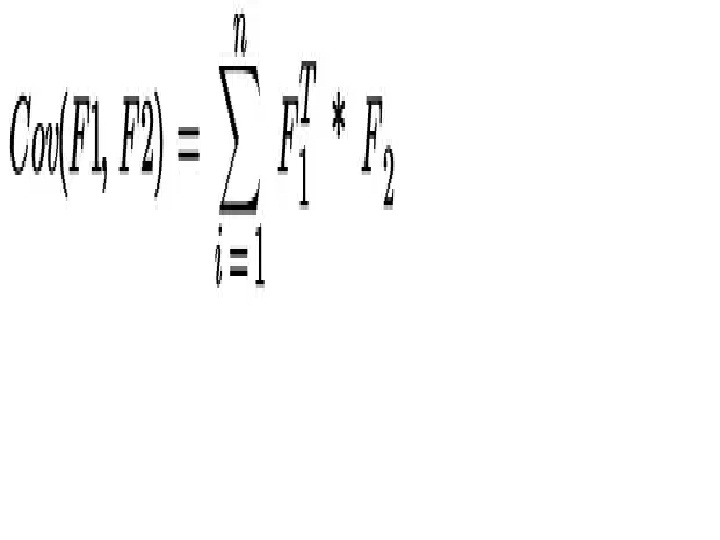
Xi1 : Data points of F1 | Xi2 : Data points of F2 | μ1 & μ2: Mean of the respective columns.

Lets assume data is standardized which means **μ = 0 & σ = 1**. Therefore the above equation becomes-

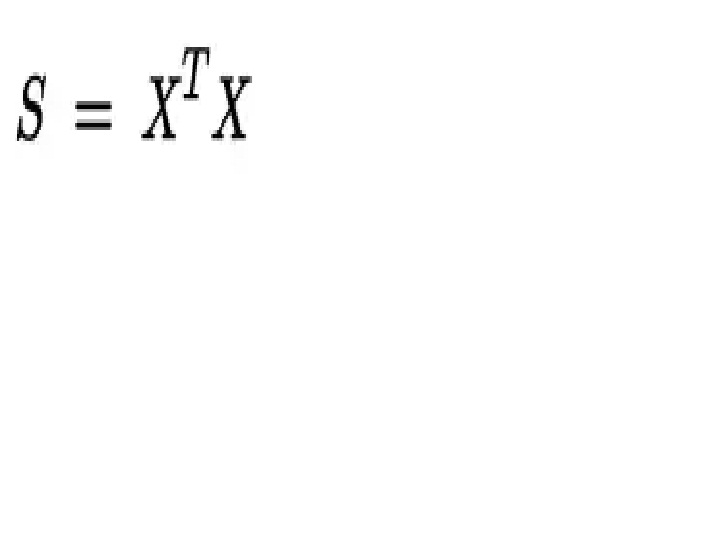


Covariance when **μ = 0 & σ = 1**

Therefore,



Thus we can generalize it like below



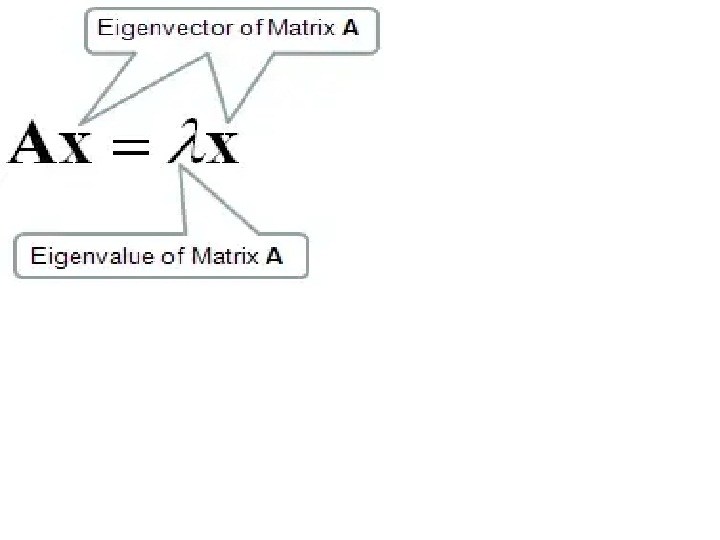
Covariance Matrix

where S = Covariance Matrix | X = Datapoints

**7. Eigen Value & Eigen Vector**

The word ‘Eigen’ is taken from ‘German’ which means ‘Self or own’. It describes a special relationship between two things. For example, “mein eigenes Auto” in German means “My Own Car”. Here eigen is explaining the relationship between ‘car and my own’. Thus we can think of the same relationship of eigenvectors with a square matrix. In the world of linear algebra, all the square matrix (for example, 2x2 or 4x4)have eigenvectors associated with them.

Let A be the square matrix of order (n x n), then a number (Real or Complex) **λ**is said to be the eigenvalue of matrix **A** if there exists a column matrix **x** of order (n x 1) such that -



Therefore,

**Ax -λx = 0**

**(A-λI)x = 0**

where I = Identity matrix.

In order to find eigenvalues and eigenvectors first, we need to convert the above equation to a characteristic equation -

**|** **A-λI | = 0 …..(Characteristic Equation)**

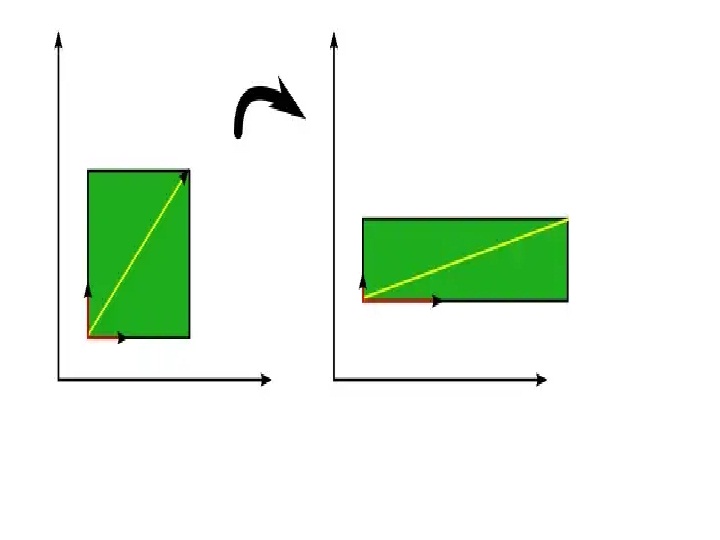
Here basically we are subtracting **λ**value from the diagonal of the square matrix **A**then we are taking the determinant of the resulting matrix. Thus the solution to the characteristic equation will give you eigenvalues.

***Note:****Total number of eigenvalues is same as an order of square matrix. Let say we have 2 x 2 data matrix then there will be 2 eigenvalues. And we find eigenvectors corrosponding to eigenvalues.*

Once we get **λ**value by using the above equation then put the value of **λ**in the equation **(A-λI)x = 0**to get the corresponding eigenvector.

**Important Pointers:-**

1. If you multiply matrix A with vector X then we will get another vector where we can say matrix performed linear transformation on an input vector
2. An eigenvector is a vector whose direction remains unchanged when a linear transformation is applied to it.



Eigenvectors (red) do not change direction when a linear transformation (e.g. scaling) is applied to them. Other vectors (yellow) do.

The green square is only drawn to illustrate the linear transformation that is applied to each of these three vectors.

3. Here square matrix is nothing but our calculated covariance matrix and we are using that to find eigenvalues and eigenvectors. Let **S** be the covariance matrix then corresponding eigenvalue and eigenvectors are -

Eigenvalues(S) = **λ 1≥ λ2 ≥ λ3 ≥ … ≥ λd**

Eigenvectors(S)= **V1 ≥ V2 ≥ V3 ≥ … ≥ Vd**

Where **λ1** = Maximal eigenvalue (means maximum variance) & **V1** is an eigenvector corresponding to maximal eigenvalues

Thus we can say,

**Vi ⊥ Vj = Vi \* Vj = ||Vi|| \* ||Vj||\* cos ϴ**

Since **Vi ⊥ Vj**then **cos ϴ = cos 90 = 0**

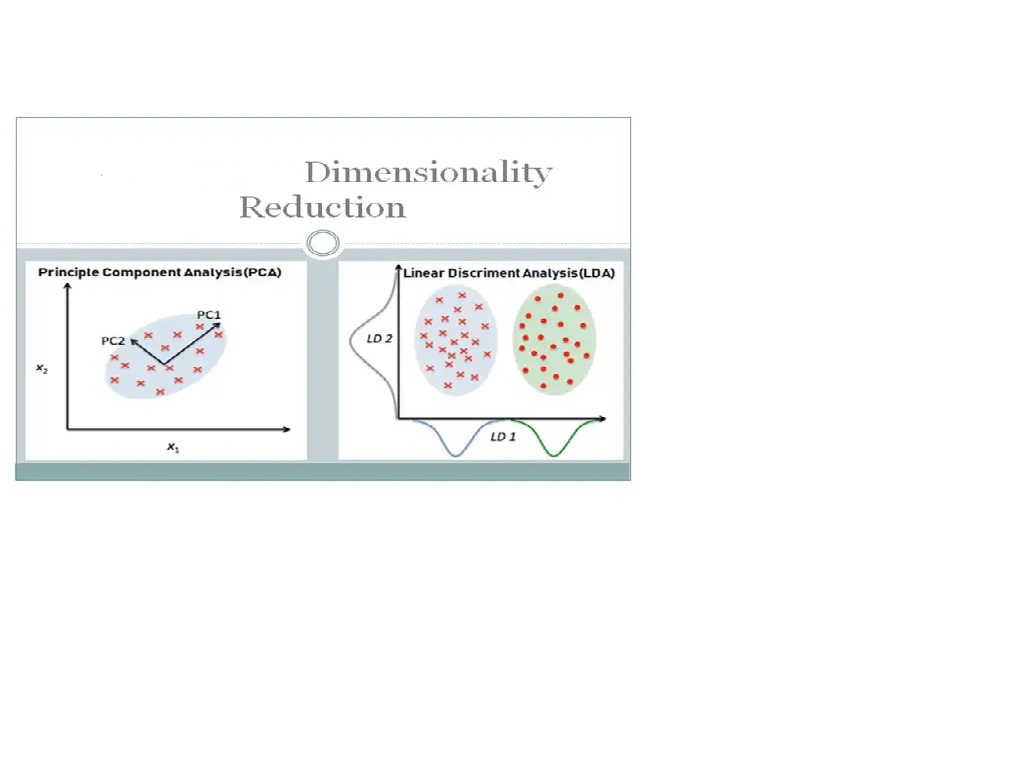
**Vi ⊥ Vj = Transpose(Vi) \* Vj = 0**

Therefore, every pair of the eigenvector is **⊥**to each other.

