# **Understanding Clustering**

You will start by learning what “clustering” is. It is an unsupervised learning technique, where you try to find patterns based on similarities in the data. Then, you will be introduced to a case study that shows the applicability of clustering in the industry.

In the previous modules, you saw various supervised machine learning algorithms. Supervised machine learning algorithms make use of labelled data to make predictions.

For example, an email will be classified as spam or ham, or a bank’s customer will be predicted as ‘good’ or ‘bad’. You have a target variable Y which needs to be predicted.

On the other hand, in unsupervised learning, you are not interested in prediction because you do not have a target or outcome variable. The objective is to discover interesting patterns in the data, e.g. are there any subgroups or ‘clusters’ among the bank’s customers?

**PRACTICAL APPLICATIONS OF CLUSTERING**

1. **Customer Insight:** Say, a retail chain with so many stores across locations wants to manage stores at best and increase the sales and performance. Cluster analysis can help the retail chain to get desired insights on customer demographics, purchase behaviour and demand patterns across locations. This will help the retail chain for assortment planning, planning promotional activities and store benchmarking for better performance and higher returns.
2. **Marketing:** Cluster Analysis can help with In the field of marketing, Cluster Analysis can help in market segmentation and positioning, and to identify test markets for new product development.
3. **Social Media:** In the areas of social networking and social media, Cluster Analysis is used to identify similar communities within larger groups.
4. **Medical**: Cluster Analysis has also been widely used in the field of biology and medical science like human genetic clustering, sequencing into gene families, building groups of genes, and clustering of organisms at species.

**In Clustering**, we group the data points into different categories based on the given set of attributes. There are no dependent and independent variables.

Customer segmentation for targeted marketing is one of the most vital applications of the clustering algorithm. Here, as a manager of the online store, you would want to group the customers into different clusters, so that you can make a customised marketing campaign for each of the group. You do not have any label in mind, such as good customer or bad customer. You want to just look at patterns in customer data and then try and find segments. This is where clustering techniques can help you with segmenting the customers. Clustering techniques use the raw data to form clusters based on common factors among various data points. This is exactly what will also be done in segmentation, where various people or products will be grouped together on the basis of similarities and differences between them.

As a manager, you would have to decide what the important business criteria are on which you would want to segregate the customers. So, you would need a method or an algorithm that itself decides which customers to group together based on this criteria.

Sounds interesting? Well, that is the beauty of unsupervised learning, especially clustering.

**Clustering v/s Segmentation:**

1. Clustering - Analytics Technique
2. Segmentation – Business Problem

To be able to do segmentation we use clustering techniques.

for successful segmentation, the segments formed must be stable. This means that the same person should not fall under different segments upon segmenting the data on the same criteria. You also saw that segments should have **intra-segment homogeneity** and **inter-segment heterogeneity**. You will see in later sessions how this can be defined mathematically.

3 types of segmentation are used for customer segmentation:

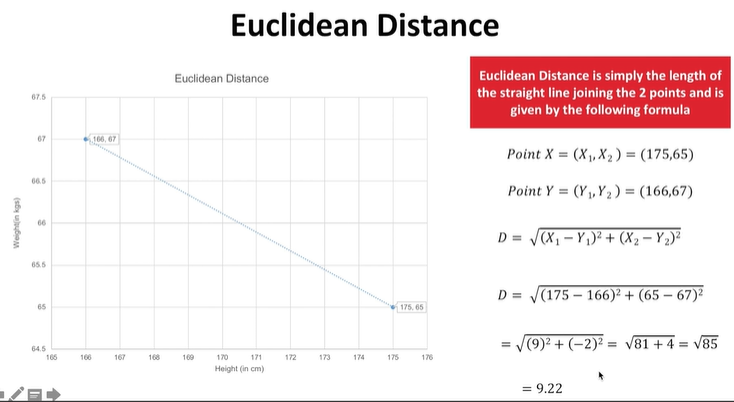
* **Behavioural segmentation**: Segmentation is based on the actual patterns displayed by the consumer
* **Attitudinal segmentation**: Segmentation is based on the beliefs or the intents of people, which may not translate into similar action
* **Demographic segmentation**: Segmentation is based on the person’s profile and uses information such as age, gender, residence locality, income, etc.
* **Clustering**
* Which of the following are applications of clustering?
* Top of Form
* 
* Looking at social media behaviour to find out what types of online communities are there
* 
* Identify consumer segments and their properties to position products appropriately
* 
* Identifying patterns of crime in different regions of a city and managing police enforcement based on frequency and type of crime
* 
* All of the above
* **Feedback :***All the above are activities that involve looking at the data and identifying hidden patterns and properties and seeing if some data is different from other and in what aspects is it different*
* **Correct**
* Bottom of Form

# **K-Means Clustering**

Clustering works on the basis of grouping the observations which are the most similar to each other. What does this exactly mean?

In simple terms, the algorithm needs to find data points whose values are **similar** to each other and therefore these points would then belong to the same cluster. The method in which any clustering algorithm goes about doing that is through the method of finding something called a “**distance measure**”. The distance measure that is used in K-means clustering is called the **Euclidean Distance** measure.

## **Euclidean Distance**



Euclidean Distance between the 2 points is measured as follows: If there are 2 points X and Y having n dimensions

X=(X1,X2,X3,...Xn)

Y=(Y1,Y2,Y3,....Yn)

Then the **Euclidean Distance D**is given as

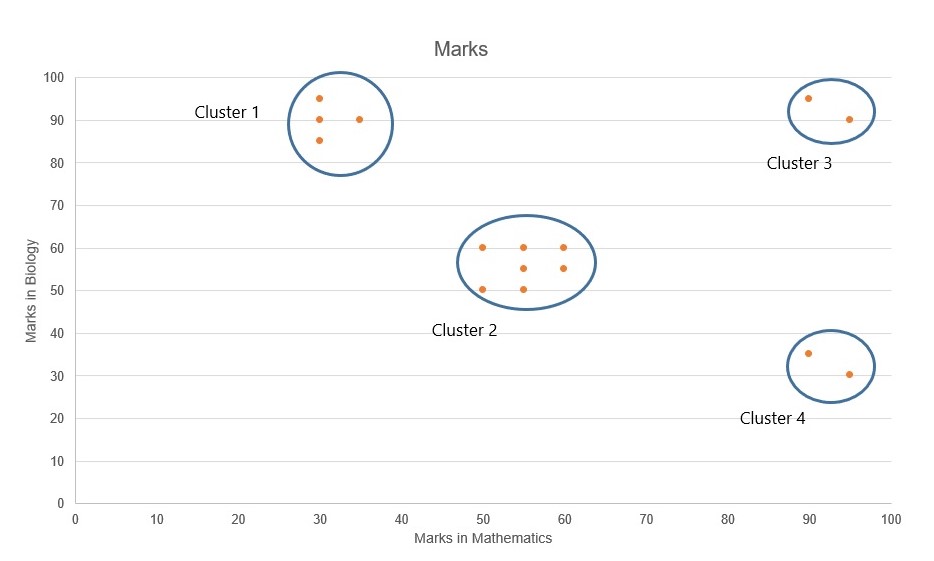
D=√(X1−Y1)^2+(X2−Y2)^2+...(Xn−Yn)^2

The idea of distance measure is quite intuitive. Essentially, the observations which are closer or more similar to each other would have a low Euclidean distance and the observations which are farther or less similar to each other would have a higher Euclidean distance.

# **Centroid**

The next concept that is crucial for understanding how clustering generally works is the idea of centroids. If you remember your high school geometry, centroids are essentially the centre points of triangles. Similarly, in the case of clustering, centroids are the**centre points of the clusters**that are being formed.

Now before going to the formula part, here is an intuition for the need of a centroid. Imagine you have the following clusters of the marks of a group of students in Mathematics and Biology and someone asks you to explain them. From a glance, you can easily interpret the 4 clusters that are being formed

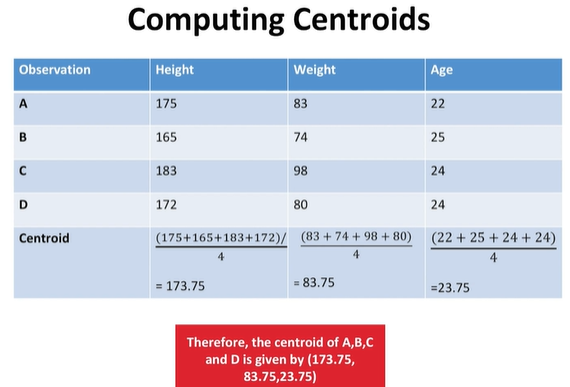


 So the four clusters that are being formed are as follows:

Cluster 1: Students who have scored high marks in Bio, but poor marks in Maths  
Cluster 2: Students who have scored average  marks in Bio and  Maths  
Cluster 3: Students who have scored high marks in both Bio and Maths.  
Cluster 4: Students who have scored high marks in Maths, but poor marks in Bio

 Now the above representation is fine and correct, but it is missing one crucial information - **the numerical order**. For example, when you want to compare two clusters say Cluster 1 and Cluster 2 can you say by how much marks on average do the students from Cluster 1 outperform or underperform the Cluster 2 students in a particular subject just by taking a look at the above visualisation alone? Is it by 10 marks? Or 15?

This is where the concept of **Centroids** come in handy



The centroid is calculated by computing the mean of each and every column/dimension that you have and then ordering them in the same way as above

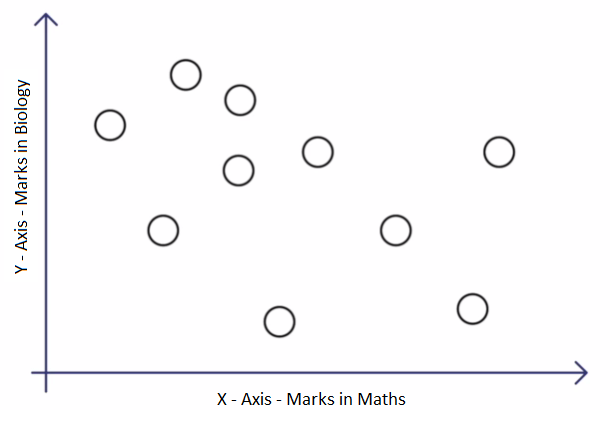
Therefore, Height-mean = ((175+165+183+172)/)/4  = 173.75  
                    Weight-mean = ((83+74+98+80))/4 = 83.75  
                    Age - mean = ((22+25+24+24))/4 =23.75

Thus the centroid of the above group of observations is (173.75, 83.75 and 23.75)

Let’s say you have the data of 10 students and their marks in Biology and Math (as shown in the plot below). You want to divide them into two clusters so that you can see what kind of students are there in the class.

The y-axis shows the marks in Biology, and the x-axis shows the marks in Math.

Imagine two clusters dividing this data — one red and the other yellow. How many points would each cluster have?



**Fig 1: Random points to be divided into 2 clusters**

Each time the clusters are made, the centroid is updated. The updated centroid is the centre of all the points which fall in the cluster associated with the centroid. This process continues till the centroid no longer changes, i.e. the solution converges.

Thus, you can see that the K-means algorithm is a clustering algorithm that takes N data points and groups them into K clusters. In this example, we had N =10 points and we used the K-means algorithm to group these 10 points into K = 2 clusters.



**Fig 2: Final cluster**

**Practice Question**: Repeating the previous two steps again, you get the new cluster centaers (H38:I39).Continue this process until the algorithm converges (Two consecutive iterations have the same center). What are the x and y coordinates of the center of cluster 1 finally?

The centers of the clusters are calculated in the following way:

1. The centers at first are randomly selected, and the euclidean distance between the points and the centers are calculated.

2. Based on the distance with the centers, the points are grouped into different clusters(Here: Cluster 1 and Cluster 2).

3. Once the grouping is done, the centroid value of all the points belonging to cluster 1 is calculated. This means we have to find the average value of x and y coordinates of all points belonging to cluster 1. In the Excel sheet, we use the averageif method, to calculate the average of x and y values if they belong to any particular cluster. This gives the center of the cluster 1 and it is repeated to get the center of cluster 2 as well.

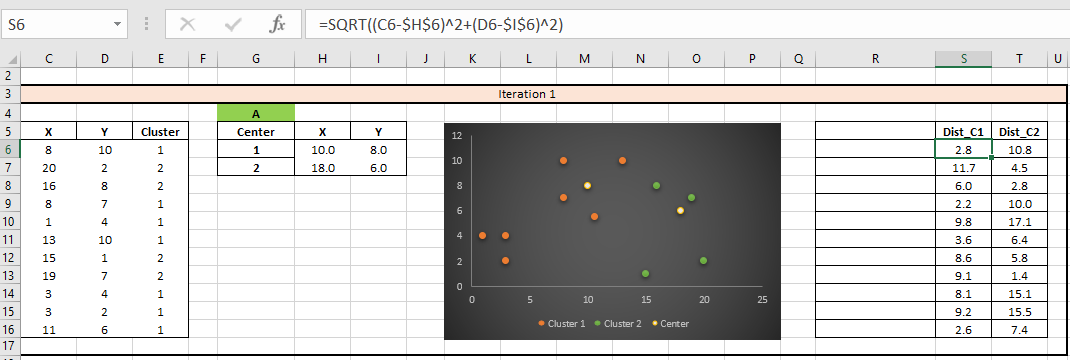
AVERAGEIF(range, criteria, average\_range)

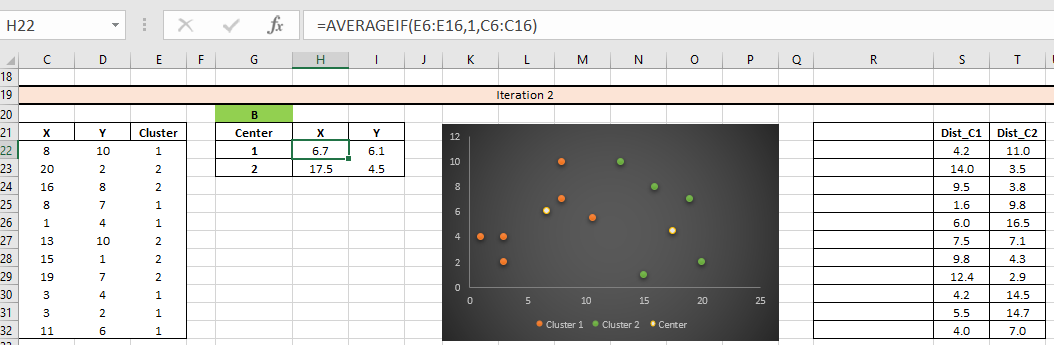
range - The column containing the cluster values 1/2. (Column E)

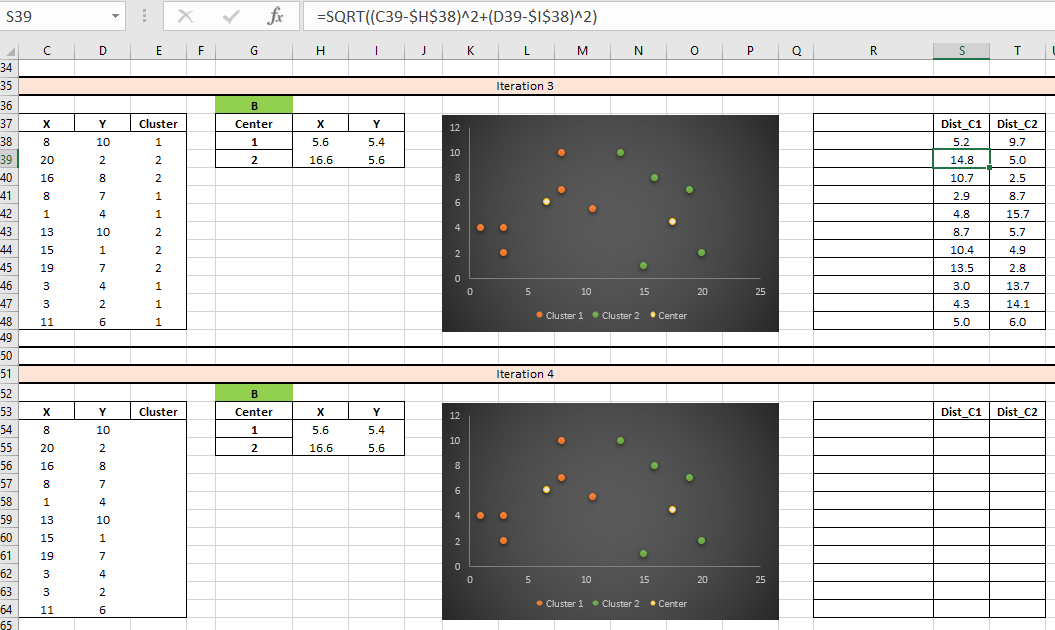
Criteria - 1 for cluster 1, 2 for cluster 2

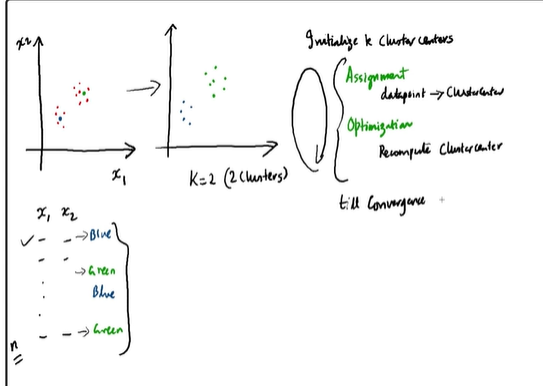
average\_range - X values (Column C) for X value of centroid, Y values (Column D) for Y value of centroid

4. Once the centers of both clusters are calculated, the euclidean distance is calculated again for all points and steps are repeated again till we have the same values for consecutive iterations.





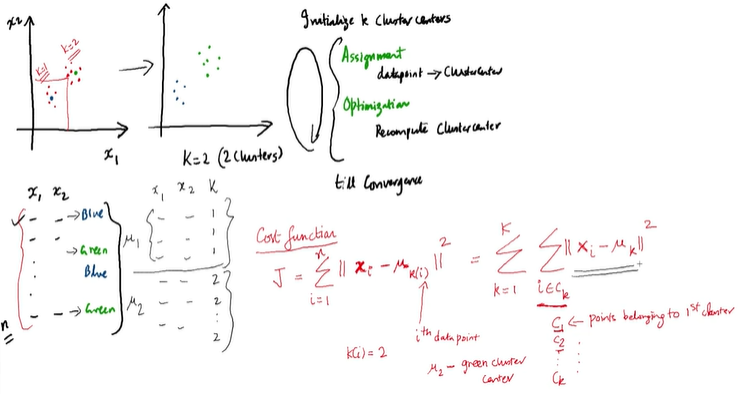




we understood that the algorithm’s inner-loop iterates over two steps:

1. Assign each observation Xi to the closest cluster centroid μk
2. Update each centroid to the mean of the points assigned to it.

In the next lecture, we will learn about the Kmeans cost function and will also see how to compute the cost function for each iteration in the K-means algorithm.