4. The eigenvector is a vector with the direction of maximum variance and eigenvalue is the number (scalar) which represent maximum variance.

https://medium.com/machine-learning-researcher/dimensionality-reduction-pca-and-lda-6be91734f567

# Dimensionality Reduction(PCA and LDA) with Practical Implementation



In this chapter, we will discuss Dimensionality Reduction Algorithms (Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA)).

This chapter spans 5 parts:

1. What is Dimensionality Reduction?
2. How the Principal Component Analysis(PCA) Work?
3. How the Linear Discriminant Analysis (LDA) Work?
4. Practical Implementation of Principle Component Analysis(PCA).
5. Practical Implementation of Linear Discriminant Analysis (LDA).

# 1. What is Dimensionality Reduction?

In Machine Learning and Statistic, Dimensionality Reduction the process of reducing the number of random variables under consideration via obtaining a set of principal variables. It can be divided into feature selection and feature extraction.

**We will deal with two main algorithms in Dimensionality Reduction**

1. Principle Component Analysis (PCA)
2. Linear Discriminant Analysis (LDA)

# 2: How Dimensionality Reduction Algorithms Work?

## 2.1: Principal Component Analysis (PCA).

## 2.1.1: What is Principle Component Analysis (PCA)?

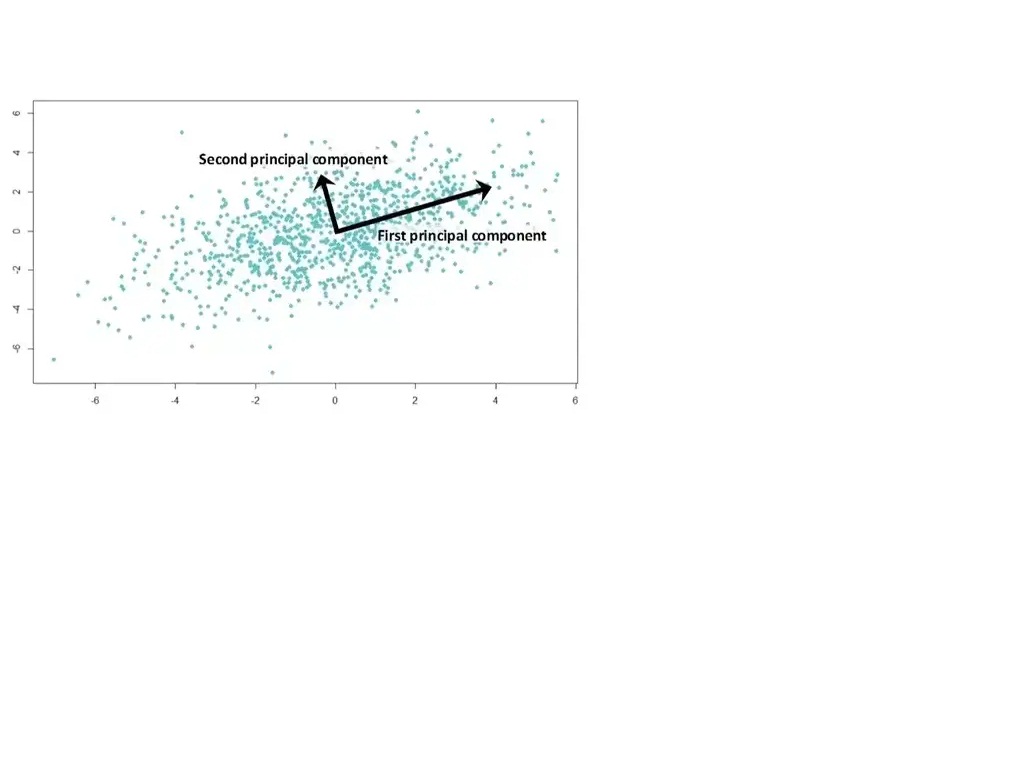
If you’ have worked with a lot of variables before, you know this can present problems. Do you understand the relationship between each variable? Do you have so many variables that you are in danger of overwriting your model to your data or that you might be violating the assumptions of whichever modeling tactic you’re using?

You might ask the question “how do I take all of the variables. I’ve collected and focused on only a few of them? In technical terms, you want to “reduce the dimension of your feature space. By reducing the dimension of your feature space, you have fewer relationships between variables to consider and less likely to overheat your model.

Somewhat unsurprisingly, reducing the dimension of the feature space is called “dimensionality reduction” There are many ways to achieve dimensionality reduction, but most of the techniques fall into one of two classes.

· Feature Elimination

· Feature extraction



**Feature Elimination:**we reduce the feature space by elimination feature. The advantages of the feature elimination method include simplicity and maintainability features. We’ve also eliminated any benefits those dropped variables would bring.

**Feature Extraction:**PCA is a technique for feature extraction. So it combines our input variables in a specific way, then we can drop the “least important” variables while still retaining the most valuable parts of all the variables.

**When should I use PCA?**

1. Do you want to reduce the no. of variables, but are not able to identify variables to completely remove from consideration?

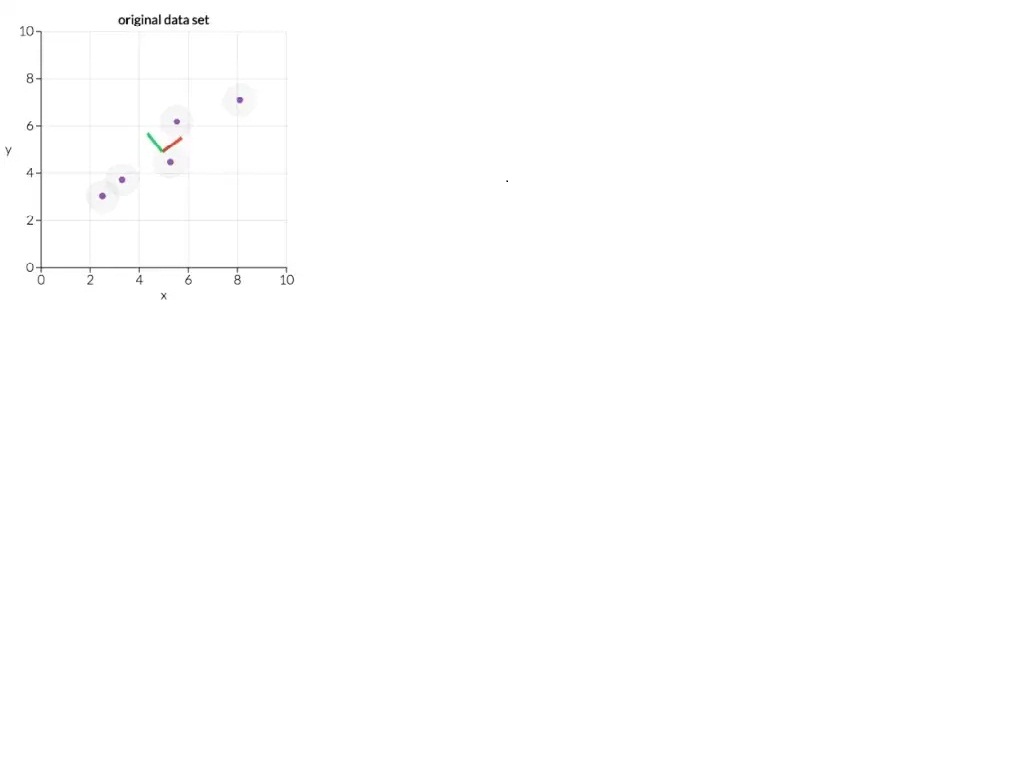
2. Do you want to ensure your variables are independent of one another?

3. Are you comfortable making your independent variable less interpretable?

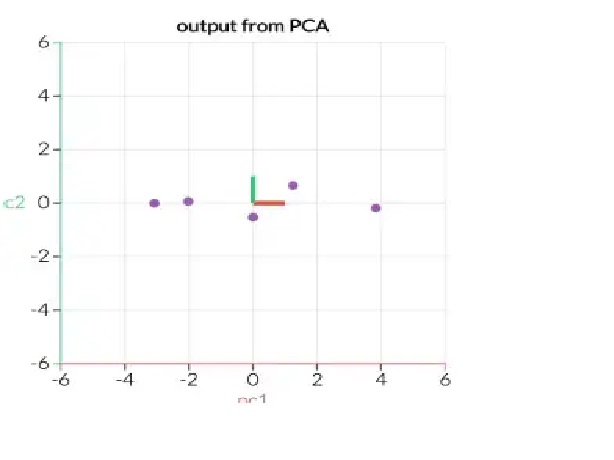
## 2.1.2: How Principle Component Analysis (PCA) work?

We are going to calculate a matrix that summarizes how our variables all relate to one another.

We’ll then break this matrix down into two separate components: direction and magnitude. we can then understand the direction of our data and its magnitude.



The above picture displays the two main directions in this data: the back direction and the blue direction. In red direction is the most important one. We’ll get into why this is the case later, but given how the case later, but given how the dots are arranged can you see why the red direction looks more important than the green direction (What would be fitting a line of best fit to the data look like?



In the above pic, we will transform our original data into aligning with these important directions. The fig. The show is the same extract data as above but transformed. So that the X- & Y axis are now direction.

What would the line of best fit look like here:

1. Calculate the covariance matrix X of data points.

2. Calculate eigenvectors and correspond eigenvalues.

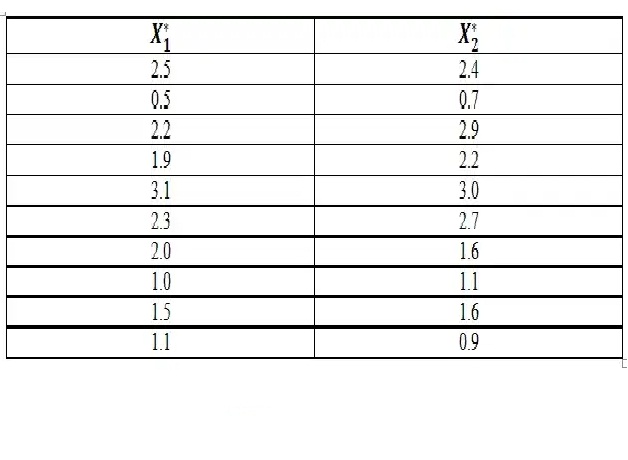
3. Sort eigenvectors accordingly to their given value in decrease order.