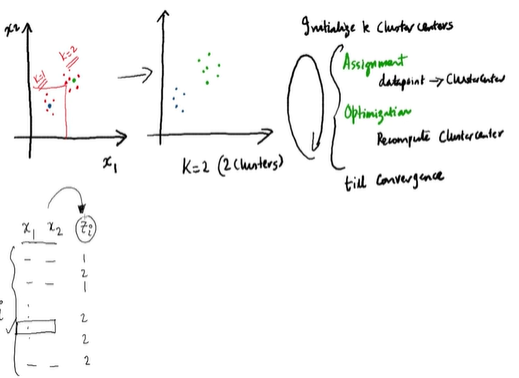


So the cost function for the K-Means algorithm is given as:

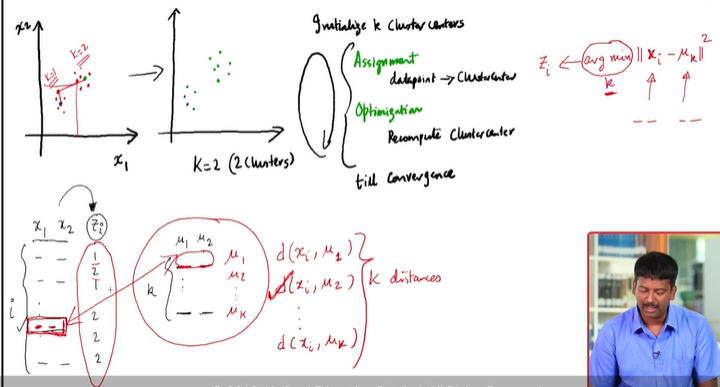
J=∑ni=1||Xi−μk(i)||^2=∑KK=1∑iϵck||Xi−μk||^2

Now in the next video, we will learn what exactly happens in the assignment step? and we will also look at how to assign each data point to a cluster using the K-Means algorithm assignment step

In **assignment step (1st step)**, for every data point of our features x1, x2, for ith data point we will be assigning a Zi whose value would be either 1 or 2. Because we have two cluster K=2.



For **optimizing step (2nd stpe)**, we need to calculate cluster centers or centroids which are μ1 and μ2. Since we have k=2, we calculate two centroids.



**K-Means**

What is the significance of "argmin" in the assignment step equation?

Once we calculate the euclidean distance between each data points with mean (mu), need to pick up the min of these calculated distances. Idea is to find the min distance of the sample data point and assign that into a particular group .

**Suggested Answer**

For a ith data point which is a 2d object and μ which is again a 2d object, we compute the distance between these two, this is given by d(xi,μk) where k is the number of clusters and then from these k different results we will choose the minimum of all.

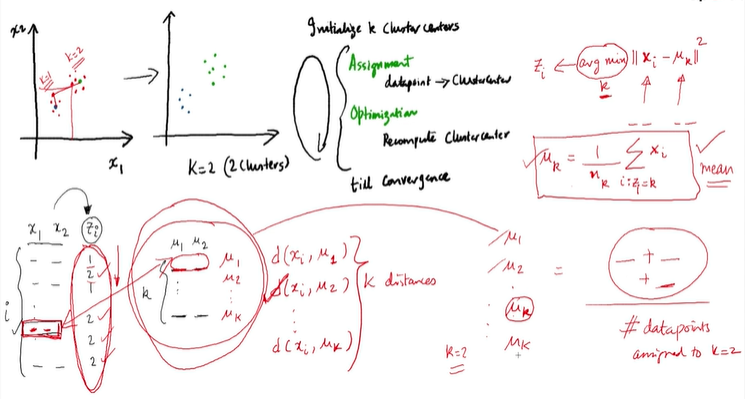
In the assignment step, we assign every data point to K clusters. The algorithm goes through each of the data points and depending on which cluster is closer, in our case, whether the green cluster centroid or the blue cluster centroid; It assigns the data points to one of the 2 cluster centroids.

The equation for the assignment step is as follows:

Zi=argmin||Xi−μk||2

Now having assigned each data point to a cluster, now we need to recompute the cluster centroids. In the next lecture, Prof.Dinesh will explain how to recompute the cluster centroids or the mean of each cluster.

Before the assignment step we went over all Zi’s and assign 1 or 2. Now again we recompute our μ1, μ2 … upto μk.



In the optimisation step, the algorithm calculates the average of all the points in a cluster and moves the centroid to that average location.

The equation for optimisation is as follows:

μk=1/nk∑i:zi=kXi

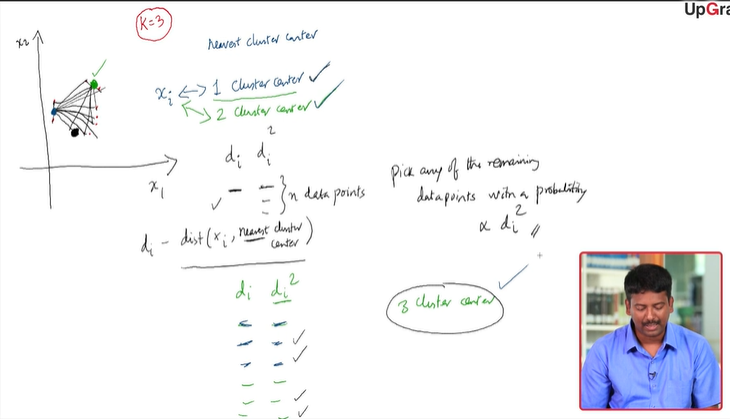
The process of assignment and optimisation is repeated until there is no change in the clusters or possibly until the algorithm converges.

# **K Means++ Algorithm**

We looked in the previous segment that for K-Means optimisation problem, the algorithm it iterate between two steps and tries to minimise the objective function given as,

Zi=argmin||Xi−μk||2

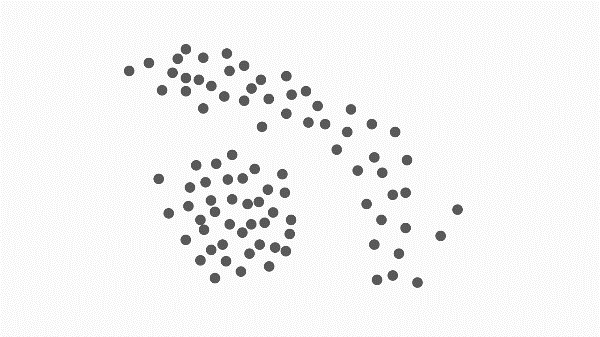
To choose the cluster centers smartly, we will learn about K-Mean++ algorithm. K-means++ is just an initialisation procedure for K-means. In K-means++ you pick the initial centroids using an algorithm that tries to initialise centroids that are far apart from each other.

****

To summarise, In K-Means++ algorithm,

1. We choose one center as one of the data points at random.
2. For each data point Xi, We compute the distance between Xi and the nearest center that had already been chosen.
3. Now, we choose the next cluster center using the weighted probability distribution where a point X is chosen with probability proportional to d(X)2 .
4. Repeat Steps 2 and 3 until K centers have been chosen.

Let’s see the K-Means algorithm in action using a visualisation tool. This tool can be found on [naftaliharris.com](http://www.naftaliharris.com/blog/visualizing-k-means-clustering/).



**K-Means algorithm**

Consider the above arrangement of points. How many clusters do you intuitively feel are present. What will happen if you use K-Means clustering here? How do you think this problem can be solved?

**Suggested Answer**

*Intuitively, it looks like 2 clusters are present. If we use K means, then we will get wrong clusters since the points in the outer ring like structure will not be segmented accurately. A reason for that is K-Means looks for how close the points are to a centroid and this distance or measure of closeness is the “linear distance”. One way to correct this can be to see the distance between all the points and then cluster the closest points.*

**K-Means algorithm**

In this exercise, you will perform k-means clustering manually on a small dataset. Consider the following dataset having 2 features and 6 observations.

|  |  |  |
| --- | --- | --- |
| **Observation number** | **X1** | **X2** |
| **1** | **1** | **4** |
| 2 | 1 | 3 |
| 3 | 0 | 4 |
| 4 | 5 | 1 |
| 5 | 6 | 2 |
| 6 | 4 | 0 |

Use k = 2 for the entire exercise.

Assign clusters to each observation such that the odd numbered observations get assigned cluster = 1, i.e. points 1, 3 and 5 get assigned cluster = 1 and the even ones get assigned cluster = 2.

Compute the centroid of the two clusters

Assign each observation to the centroid to which it is closest (using euclidean distance). Report the new cluster labels for each observation.

Repeat the above steps until the clusters stop changing

(1, 2, 3) and (4, 5, 6)

**Feedback :***You will see that the solution converges really fast. After just the first iteration, the solution converges. After the first iteration, observation 1,2,3 get assigned to cluster 1 and observation 4,5,6 get assigned to cluster 2. The initial cluster centroids were (2.3,3.3) and (3.3,1.3) and the distances from these is calculated to assign the clusters.*

**Correct**



# **Practical Consideration in K Means Algorithm**

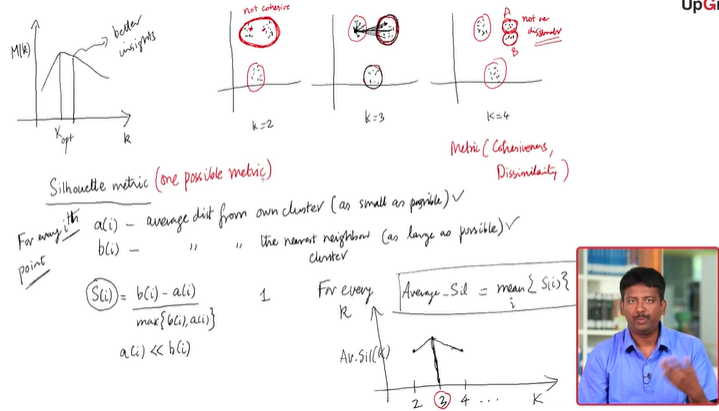
The major practical considerations involved in K-Means clustering are:

* The number of clusters that you want to divide your data points into, i.e. the value of K has to be pre-determined.
* The choice of the initial cluster centres can have an impact on the final cluster formation.
* The clustering process is very sensitive to the presence of outliers in the data.
* Since the distance metric used in the clustering process is the Euclidean distance, you need to bring all your attributes on the same scale. This can be achieved through standardisation.
* The K-Means algorithm does not work with categorical data.
* The process may not converge in the given number of iterations. You should always check for convergence.

**How to choose K for K-Means algorithm – Silhouette Analysis**

To compute silhouette metric, we need to compute two measures i.e. a(i) and b(i) where,

* a(i) is the average distance from own cluster(Cohesion).
* b(i) is the average distance from the nearest neighbour cluster(Separation).



**Silhouette score** is a way to measure how close each point in a cluster is to the points in its neighboring clusters. It is used to determine the optimum value of k i.e the number of clusters to be created in K means algorithm. Silhouette values lies in the range of [-1, 1].

* A value of +1 indicates that the sample is far away from its neighboring cluster and very close to the cluster its assigned.
* And, a value of 0 means its at the boundary of the distance between the two cluster.
* Similarly, value of -1 indicates that the point is close to its neighboring cluster than to the cluster its assigned.

The goal is to find group(s) of clusters. Clusters are visually identified by the proximity of the points around a centre point.

Smilarly, the idea to minimise the distance between the data points to the centroid is to ensure that it is not loosely bound (or) to ensure that the data points are closely bound.

Likewise, in order to ensure that data point of cluster (say X) does not belong to another cluster (say Y), the distance from the data point to the other cluster is maximized. A higher value indicates that it is away from the cluster whereas a lower value indicates it is closer to the cluster (Y). This can result in moving the centroids.

**K-Means algorithm**

If we are worried about K-means getting stuck in bad local optima, one way to solve this problem is if we try using multiple random initializations. Is this true or false? You can read about local optimum and global optimum [here](https://en.wikipedia.org/wiki/Local_optimum)

Top of Form



True

**Feedback :**Since each run of K-means is independent, multiple runs can find different local optima, and this can help in choosing the global optimum value.

Bottom of Form

**Cluster Tendency**

The process to evaluate the data to check if the data is feasible for clustering or not is know as the clustering tendency.

To check cluster tendency, we use Hopkins test. Hopkins test examines whether data points differ significantly from uniformly distributed data in the multidimensional space.

**K-Means algorithm**

Arrange the steps of k-means algorithm in the order in which they occur:

1. Randomly selecting the cluster centroids
2. Updating the cluster centroids iteratively
3. Assigning the cluster points to their nearest center
4. 1-3-2
5. **Feedback :***First the cluster centers are pre-decided. Then all the points are assigned to their nearest cluster center and then the center is recalculated as the mean of all the points which fall in that cluster. Then the clustering is repeated with the new centers and the centers are updated according to the new cluster points.*

**K-Means algorithm**

Consider three cluster centres A(2,3), B(4,5) and C(6,2). A point (1,2) is to be assigned to one of these clusters. According to k-means clustering concepts and using euclidean distance as the measure of closeness, which cluster should it be assigned to?

Top of Form



A

**Feedback :***According to k-means algorithm, the point should be assigned to the centre with the minimum distance from the point. The distances for A, B, C are sqrt(2), sqrt(18) and sqrt(25)*

**Correct**



Bottom of Form

# **Behavioral Segmentation**

RFM (Recency-Frequency-Monitory) :

* Recency: How recently the customer made the last purchase or last visited the platform
* Frequency: How frequently does a customer visits or purchases from the platform
* Monetary: What revenue does the customer generate for the platform

RPI (Relation-Persona-and-Intent):

* Relationship: Past interaction with company
* Persona: Type of person
* Intent: Intention at the time of purchase based on browsing at that particular time

CDJ (Consumer Decision Journey):

* Based on customer’s life journey with brand or product. What segmentation of people are churning.

**Summary**

You learnt how to create clusters using the K-means algorithm in Python with the analysis of the Online Store data set. We wanted to group the customers of the store into different clusters based on their purchasing habits. The different steps involved were:

* Missing values treatment
* Data transformation
* Outlier treatment
* Data standardisation
* Finding the optimal value of K
* Implementing K Means algorithm
* Analysing the clusters of customers to obtain business insights

Once we are through with the data preparation, the K-means algorithm is quite easy to implement. All it takes is running the KMeans() function. The only ambiguous point you may notice here is that you need to decide the number of required clusters beforehand and in fact run the algorithm multiple times with a different number K before you can figure out the most optimal number of clusters.

This is also what happens in the industry practices that we run the algorithm multiple times with different values of K and then pick the clusters which make the most business sense. In fact, the k-means algorithm finds large application in the industry. For example, it can be used to find out the most optimal centre to install the mobile towers by clustering the customers geographically. Similarly, it has wide application in medical science, where say the patients can be clustered together on the basis of their symptoms, and then analysed to figure out the cause of their illness.

However, K means was just one of the clustering algorithm. In the next session, we will learn about another clustering algorithm called hierarchical clustering, which does not require you to decide the number of clusters beforehand.

Questions:1/2

**K-Means algorithm**

Select the problem sets, where k-means clustering can be applied.

Top of Form



Weather forecast for next week, given the dataset of weather information for last five years



Given an ecom company’s customer details - the products they purchased and the amount spent. The company wants to group it’s customers based on their buying behaviour.



Predict whether a new customer would respond to a bank’s new product basis his historical information



All of the above

Bottom of Form

Given an ecom company’s customer details - the products they purchased and the amount spent. The company wants to group it’s customers based on their buying behaviour.

**Feedback :***The given option is correct. The other options are predicting the response based on independent variables.*

**K-Means algorithm**

Select the correct statement among the following:

The results of k-means algorithm get impacted by outliers and range of the attributes.

**Feedback :***Depending on the initial selection of centers, the formed clusters might be different. The value of k has to be decided by the user*

# **Hierarchical Clustering**

You will learn about another algorithm to achieve unsupervised clustering. This is called **Hierarchical Clustering**. Here, instead of pre-defining the number of clusters, you first have to visually describe the similarity or dissimilarity between the different data points and then decide the appropriate number of clusters on the basis of these similarities or dissimilarities.

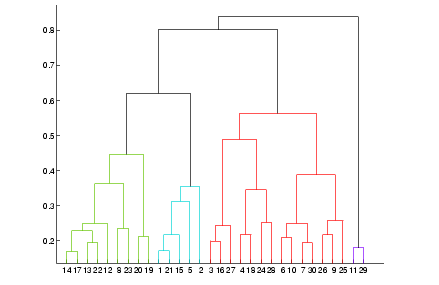
You will learn about:

* Hierarchical clustering algorithm
* Interpreting the dendrogram
* Cutting the dendrogram
* Types of linkages

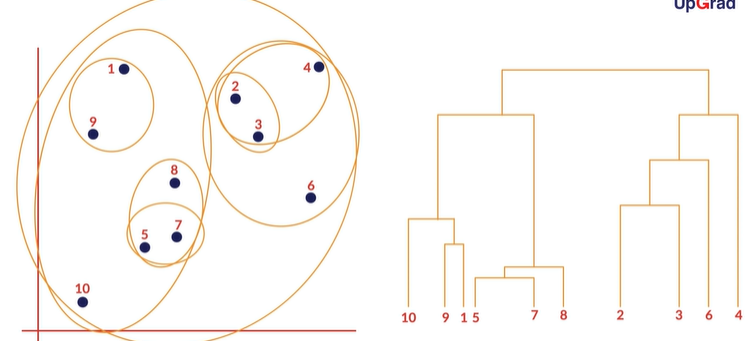
**Hierarchical Clustering Algorithm**

One of the major considerations in using the K-means algorithm is deciding the value of K beforehand. The hierarchical clustering algorithm does not have this restriction.

 The output of the hierarchical clustering algorithm is quite different from the K-mean algorithm as well. It results in an inverted tree-shaped structure, called the dendrogram. An example of a dendrogram is shown below.



**Dendrogram**



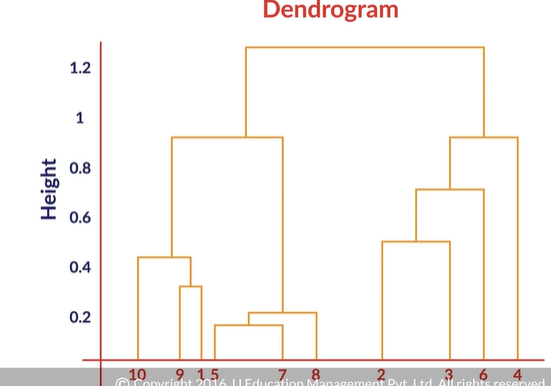
In the K-Means algorithm, you divided the data in the first step itself. In the subsequent steps, you refined our clusters to get the most optimal grouping. In hierarchical clustering, the data is not partitioned into a particular cluster in a single step. Instead, a series of partitions/merges take place, which may run from a single cluster containing all objects to n clusters that each contain a single object or vice-versa.

This is very helpful since you don’t have to specify the number of clusters beforehand.

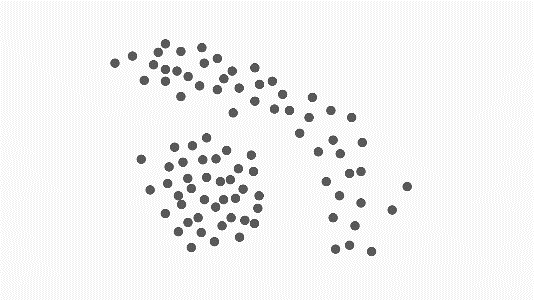
Given a set of N items to be clustered, the steps in hierarchical clustering are:

1. Calculate the NxN distance (similarity) matrix, which calculates the distance of each data point from the other
2. Each item is first assigned to its own cluster, i.e. N clusters are formed
3. The clusters which are closest to each other are merged to form a single cluster
4. The same step of computing the distance and merging the closest clusters is repeated till all the points become part of a single cluster

Thus, what you have at the end is the dendrogram, which shows you which data points group together in which cluster at what distance.



The Height of dendogram at which the different clusters are fused together represents the dissimilarity measure. Which is this case as Euclidean distance between the clusters. Thus the clusters are fused together at top of the tree are more dissimilar to each other than the ones are fused towards to the bottom.



**Hierarchical Clustering**

You had made clusters for it using the K-Means algorithm. How do you think clusters will be made using hierarchical algorithm on this data?