4. Choose first k eigenvectors and that will be the new k dimensions.

5. Transform the original n-dimensional data points into k\_dimensions

**Let’s dive into mathematics:**

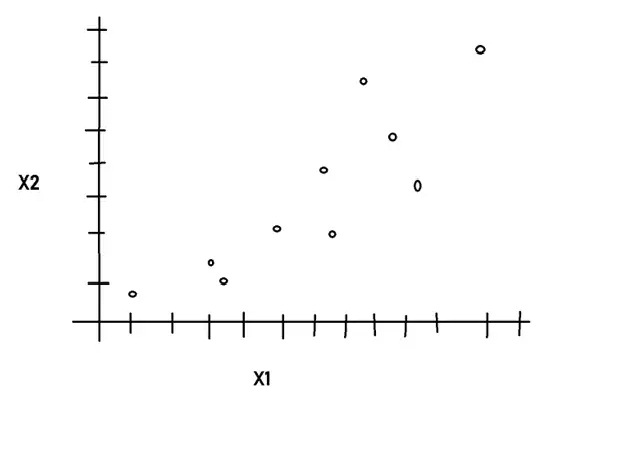
**Dataset:**



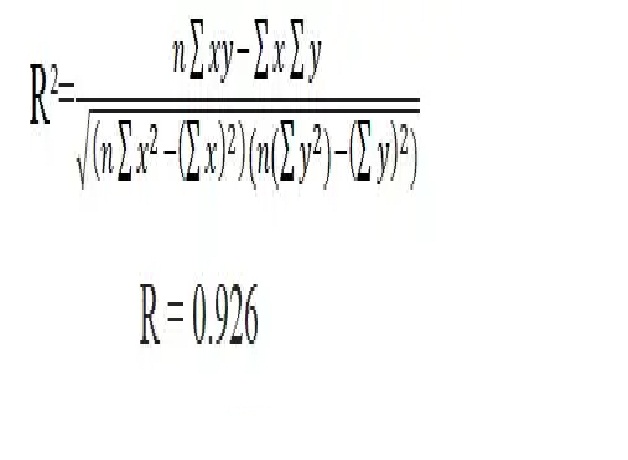
**Sample size n = 10**

**Variables p = 2**

Construct a scatter plot to see how the data is distributed.

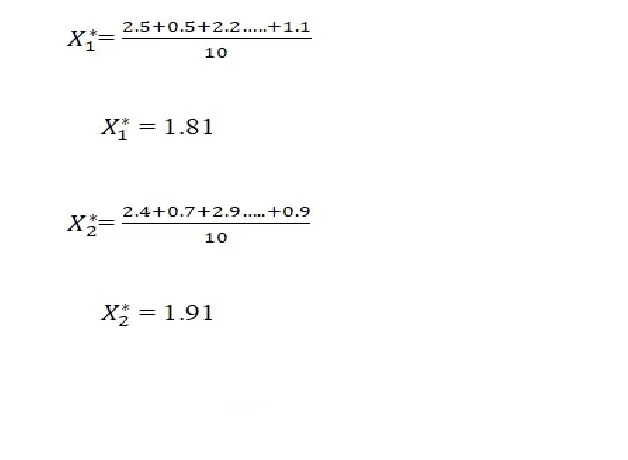


So Correlation

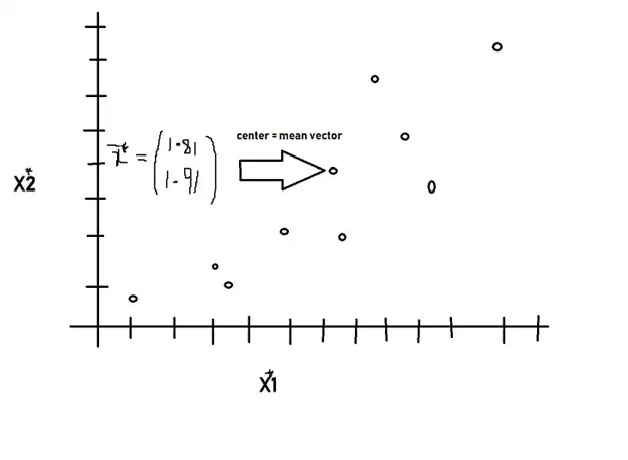
****

Positive correlation high redundancy

**Mean of our variables**



Now



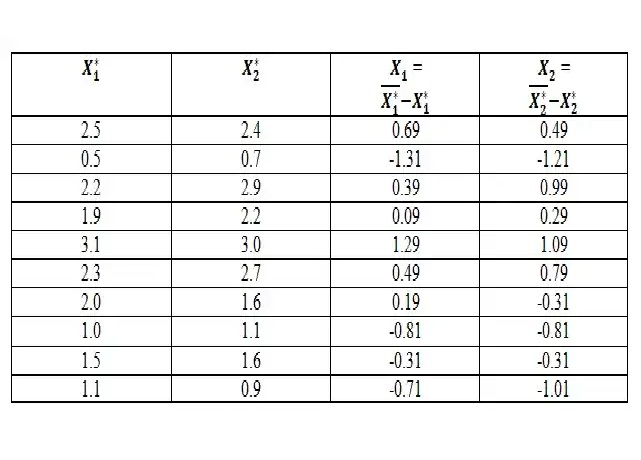
**Step 1:**

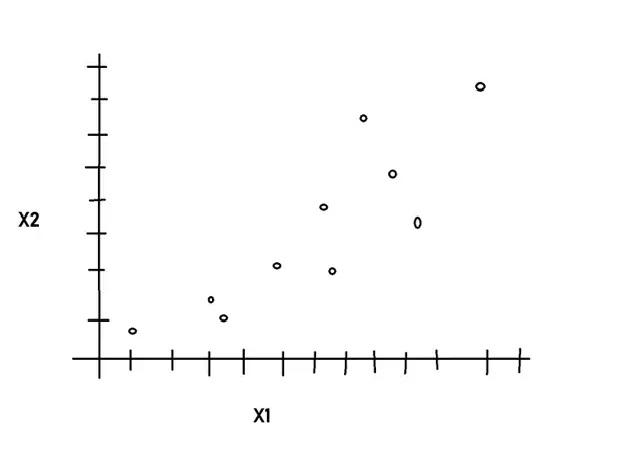
· Subtract the mean from the corresponding data component to recentre the dataset.

· Reconstruct the scatter plot to view.

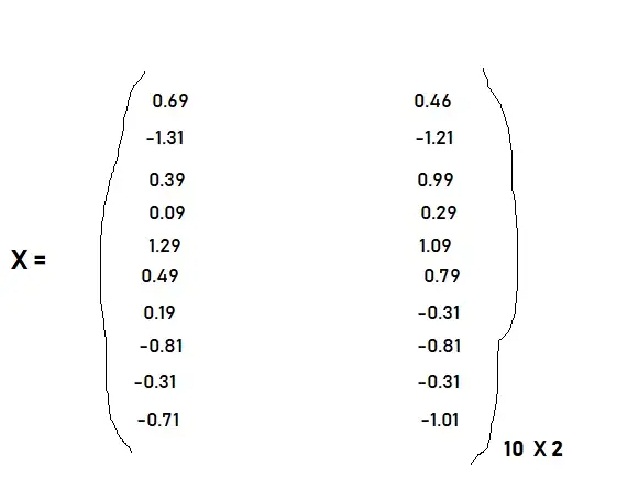
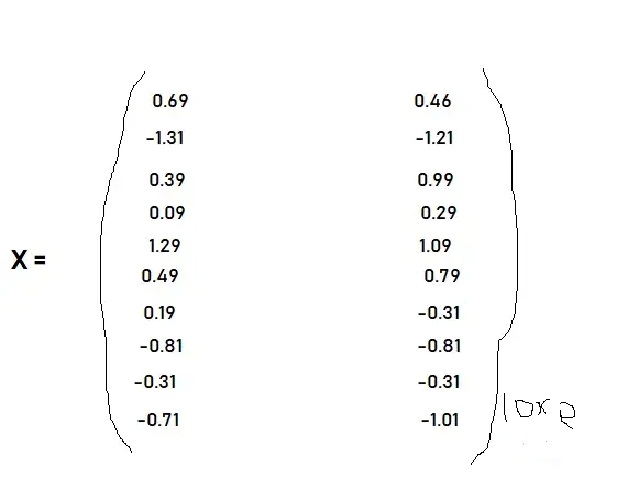
· Write the “adjusted” data as a matrix X.

Note: that “adjusted” data set will have mean zero.

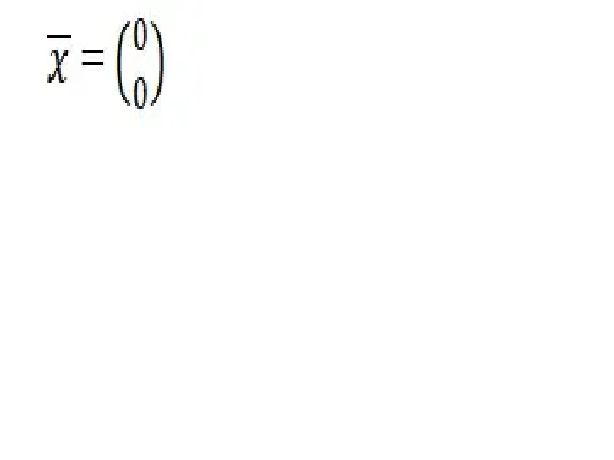




Now write the “adjusted” data as a Matrix X.