**Suggested Answer**

*Since now you are looking at the closest distance between clusters, you will get two clusters - one at the center and the one which contains the points at the edges.*

**Hierarchical Clustering**

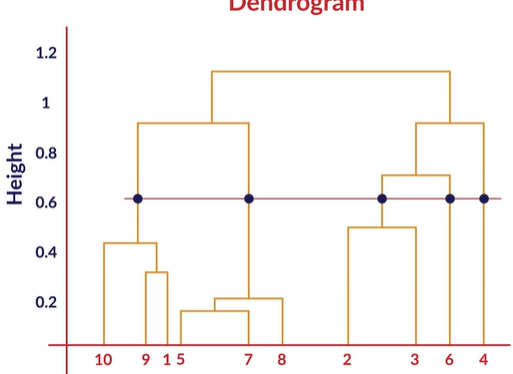
Look at the following matrix. This is the distance matrix between 4 points - A, B,C, D. Find out which 2 clusters will merge first.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | B | C | D |
| A |  |  |  |  |
| B | 2.24 |  |  |  |
| C | 8.06 | 10.00 |  |  |
| D | 5.83 | 8.06 | 5.00 |  |

A-B

**Feedback :***Look at the data and see that the minimum distance is between A and B*

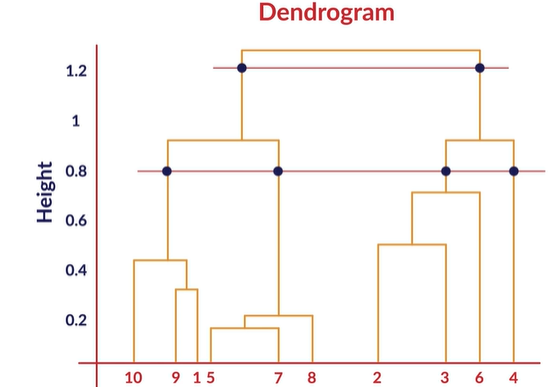
To figure out the no of cluster we choose for data points we cut the dendogram at certain point. We can think of number of verticle lines intersect a horizontal line. The total count of the intersect points are the total cluster or K.



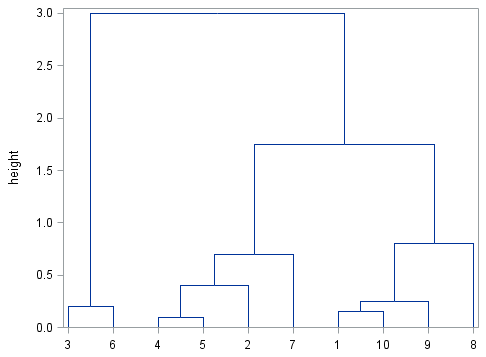
In this diagram we have 5 (four) clusters.

So as we get the dendogram we need to decide on a threshold value on which we will cut the dendogram. Number of clusters would depend on this threashold value.

In the below graphic if we decide the threshold at 0.8 then we get 4 clusters where as for threshold 1.2 we get 2 clusters.



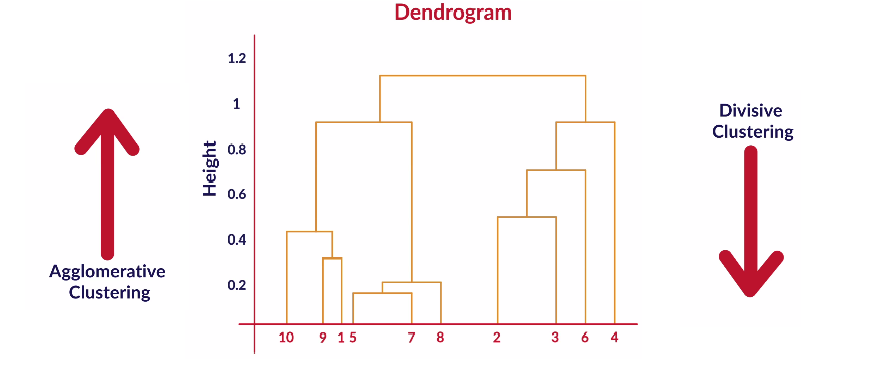
As you saw, the y-axis of the dendrogram is some measure of the dissimilarity or distance at which clusters join.



In the dendrogram shown above, samples 4 and 5 are the most similar and join to form the first cluster, followed by samples 1 and 10. The last two clusters to fuse together to form the final single cluster are 3-6 and 4-5-2-7-1-10-9-8.

Determining the number of groups in a cluster analysis is often the primary goal. Typically, one looks for natural groupings defined by long stems. Here, by observation, you can identify that there are 3 major groupings: 3-6, 4-5-2-7 and 1-10-9-8.

You also saw that hierarchical clustering can proceed in 2 ways — agglomerative and divisive. If you start with n distinct clusters and iteratively reach to a point where you have only 1 cluster in the end, it is called agglomerative clustering. On the other hand, if you start with 1 big cluster and subsequently keep on partitioning this cluster to reach n clusters, each containing 1 element, it is called divisive clustering.



For divisive cluster read : <http://luthuli.cs.uiuc.edu/~daf/courses/probcourse/notesclustering.pdf>

**Comprehension - Hierarchical Clustering Algorithm**

Given below are five data points having two attributes x and y:

|  |  |  |
| --- | --- | --- |
| Observation | **x** | **y** |
| 1 | 3 | 2 |
| 2 | 3 | 5 |
| 3 | 5 | 3 |
| 4 | 6 | 4 |
| 5 | 6 | 7 |

The distance matrix of the points, indicating the Euclidean distance between points, is as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Label | 1 | 2 | 3 | 4 | 5 |
| 1 | 0.00 | 3.00 | 2.24 | 3.61 | 5.83 |
| 2 | 3.00 | 0.00 | 2.83 | 3.16 | 3.61 |
| 3 | 2.24 | 2.83 | 0.00 | 1.41 | 4.12 |
| 4 | 3.61 | 3.16 | 1.41 | 0.00 | 3.00 |
| 5 | 5.83 | 3.61 | 4.12 | 3.00 | 0.00 |

Take the distance between two clusters as the minimum distance between the points in the two clusters. Based on this information, answer the following questions.

**Hierarchical Clustering**

How many clusters are there initially (before any fusions have happened)?

5

**Feedback :***Since this is* ***agglomerative clustering****, initially, all the points are 1 cluster.*

**Hierarchical Clustering**

Which two clusters will be fused first?

3 and 4

**Feedback :***These two points(clusters) have the minimum distance.*

**Hierarchical Clustering**

Which clusters will be fused in step two?

1 will be fused with the cluster(3, 4)

**Feedback :***The distance between points 1 and 3 is 2.24 units, which is the minimum among all the new clusters. Hence they will be joined now.*

**Hierarchical Clustering**

How many total clusters are there right after point number 1 fuses with the cluster(3, 4)?

3

**Feedback :***Since three points have fused into 1 cluster, total clusters left are (1,3,4) - (2) - (5).*

**Hierarchical Clustering**

Which clusters will be fused after 1 fuses with (3, 4)?

2 will fuse with the cluster (1, 3, 4)

**Feedback :***The distance of point 2 from point 3 is 2.83 units, whereas the minimum distance of point 5 from the cluster (1,3,4) is 3 units.*

**Hierarchical Clustering**

What happens in the last step of the algorithm?

5 fuses with ( 1, 2, 3, 4)

**Feedback :***Since all the other points are already part of one cluster, the last point will also join that cluster at this step.*

**Hierarchical Clustering**

Can you use the dendrogram to make meaningful clusters? (By looking at which elements leave and join at what height)

**Suggested Answer**

*Yes. It is a great tool. You can look at what stage an element is joining a cluster and hence see how similar or dissimilar it is to the rest of the cluster. If it joins at the higher height, it is quite different from the rest of the group. You can also see which elements are joining which cluster at what stage and can thus use business understanding to cut the dendrogram more accurately.*

**Types of Linkages**

In our example, we took the minimum of all the pairwise distances between the data points as the representative of the distance between 2 clusters. This measure of the distance is called single linkage. Apart from using the minimum, you can use other methods to compute the distance between the clusters.

Let’s see once again the different types of linkages.

* **Single Linkage:**Here, the distance between 2 clusters is defined as the shortest distance between points in the two clusters
* **Complete Linkage:**Here, the distance between 2 clusters is defined as the maximum distance between any 2 points in the clusters
* **Average Linkage:**Here, the distance between 2 clusters is defined as the average distance between every point of one cluster to every other point of the other cluster.

The choice of the linkage varies the shape of the dendogram. Taking the **single linkage** which means the smallest possible distance between the two clusters leads to the generation of very loose clusters. This means the intra cluster variance would be very high.

Taking the **complete linkage** which takes into account the maximum possible pairwise distance between the two clusters leads to generation of stable and close-net clusters.

You have to decide what type of linkage should be used by looking at the data. One convenient way to decide is to look at how the dendrogram looks. Usually, single linkage type will produce dendrograms which are not structured properly, whereas complete or average linkage will produce clusters which have a proper tree-like structure. You will see later what this means when you run the hierarchical clustering algorithm in Python.

**Hierarchical Clustering**

Select the appropriate option which describes the Complete Linkage method.

Top of Form



In complete linkage hierarchical clustering, the inter cluster distance is defined as the shortest distance between two points (one point in each cluster).



In complete linkage hierarchical clustering, the inter cluster distance is defined as the longest distance between two points (one point in each cluster)

**Feedback :***In the complete linkage, inter cluster distance is calculated as the maximum distance between 2 points (one in each cluster), However, the point is assigned to a new cluster basis it’s minimum distance from the clusters*

Bottom of Form

**Hierarchical Clustering**

Select the points which get clustered in the first iteration.(First iteration is defined as the first merging of clusters - ie, from n clusters to n-1 clusters)

Top of Form



A, B



B, C



C, D

**Feedback :***Look at the distance between 2 points. You will see that the minimum distance is 2.2 units, which is between C and D*

Bottom of Form

**Hierarchical Clustering**

Select the points which get clustered in the second iteration. Use single linkage method

Top of Form



A, B



A, F



B, E



A, E

**Feedback :***You can see that the minimum distance now is between point A and E, which is 3.2 units.*

**Correct**

Bottom of Form

**Hierarchical Clustering**

How many iterations are required to form the final single cluster?

Top of Form



2



3



4



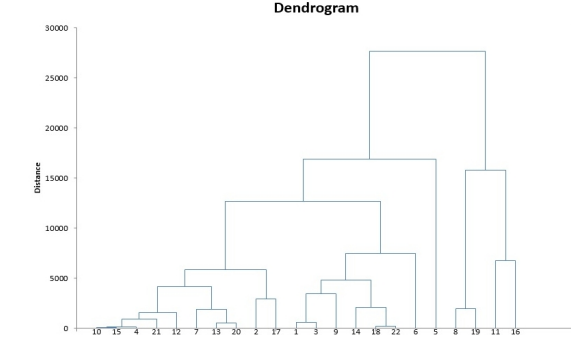
5

**Feedback :***Initially, n clusters are made and in each iteration, the number of clusters gets reduced by 1. So the number of iterations required is n-1. Here n = number of points = 6. So the correct answer is 5.*

Bottom of Form

One major advantage is that you do not have to pre-define the number of clusters. However, since you compute the distance of each point from every other point, it is time-consuming and needs a lot of processing power.

Now look at the following dendrogram and answer the questions that follow.



**Hierarchical Clustering**

Consider the above dendrogram for agglomerative clustering and answer the following questions.

 Find number of clusters if threshold value is 10000. (refer fig)

5

**Feedback :***Draw a horizontal line at that height. It cuts 5 vertical lines, all of which represent a cluster.*

**Correct**

**Hierarchical Clustering**

Find the threshold value if the no. of clusters to be formed is 4. (refer fig)

15000

**Feedback :***Look at the height at which a horizontal line will cut 4 vertical lines.*

# **PCA – Principal Component Analysis**

**Principal component analysis** (PCA) is one of the most commonly used dimensionality reduction techniques in the industry. By converting large data sets into smaller ones containing fewer variables, it helps in improving model performance, visualising complex data sets, and in many more areas.

Fundamentally, PCA is a dimensionality reduction technique, i.e., it approximates the original data set to a smaller one containing fewer dimensions. To understand it visually, take a look at the following image.  
****

In the image above, you can see that a data set having N dimensions has been approximated to a smaller data set containing 'k' dimensions. In this module, you will learn how this manipulation is done. And this simple manipulation helps in several ways such as follows:

* For data visualisation and EDA
* For creating uncorrelated features that can be input to a prediction model:  With a smaller number of uncorrelated features, the modelling process is faster and more stable as well.
* Finding latent themes in the data: If you have a data set containing the ratings given to different movies by Netflix users, PCA would be able to find latent themes like genre and, consequently, the ratings that users give to a particular genre.
* Noise reduction
* **Multicollinearity**
* Which of the following is **not true** about multicollinearity?
* Top of Form
* 
* Multicollinearity generally occurs when there are high correlations between two or more predictor variables.
* 
* **In the case of multicollinearity, some of the predictor variables can be used to predict some other predictor variables.**
* **Feedback :**
* *This is correct.*
* **Incorrect**
* 
* An easy way to detect multicollinearity is to calculate correlation coefficients for all combinations of predictor variables.
* 
* Multicollinearity helps in better convergence of a regression problem.
* **Feedback :**
* *Multicollinearity leads to incorrect estimates of the model parameters. Hence, never helps in a better convergence*
* Bottom of Form

In simple terms, dimensionality reduction is the exercise of dropping the unnecessary variables, i.e., the ones that add no useful information. Now, this is something that you must have done in the previous modules. In EDA, you dropped columns that had a lot of nulls or duplicate values, and so on. In linear and logistic regression, you dropped columns based on their p-values and VIF scores in the feature elimination step.

Similarly, what PCA does is that it converts the data **by creating new features from old ones**, where it becomes easier to decide which features to consider and which not to.

PCA is a statistical procedure to convert observations of possibly correlated variables to ‘principal components’ such that:

* They are **uncorrelated** with each other.
* They are **linear combinations** of the original variables.
* They help in capturing maximum **information** in the data set.

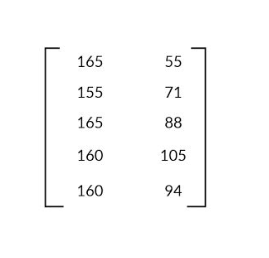
Now, the aforementioned definition introduces some new terms, such as ‘**linear combinations**’ and ‘**capturing maximum information**’, for which you will need some knowledge of linear algebra concepts as well as other building blocks of PCA. In the next segment, we will start our journey in the same direction with the introduction of a very basic idea: the **vectorial representation of data**.

**Vectorial Representation of Data**

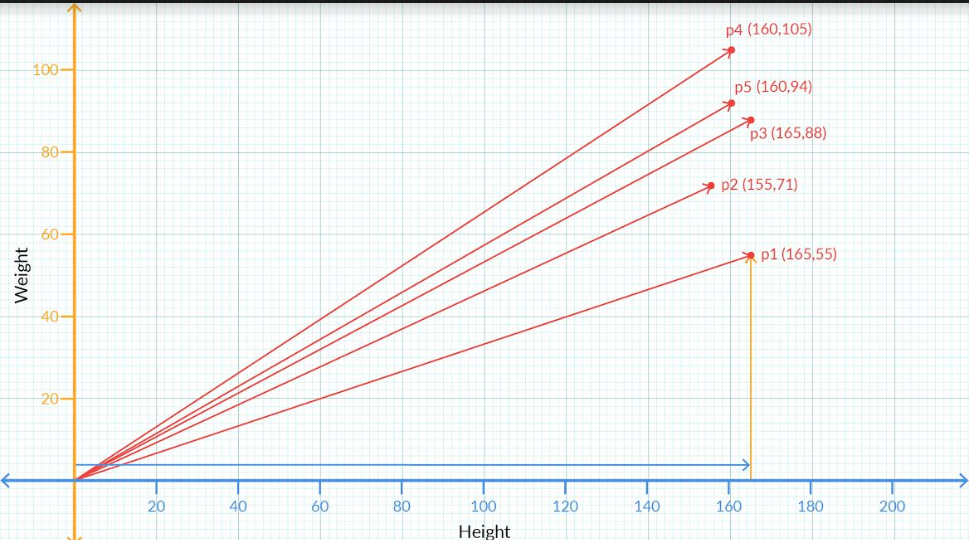
consider the following data set containing the height and weight of five patients:



 The height and weight information can be represented in the form of a matrix as follows

****

with each row representing a particular patient's data and each column representing the original variable. Geometrically, these patients can be represented as shown in the following image:

****

**Vector Representation**

The vector associated with the first patient is given by the values (165, 55). This value can also be written in the following way:

1. 1. A column containing the values along the rows. This is also known as the column-vector representation.  
   [165

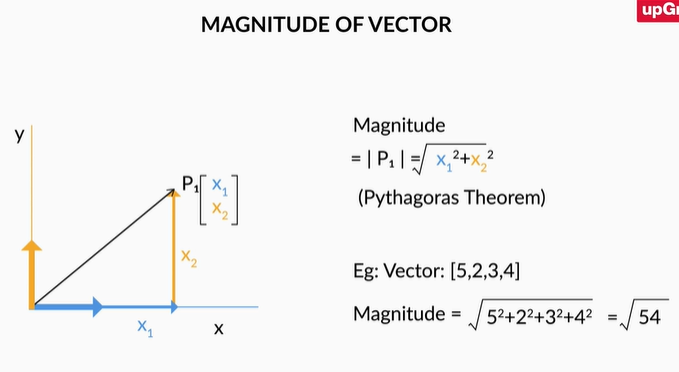
55]

1. As a transpose of the above form. Essentially, it is the same column vector but now written as a transpose of a row vector.  
   [165,55]T  
   [Note: Transpose is something you must have learnt in your Python for DS  module. If you need some brushing up on this topic, you can take a look at this[link](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.transpose.html)]
2. In terms of the basis vectors   
   This is something which you'll learn in detail in later segments. To give a brief idea, the vector (165,55) can also be written as 165**i** +55**j**, where **i** and **j** are the unit vectors along X and Y respectively and are the basis vectors used to represent all vectors in the 2-D space.

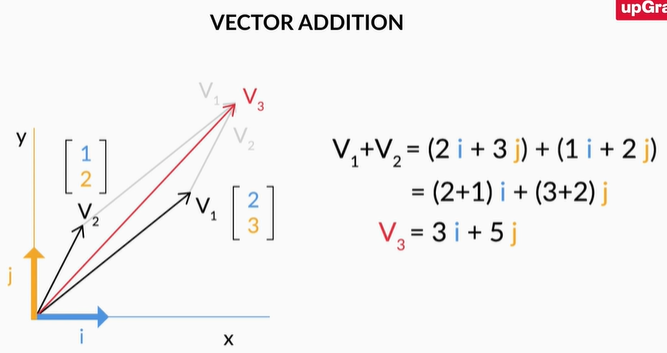
**Vector Representation for n-dimensional data**

Each vector will contain values representing all the dimensions or variables in the data. For example, if there was an age variable also included in the above dataset and the first patient had an age of 22 years, then the vector representing him would be written as  (165, 55, 22). Similarly, if the dataset had 10 variables, there would be 10 dimensions in the vector representation. Similarly, you can extend it for n dimensions or variables.

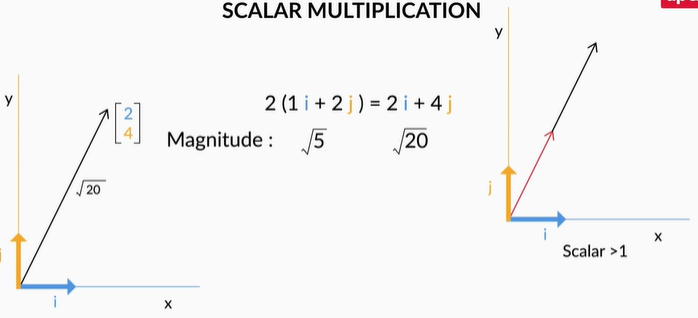
Now, these vectors have certain properties and operations associated with them.