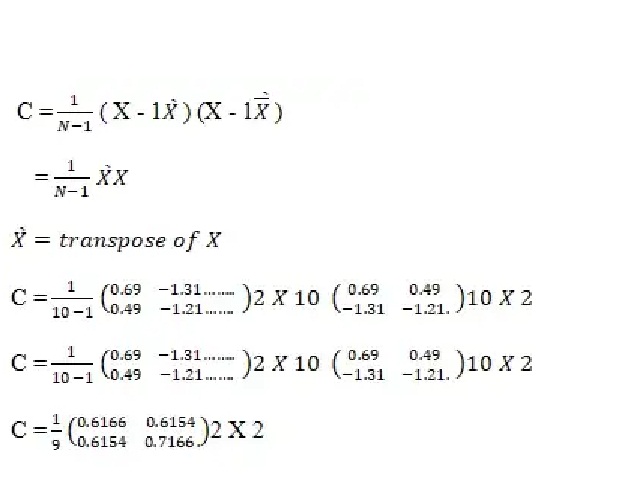


Note: that the adjusted “dataset will have means zero.



Step 2:

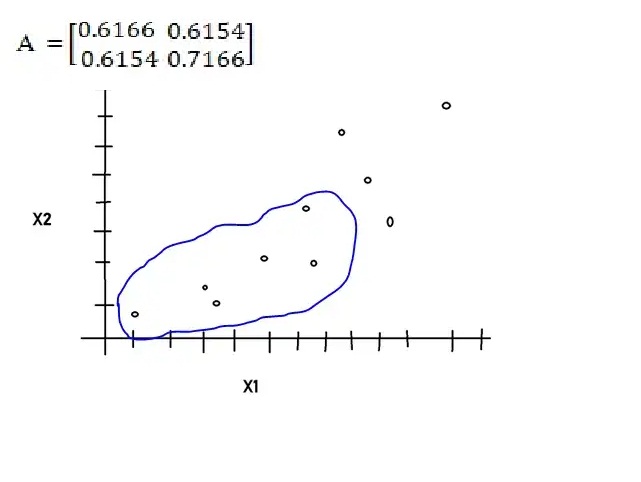
Compute the sample variance-covariance matrix C.

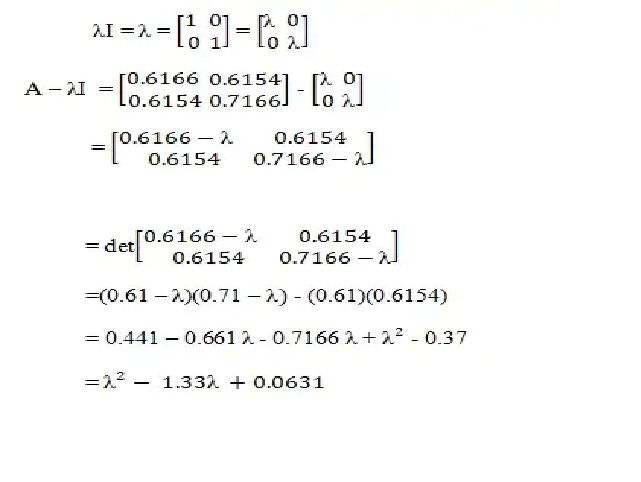


Step 3:

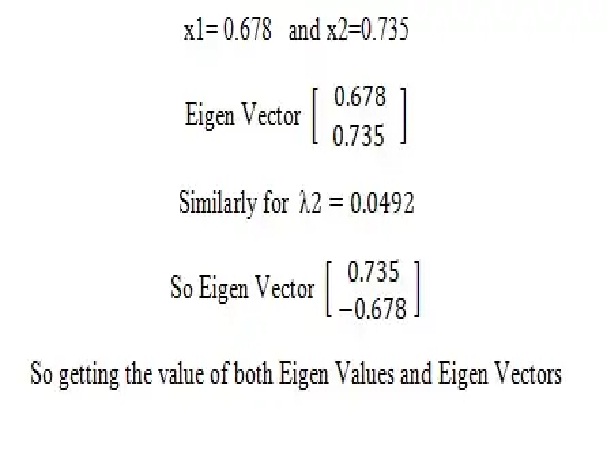
Compute the eigenvalues lambda 1 and eigenvalue

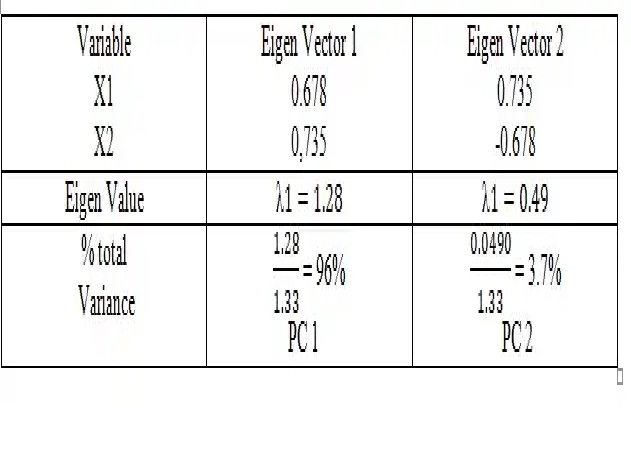
Of C order the corresponding pairs from the highest to the lowest eigenvalues.



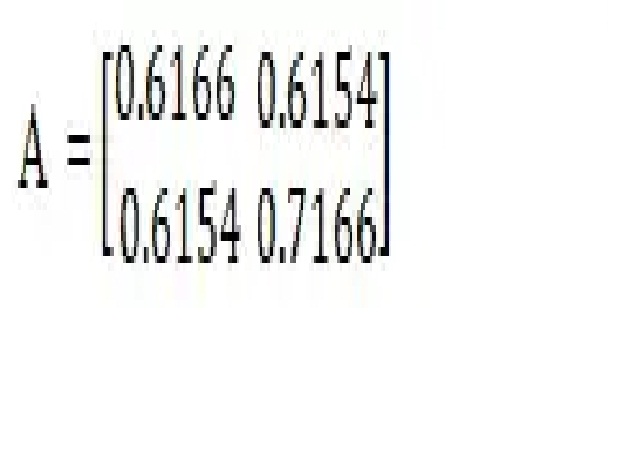


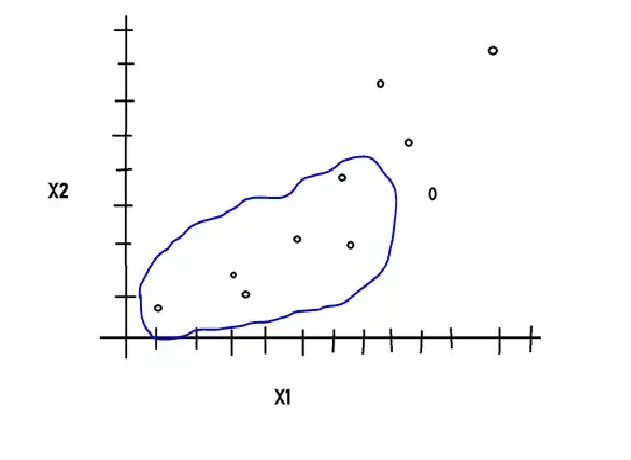
After solving that’s matrix we get the value of Eigen Vector



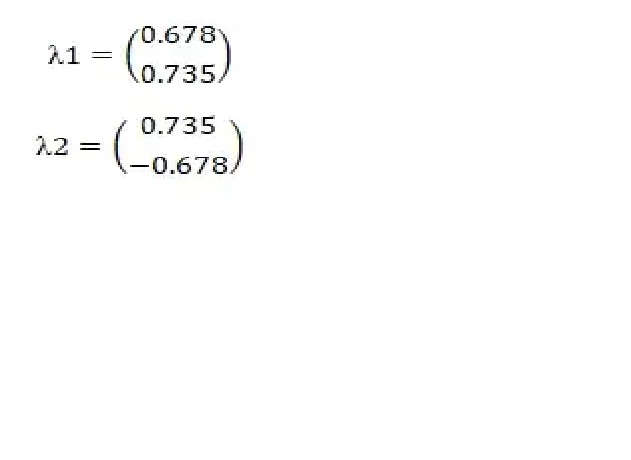


So finally



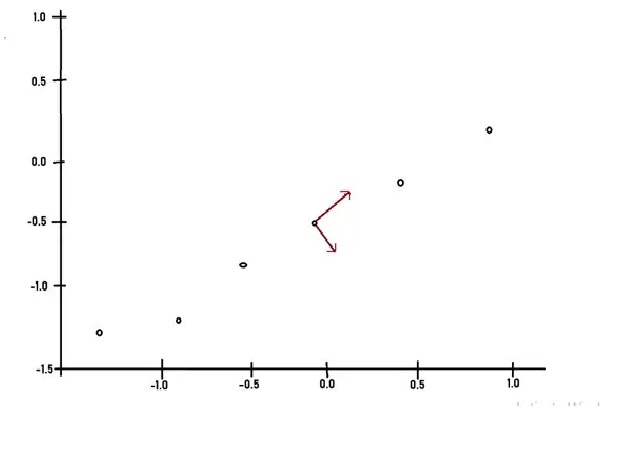


Now Eigen Vector



In Eigen Vector1 move right direction and 0.735 directions are up

In Eigen Vector2 move right direction and -0.678 directions are up



It can be proven

Total Sample Variance = Sum of Eigen Value

= 1.28 +0.0490

=1.33

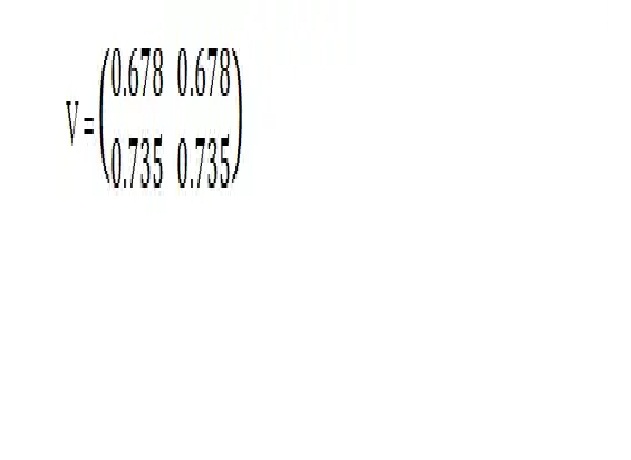
By this process, we will be able to exact lines that characterize the data. The first eigenvector will go through the middle of the data points as if it is the lines of best fit.

The Second eigenvector will give us the other less important pattern in the data. That is all the data points follow the mainline. But are the off to the side of the mainline by some amount?