**Vectors have a direction and magnitude**  
Each vector has a direction and magnitude associated with it. The direction is given by an arrow starting from the origin and pointing towards the vector's position. The magnitude is given by taking a sum of squares of all the coordinates of that vector and then taking its square root.  
  
For example, the vector (2,3) has the direction given by the arrow joining (0,0) and (2,3) pointing towards (2,3). Its magnitude is given by  √22+32=√13.  
  
Similarly, for a vector in 3 dimensions, say (2,-3,4) its direction is given by the arrow joining (0,0,0) and (2,-3,4) pointing towards (2,-3,4). And as in the 2D case, we get the magnitude of this vector as  √(2)2+(−3)2+(4)2=√29 .



1. **Vector Addition**  
   When you add two or more vectors, we essentially add their corresponding values element-wise. The first element of both the vectors get added, the second element of the both get added and so on.  
   For example, if you've two vectors say   
   V1=(2,3) and V2=(1,2) then   
   V1+V2=(2+1,3+2)=(3,5).
2. In the**i, j**notations that we introduced earlier, the above addition can be written as V1+V2=(2i+3j)+(i+2j)=(2+1)i+(1+2)j=3i+5j  
   Similarly, this idea can be extended to multiple dimensions as well.

**Scalar Multiplication**  
If you multiply any real number or scalar by a vector, then there is a change in the magnitude of the vector and the direction remains same or turns completely opposite depending on whether the value is positive or negative respectively.

**Unit Vectors**

Unit vectors are those vectors that have a unit magnitude and are signifiers of a particular direction. To find a unit vector along the direction of another vector, you divide that vector by its magnitude.

For example, if there is a vector A=Axi+Ayj whose magnitude is M, then a unit vector along the direction of A is given by AxMi+AyMj

What is the unit vector along the direction of the vector 3i + 4j?

 Top of Form



0.8i - 0.6j



0.8i + 0.6j



0.6i - 0.8j



**0.6i + 0.8j**

**Feedback :**

*The magnitude of the vector is given by*√(32+42)=√25=5*). Therefore a unit vector along A is given by (3/5)****i****+ (4/5)****j****= 0.6****i****+0.8****j***

Bottom of Form

**Vector Addition**

Let v1 and v2 be two vectors given by v1=i+2j and v2=2i−3j. Find the value of 2⋅v1+3⋅v2.

Top of Form



2i+3j



5i+4j



8i−5j

**Feedback :**

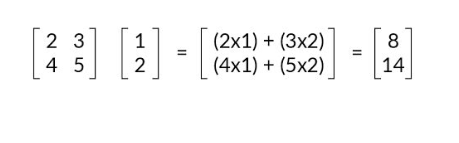
*2\*v1 +  3\*v2 = 2(i + 2j) + 3(2i -3 j) = 2i +4j + 6i - 9j = 8i -5j*

The process of **matrix multiplication** is quite simple, and it involves the element-wise multiplication followed by addition of all the elements present in it. The one key rule that it must satisfy is when you multiply 2 matrices, say A and B, the number of columns of A must equal the number of rows in B. Visually, you can take a look at the following image to get the idea of how that should be:



As shown in the example, since the number of columns in the first matrix and the number of rows in the second column are equal to 4, matrix multiplication is possible and the resultant matrix has a shape of 5 x 6.

The element-wise multiplication followed by addition is also pretty straightforward as can be seen in the following example:



**Matrix Multiplication**

If matrix dimensions of two matrices are given, say,  A =a1 x a2 (dimensions) and  B=b1 x b2 (dimensions), then matrix multiplication is valid if \_\_?

a2=b1

**Feedback :**

*We need the number of columns of A, i.e a2, to be equal to the number of rows of B, i.e b1, for matrix multiplication to be possible.*

To understand what the inverse of a matrix is, let's take a look at the following example:

Let's say you have 2 matrices A and B such that  A= [2 −1

1 1]

and B = [1/3 1/3

−1/3 2/3]

If you multiply B with A like this: B x A, you get the following result-  [1 0

0 1]

The matrix that you got after the multiplication above is also known as an **Identity matrix**. In matrix notation, it serves the same function as that of the number 1 in the real number system. To establish an analogy, in the real number system if you multiply any number by 1, you get the number itself. Similarly, when you multiply any matrix with the identity matrix, also denoted by I, you get the same matrix once again. ( You can calculate this and verify yourselves)

Now taking the analogy of the real number system, when you multiply 2 numbers  a and b and it comes out to be 1, i.e.

a×b=1 then a and b are called reciprocal of each other.

In the matrix world, if you have two matrices A and B , and their multiplication results in the identity matrix  I, i.e.

B  x A = I, then A and B are called **inverses** of each other.

The inverse of A is also written as A−1.  Therefore B=A−1. In a later segment, we'll get to know as to how these inverses are useful.

Note that A−1A=I=AA−1. You can verify this using the above matrix.

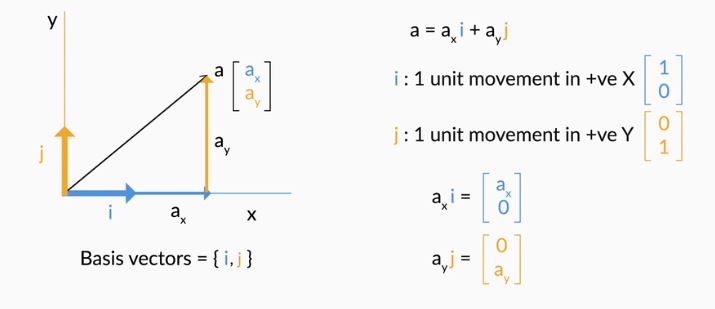
Since i and j themselves represent **(1,0)** and **(0,1)**, you can represent any vector in the 2-D space with these i and j vectors.

 Any vector **'**a**'** (ax,ay) can be represented in a 2-D space, using the following notation:

a=axi+ayj   
or

a=ax⋅[1 0]+ay⋅[0 1]

Visually, it can be represented as follows:



 For example, a vector A (2,3) can be written as 2⋅[1 0]+3⋅[0 1]. In order to obtain the vector A, we scaled **i** by 2 and**j** by 3 and then finally added them up.

This scaling and adding the vectors up to obtain a new vector is also known as a **linear combination**.

For the patients' dataset that we had earlier, we can denote each patient vector by the following notation:

https://images.upgrad.com/fa100728-9c13-4019-ba55-530abb6ab52a-PCA%20-%20patient%20notation.JPG

Therefore, now we can say that Patient 1 is represented by 165(1 cm,0) + 55(0,1kg). And similarly, we can express other patients' information as well.

The **basic definition** of basis vectors is that they're a certain set of vectors whose linear combination is able to explain any other vector in that space.

In a 2D space, the standard basis vectors are given by [1 0] and [0 1]. In a 3D space, the same are given by ⎣1 0 0⎦, ⎣0 1 0⎦ and ⎣0 0 1⎦. As you can see, an n-dimensional space or a dataset having n variables would have n standard basis vectors.

**Basis Vectors**

The vectors i and j are the basis of two-dimensional space. Which of the following is true?

Top of Form



The magnitude of both these vectors is unity



The linear combination of i and j can represent any point on the two-dimensional space



**i** cannot be expressed in terms of**j** and vice versa



**All of the above**

**Feedback :**

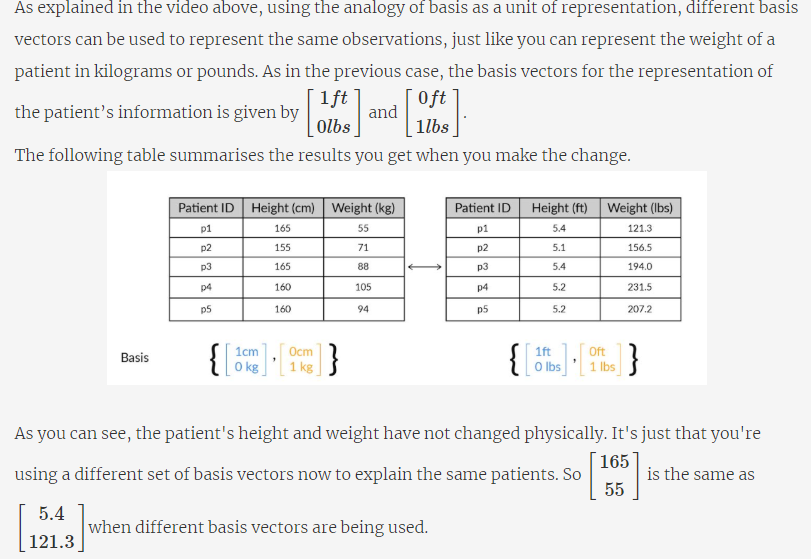
*i =(1,0) and j=(0,1) are unit vectors with magnitudes of 1.*

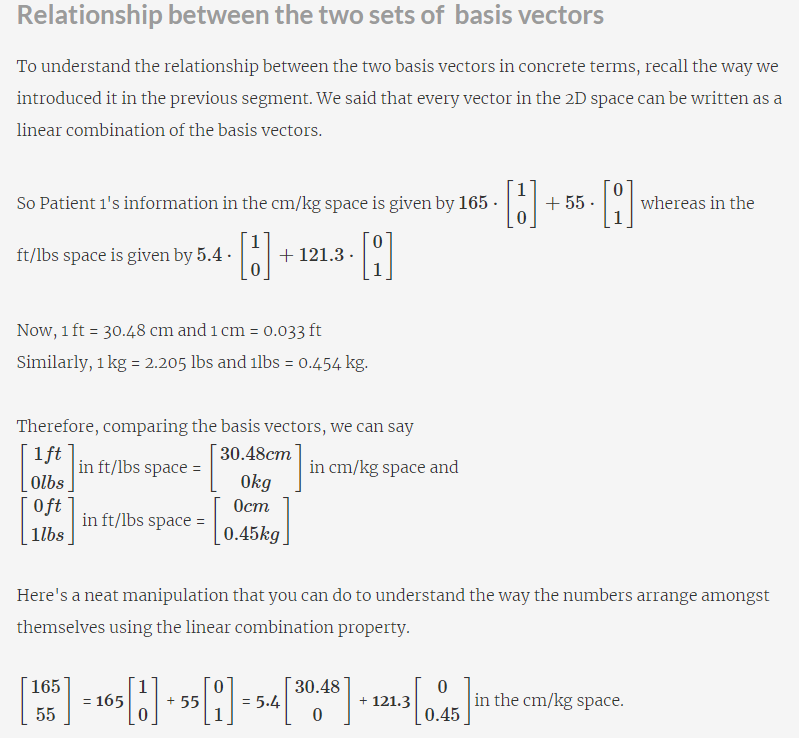
*Any point on the 2-D space is a linear combination of i and j.*