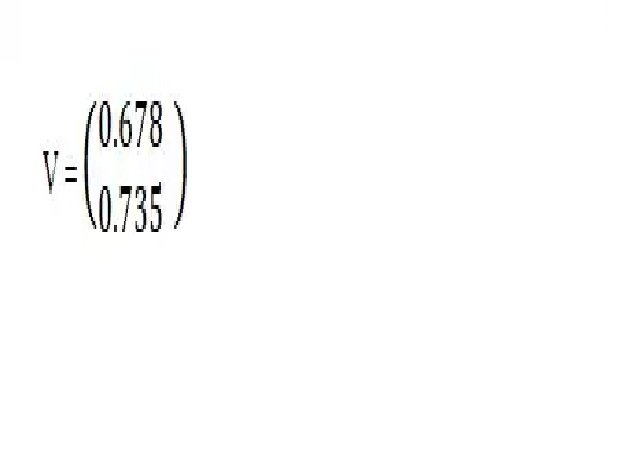
Step 4:

Choose the components and form the eigenvector matrix V. By ordering the eigenvectors according to the eigenvalue, this gives the components in order of their Significance. Hence the eigenvector with the highest eigenvalue is the principal component. The components of lesser significance can be ignored. To reduce the dimensions of the data set.

Select both components then



So discard the less Significant component (we take PC1 which is 96% information capture)

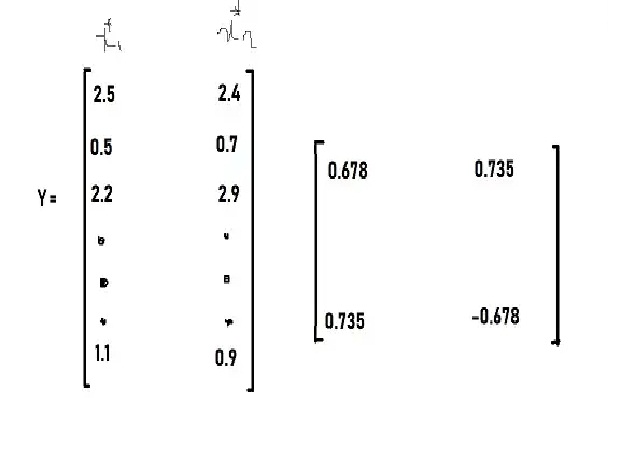


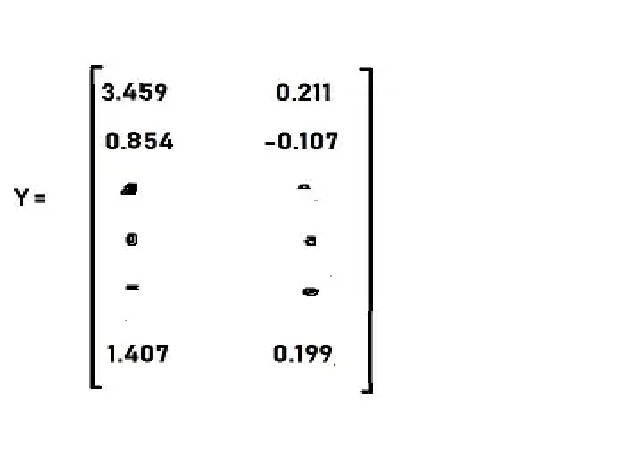
Step 5:

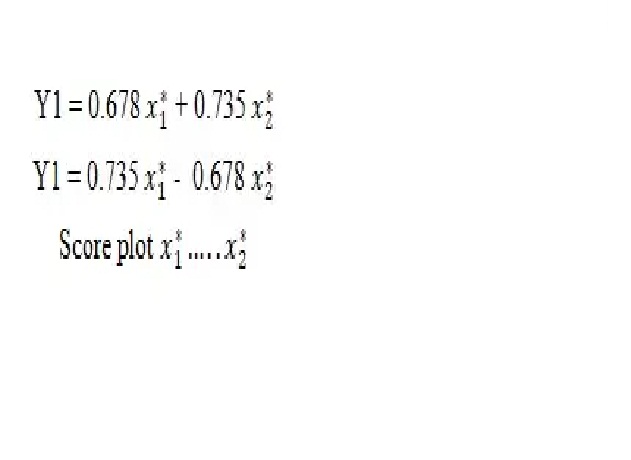
Divide the new data set by taking

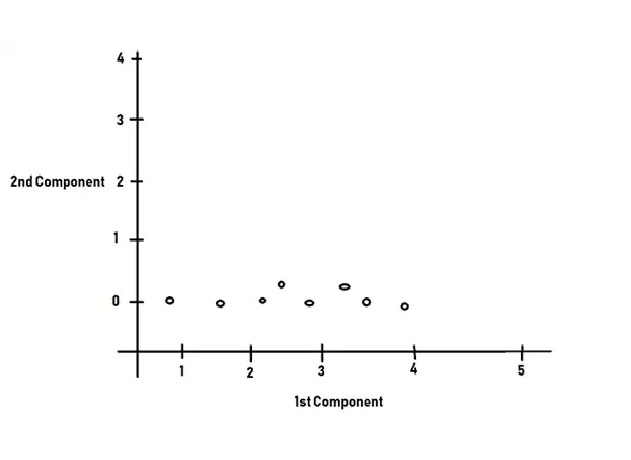
Y = XV

We have transformed our data. So that it is expressed in terms of the pattern between them, where the patterns are the lines that most closely describe the relationship between data.

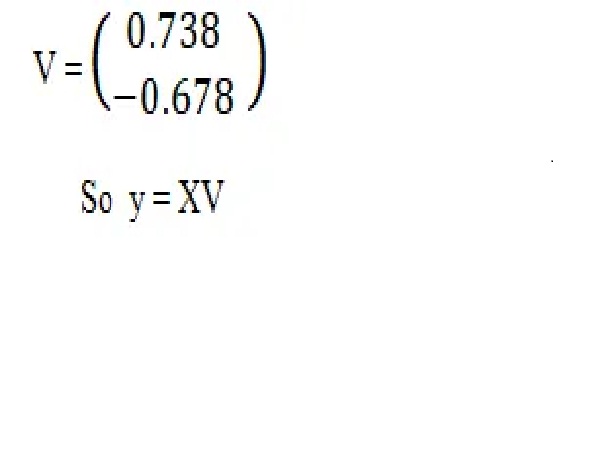


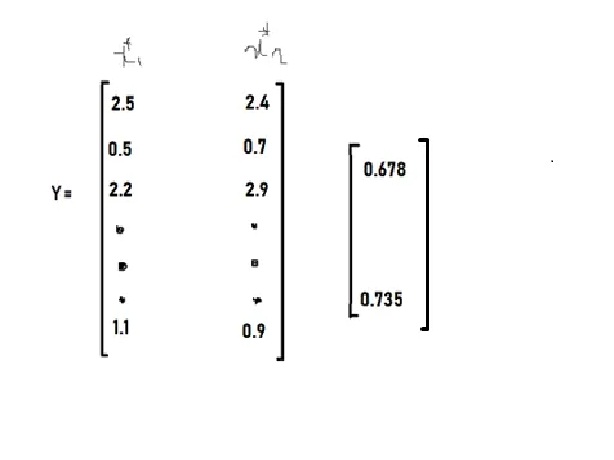


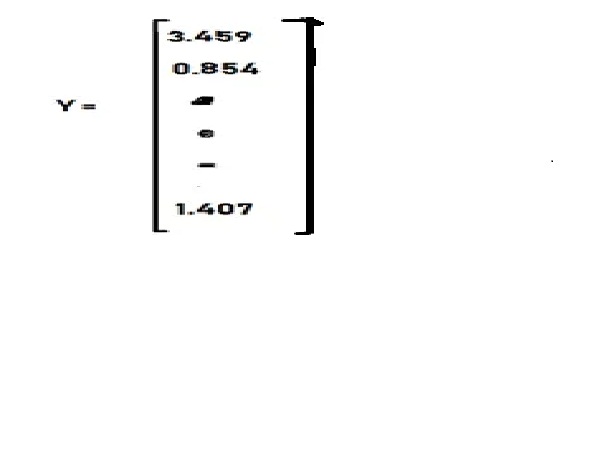




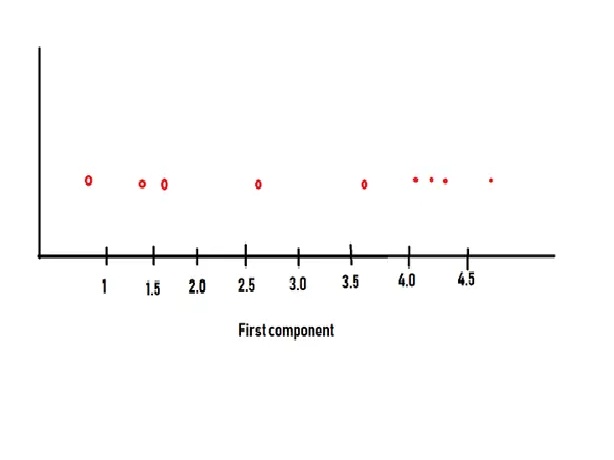
Now again discard the significant component



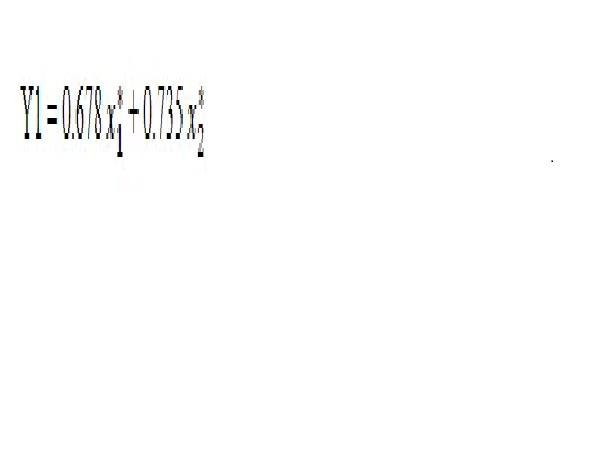




Now



In this case, PCA reduces one dimension.



In this way, PCA works.

## 2.2: Linear Discriminant Analysis (LDA).

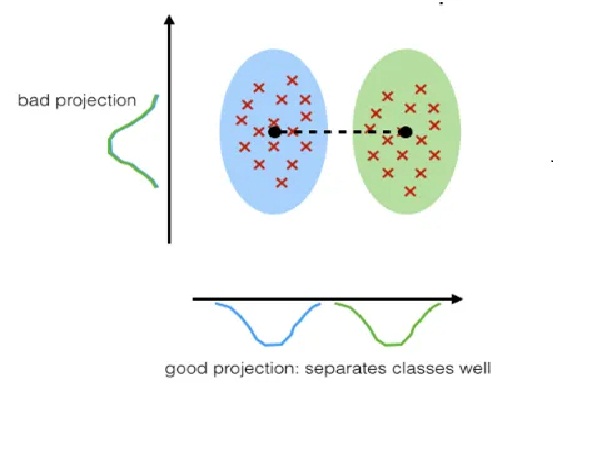
## 2.2.1: What is Linear Discriminant Analysis (LDA)?

LDA is a type of Linear combination, a mathematical process using various data items and applying a function to that site to separately analyze multiple classes of objects or items.

Following Fisher’s Linear discriminant, linear discriminant analysis can be useful in areas like image recognition and predictive analysis in marketing.

The fundamental idea of linear combinations goes back as far as the 1960s with the Altman Z-scores for bankruptcy and other predictive constructs. Now LDA helps in preventative data for more than two classes, when Logistics Regression is not sufficient. The linear Discriminant analysis takes the mean value for each class and considers variants to make predictions assuming a Gaussian distribution.

Maximizing the component axes for class-separation.



## 2.2.2: How the Linear Discriminant Analysis (LDA) work?

First general steps for performing a Linear Discriminant Analysis

1. Compute the d-dimensional mean vector for the different classes from the dataset.

2. Compute the Scatter matrix (in between class and within the class scatter matrix)

3. Sort the Eigen Vector by decrease Eigen Value and choose k eigenvector with the largest eigenvalue to from a d x k dimensional matrix w (where every column represent an eigenvector)

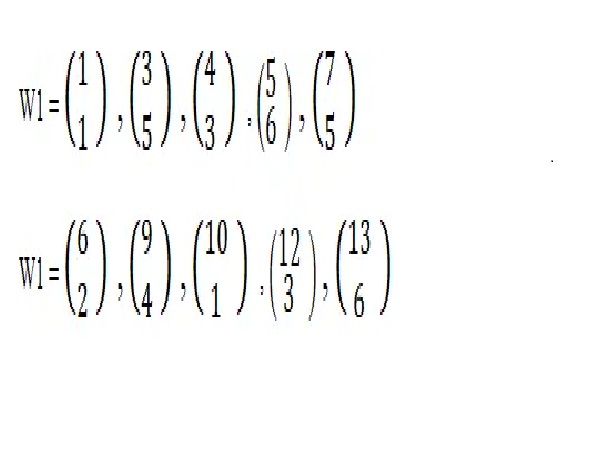
4. Used **d \* k** eigenvector matrix to transform the sample onto the new subspace.

This can be summarized by the matrix multiplication.

Y = X x W (where X is a **n \* d** dimension matrix representing the n samples and **you** are transformed **n \* k** dimensional samples in the new subspace.

**Let’s Dive into Mathematics**

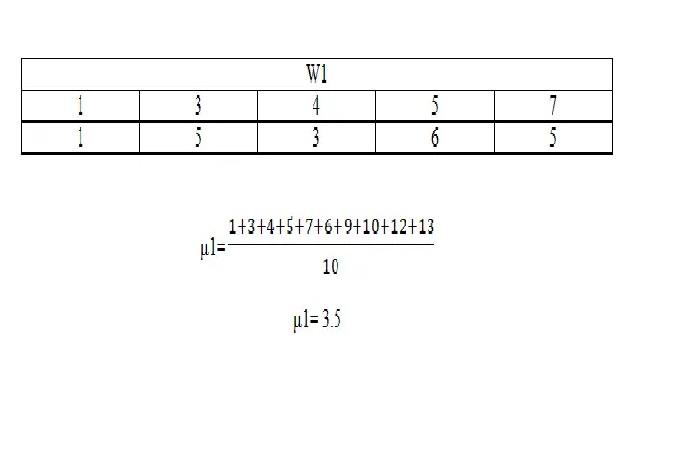
**Dataset:**



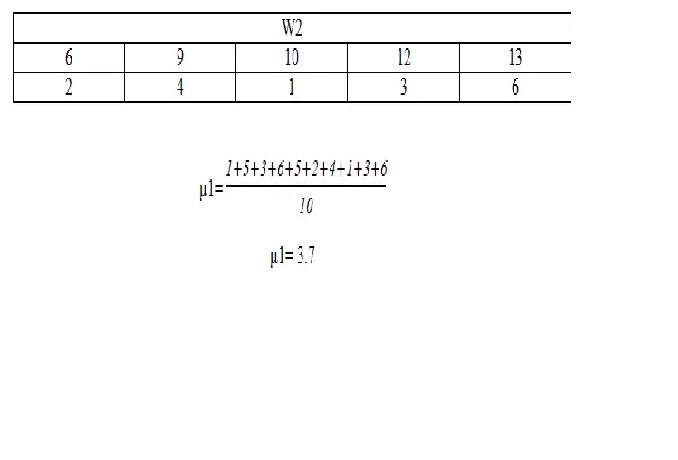
Here W1 and W 2 Two different classes w1 belong to class 1 and W 2 belongs to class 2

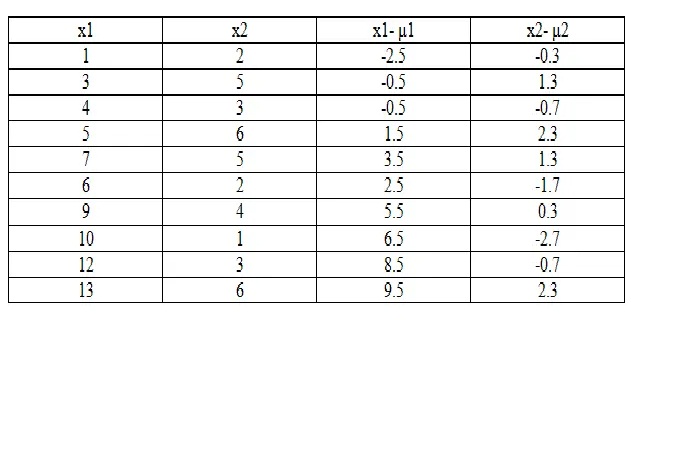
Solution:

**For Class 1**

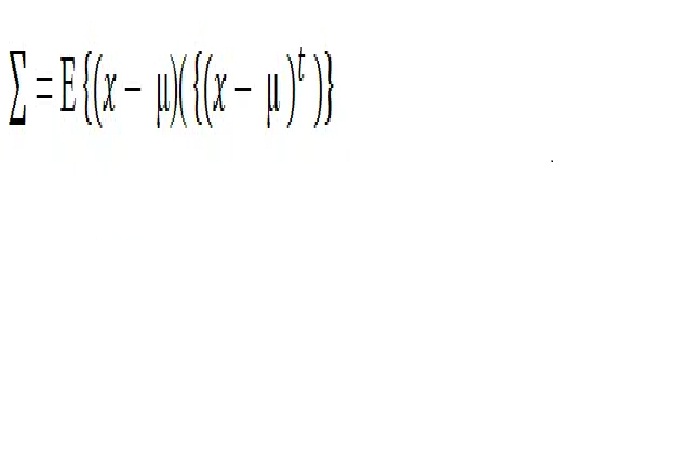


**For Class 2**

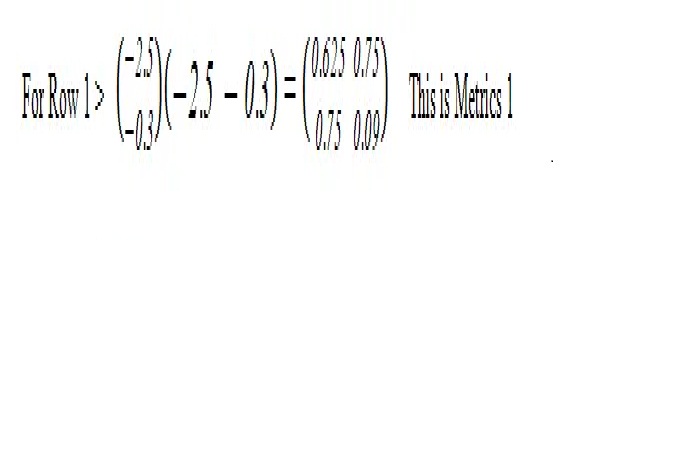




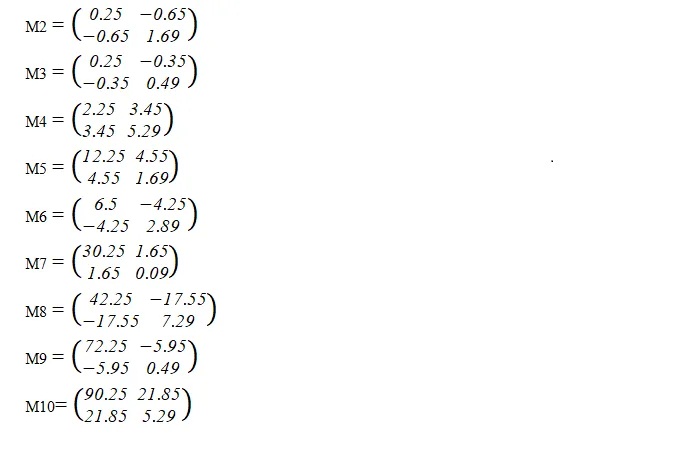
Now



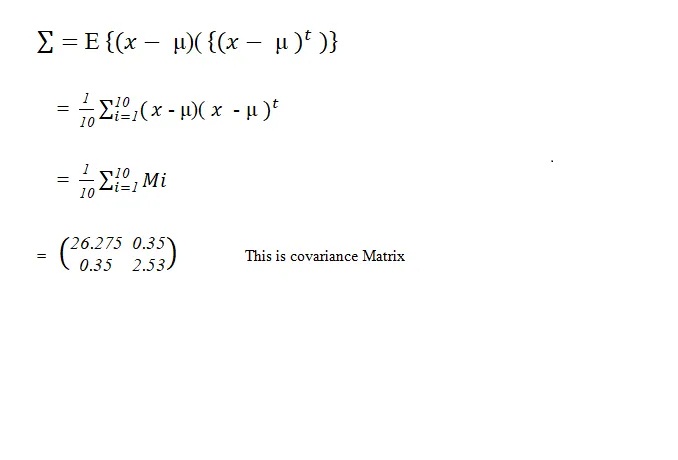
So the equation applies to the above table and we get



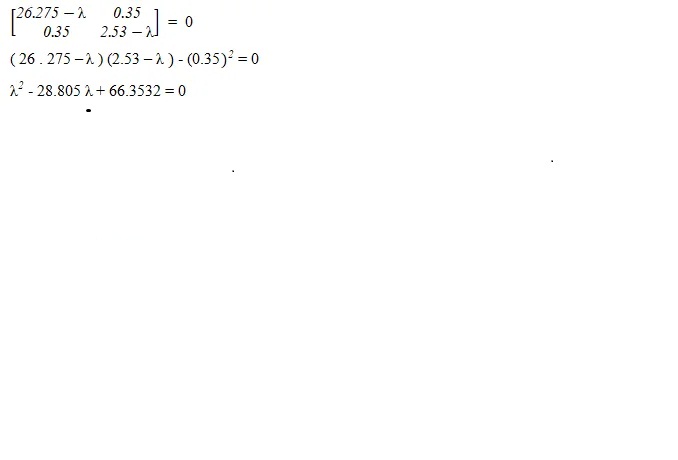
**Similarly, the procedure for others row**



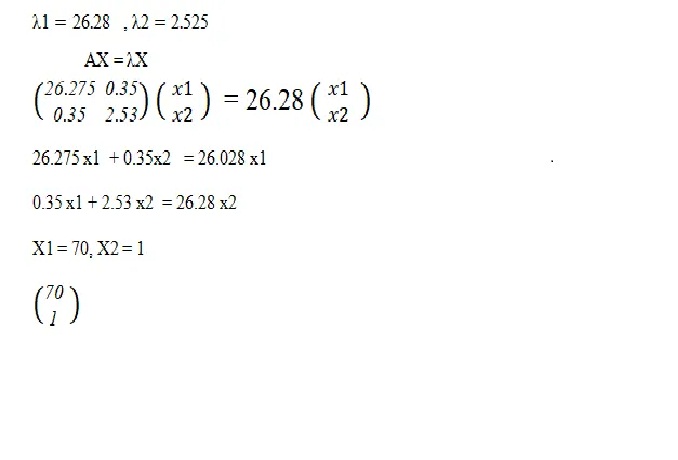
Now

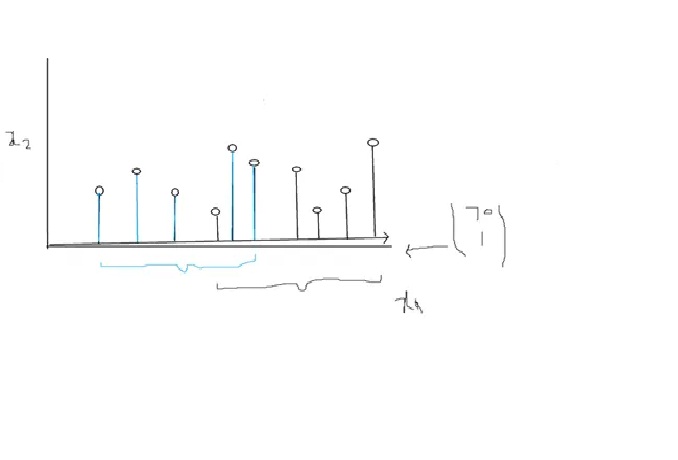


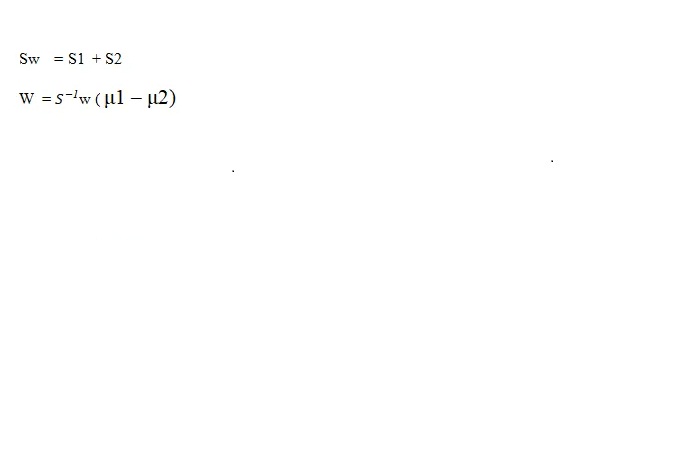
Now find Eigen Value and Eigen Matrix

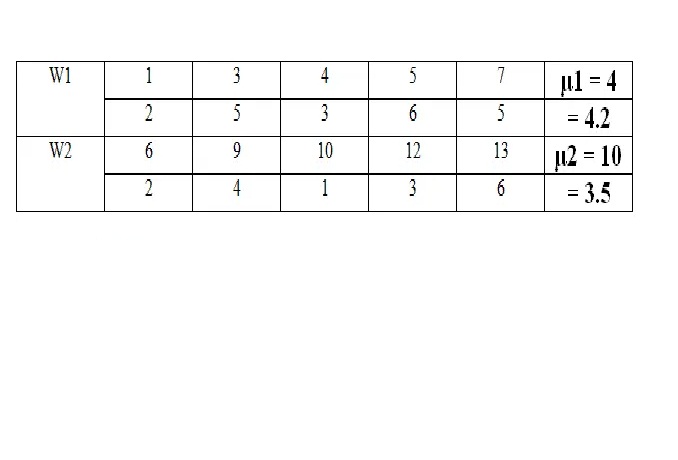


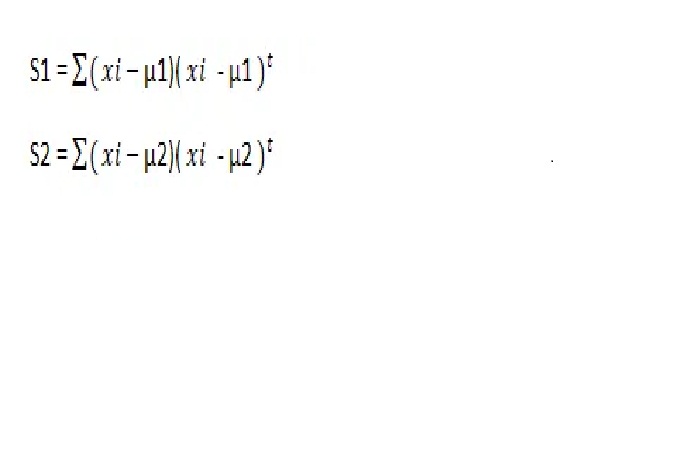
Apply the Quadratic formula and we get the 2 lambda values, these values are eigenvalues.



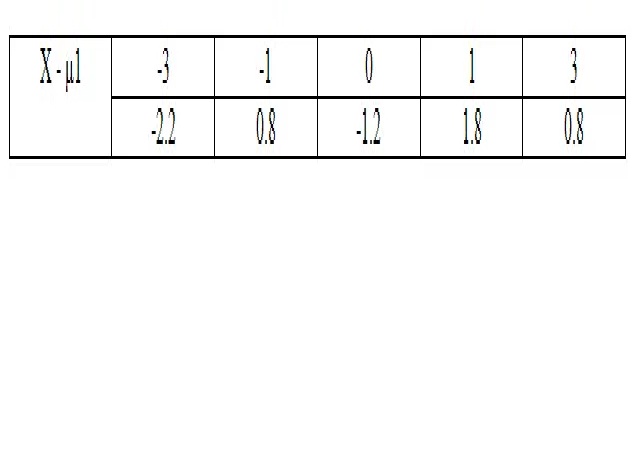




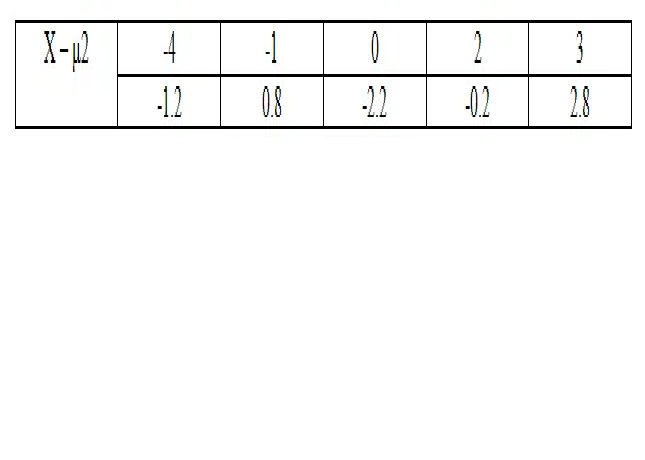


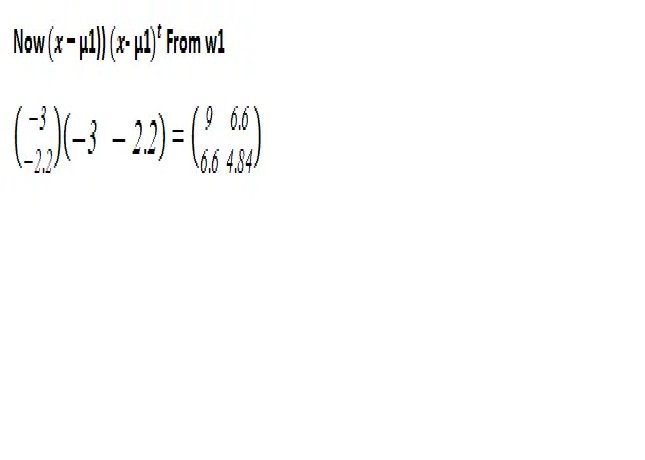


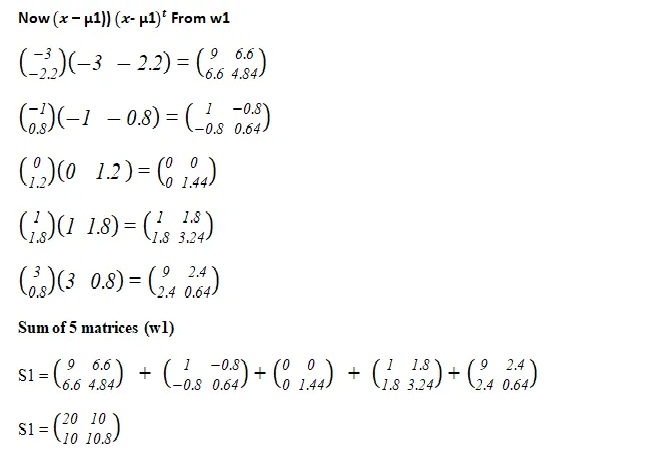
Now From w1

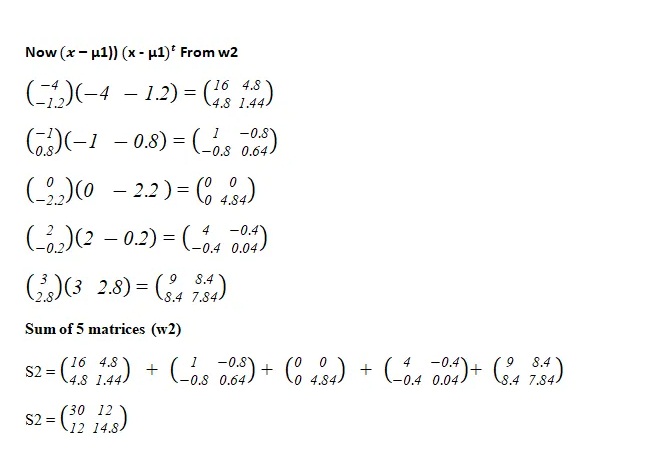


Now From w2

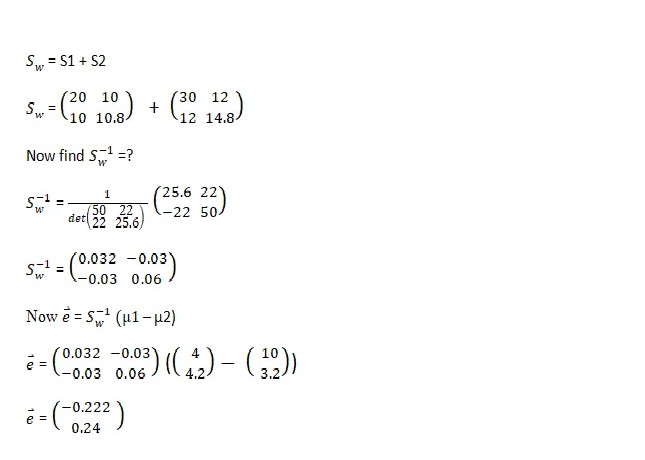


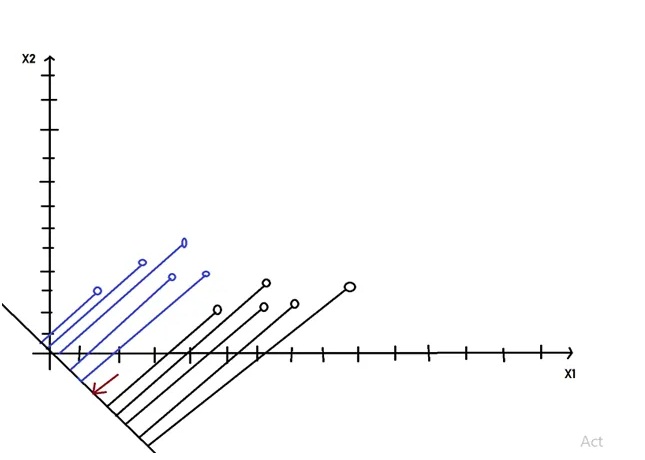






After finding the S1 and S2 we can Sw





In this way, LDA works.

# 3: Practical Implementation of Dimensionality Reduction.

## 3.1: Practical Implementation of PCA

**Dataset Description:**

These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The initial data set had around 30 variables, but for some reason Only have the 13-dimensional version. The attributes are:1) Alcohol 2) Malic acid 3) Ash 4) Alcalinity of ash 5) Magnesium 6) Total phenols 7) Flavanoids 8) Nonflavanoid phenols 9) Proanthocyanins 10)Color intensity 11)Hue 12)OD280/OD315 of diluted wines 13)Proline. All attributes are continuous: No statistics available, but suggest to standardize the variables for certain uses (e.g. For use with classifiers which are NOT scaled invariant) NOTE: 1st attribute is the class identifier (1–3). I use the PCA technique for the Dimensionality Reduction of the wine dataset.

**Part 1: Data Preprocessing:**

**1.1 Import the Libraries**

In this step, we import three Libraries in Data Preprocessing part. A library is a tool that you can use to make a specific job. First of all, we import the **numpy** library used for multidimensional array then import the **pandas** library used to import the dataset and in last we import **matplotlib** library used for plotting the graph.

**1.2 Import the dataset**

In this step, we import the dataset to do that we use the **pandas** library. After import our dataset we define our Predictor and target attribute. we call ‘**X**’ predictor here and target attribute which we call ‘**y**’ here.

**1.3 Split the dataset for test and train**

In this step, we split our dataset into a test set and train set and an 80% dataset split for training and the remaining 20% for tests.

**Feature Scaling**

Feature Scaling is the most important part of data preprocessing. If we see our dataset then some attribute contains information in Numeric value some value very high and some are very low if we see the age and estimated salary. This will cause some issues in our machinery model to solve that problem we set all values on the same scale there are two methods to solve that problem first one is Normalize and Second is Standard Scaler.

Here we use standard Scaler import from Sklearn Library.

**Part 2: Applying the principal component analysis**

In this part, we use PCA for Dimensionality Reduction.

**2.1 Import the Libraries**

In this step, we import a PCA model from Scikit Learn Library.

**2.2 Initialize our model**

In this step, we use the number of components =2 which have high covariance.

**3.3 Fitting the Model**

In this step, we fit the X data into the model.

**3.4 Check the Variance**

In this step we explain our variance (n\_componets = 2)

**Part 3: Applying a model after Dimensionality Reduction Step.**

**3.1 Import the Libraries**

In this step, we are building our model to do this first we import a model from Scikit Learn Library.

**3.2 Initialize our Logistic Regression model**

In this step, we initialize our Logistic Regression model

**3.3 Fitting the Model**

In this step, we fit the training data into our model X\_train, y\_train is our training data.

**Part 4: Making the Prediction and Visualizing the result:**

In this Part, we make a prediction of our test set dataset and visualizing the result using the **matplotlib** library.

**4.1 Predict the test set Result**

In this step, we predict our test set result.

**4.2 Confusion Metric**

In this step we make a confusion metric of our test set result to do that we import confusion matrix from sklearn.metrics then in confusion matrix, we pass two parameters first is y\_test which is the actual test set result and second is y\_pred which predicted result.

**4.3 Accuracy Score**

In this step, we calculate the accuracy score based on the actual test result and predict test results.

**Note**: accuracy is not good, but our mission is to try to understand how our model works in the practical implementation

**4.4 Visualize our Test Set Result**

In this Step, we Visualize our test set result.

If you want dataset and code you also check my **[Github](https://github.com/AmirAli5/Machine-Learning" \t "_blank)** Profile.

## 3.2: Practical Implementation of LDA.

**Dataset Description:**

**Note:** Same Wine dataset which we use in the PCA model using here in the LDA model. Because our accuracy result is not good in PCA and here we obtain much more accuracy to use this model.

These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The initial data set had around 30 variables, but for some reason Only have the 13-dimensional version. The attributes are:1) Alcohol 2) Malic acid 3) Ash 4) Alcalinity of ash 5) Magnesium 6) Total phenols 7) Flavanoids 8) Nonflavanoid phenols 9) Proanthocyanins 10)Color intensity 11)Hue 12)OD280/OD315 of diluted wines 13)Proline. All attributes are continuous: No statistics available, but suggest to standardize the variables for certain uses (e.g. For use with classifiers which are NOT scaled invariant) NOTE: 1st attribute is the class identifier (1–3). I use the PCA technique for the Dimensionality Reduction of the wine dataset.

**Part 1: Data Preprocessing:**

**1.1 Import the Libraries**

In this step, we import three Libraries in Data Preprocessing part. A library is a tool that you can use to make a specific job. First of all, we import the **numpy** library used for multidimensional array then import the **pandas** library used to import the dataset and in last we import **matplotlib** library used for plotting the graph.

**1.2 Import the dataset**

In this step, we import the dataset to do that we use the **pandas** library. After import our dataset we define our Predictor and target attribute. we call ‘**X**’ predictor here and target attribute which we call ‘**y**’ here.

**1.3 Split the dataset for test and train**

In this step, we split our dataset into a test set and train set and an 80% dataset split for training and the remaining 20% for tests.

**Feature Scaling**

Feature Scaling is the most important part of data preprocessing. If we see our dataset then some attribute contains information in Numeric value some value very high and some are very low if we see the age and estimated salary. This will cause some issues in our machinery model to solve that problem we set all values on the same scale there are two methods to solve that problem first one is Normalize and Second is Standard Scaler.

Here we use standard Scaler import from Sklearn Library.

**Part 2: Applying the Linear Discriminant analysis**

In this part, we use LDAA for Dimensionality Reduction.

**2.1 Import the Libraries**

In this step, we import an LDA model from Scikit Learn Library.

**2.2 Initialize our model**

In this step, we use the number of components =2 which have high covariance

**2.3 Fitting the Model**

In this step, we fit the X data into the model.

**Part 3: Applying a model after Dimensionality Reduction**

**3.1 Import the Libraries**

In this step, we are building our model to do this first we import a model from Scikit Learn Library.

**3.2 Initialize our Logistic Regression model**

In this step, we initialize our Logistic Regression model

**3.3 Fitting the Model**

In this step, we fit the training data into our model X\_train, y\_train is our training data.

**Part 4: Making a Prediction and Visualize the result**

In this Part, we make a prediction of our test set dataset and visualizing the result using the **matplotlib** library.

**4.1 Predict the test set Result**

In this step, we predict our test set result.

**4.2 Confusion Metric**

In this step we make a confusion metric of our test set result to do that we import confusion matrix from sklearn.metrics then in confusion matrix, we pass two parameters first is y\_test which is the actual test set result and second is y\_pred which predicted result.

**4.3 Visualize our Test Set Result**

In this Step, we Visualize our test set result.

I