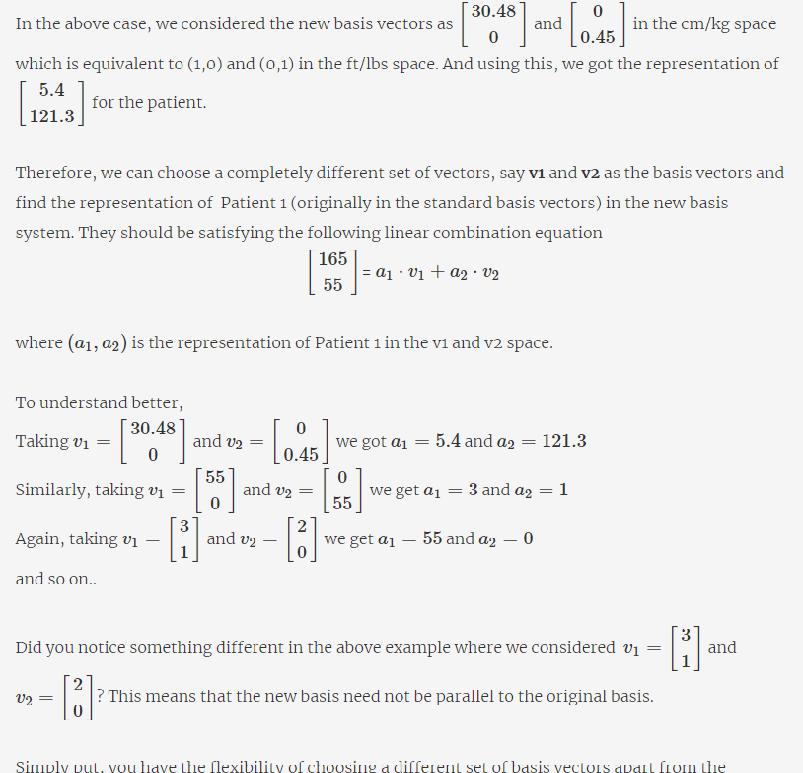
*i and j are orthogonal/ perpendicular vectors, and hence, one can't be represented by the other.*

**Correct**

Bottom of Form







Simply put, you have the flexibility of choosing a different set of basis vectors apart from the standard basis vectors that are provided to you to represent your information. The information won't change, just the numbers representing the information would change.

**Different Basis Vectors**

Let's say that you're representing the vector [120 35] using a new set of basis vectors [1 c] and [d 0.25], where c and d are unknowns.

In this new representation, that vector in the original standard basis is now written as [30 20].

From this information, find out the values of c and d

Top of Form



c =1,  d = 4.5

**Feedback :**

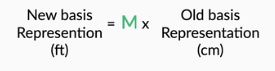
*We have*a1=30,a2=20,v1=(1,c) and v2=(d,0.25)

*Substituting the values given above in the equation*[120 35]=a1∗v1+a2∗v2*you'll get the following*[120 35]=30∗[1 c]+20∗[d 0.25]

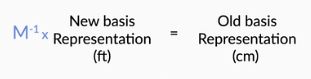
*Thus our equations are*120=30+20∗d and 35=30∗c+20∗0.25

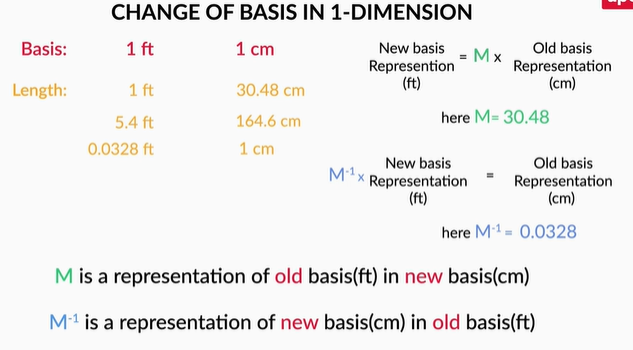
*Solving both the equations give the value of c as 1 and d as 4.5*

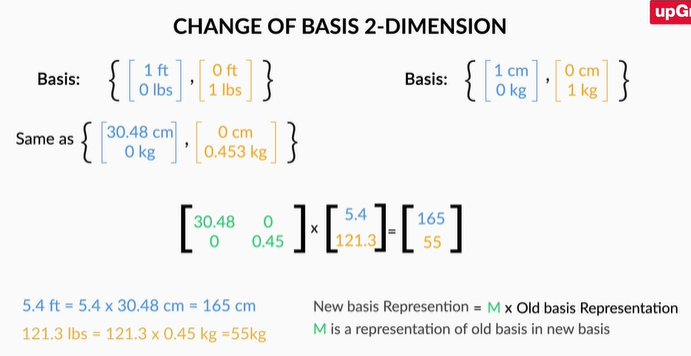
when you have one dimension, the calculations for the change of basis are pretty straightforward. All you need to do here is to multiply the factor M which gives you the method of transforming from one basis to another.

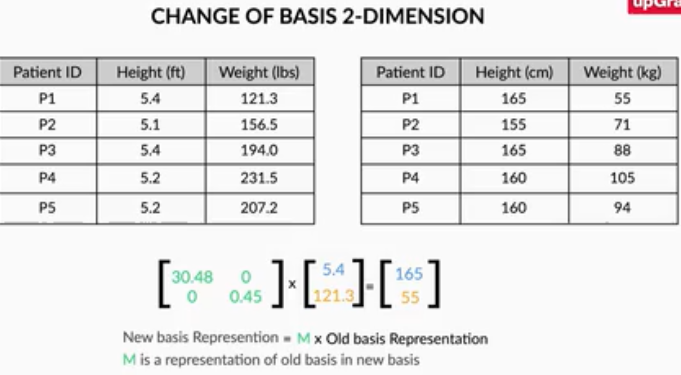


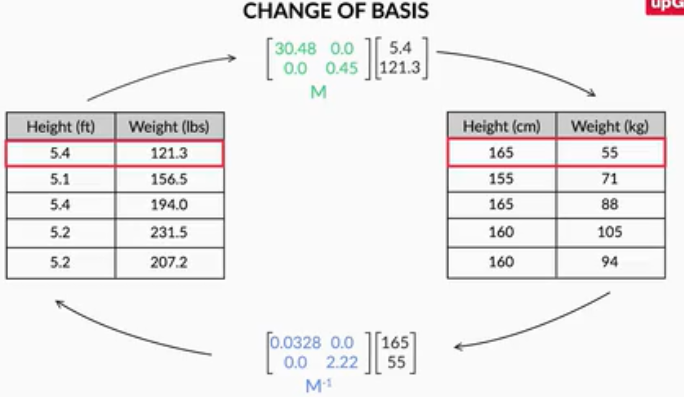
And to go in the opposite direction you simply have to divide by that factor M. Note that here M is a simple scalar since there is only one dimension involved.











As you saw in the demonstration above the original matrix gets inversed when we want to go the other way around. Therefore, the equation remains the same in both the cases, but here the M−1 would mean the inverse of the matrix rather than a simple reciprocal.

So to summarise what you saw in the video,  M=[30.48 0

1. 0.454]

which is the matrix that shows the change of basis from ft/lbs to cm/kgs and

M−1=[0.0328 0

1. 2.205]

which shows the changes from cm/kgs to ft/lbs

**Change of Basis**

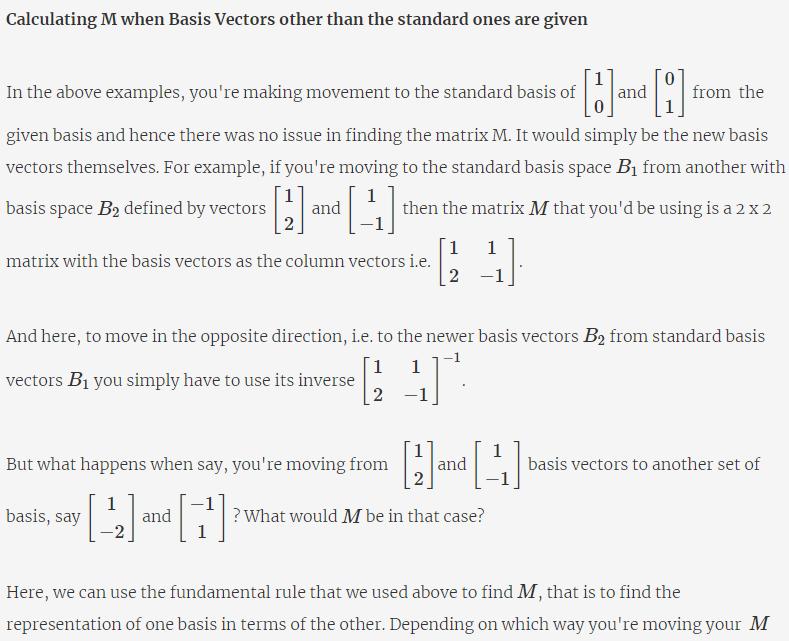
Let's say you have a set of basis vectors B given by [1 2] and [1 −1]. What is the change of basis matrix M when you move from this basis to the standard basis of i and j?

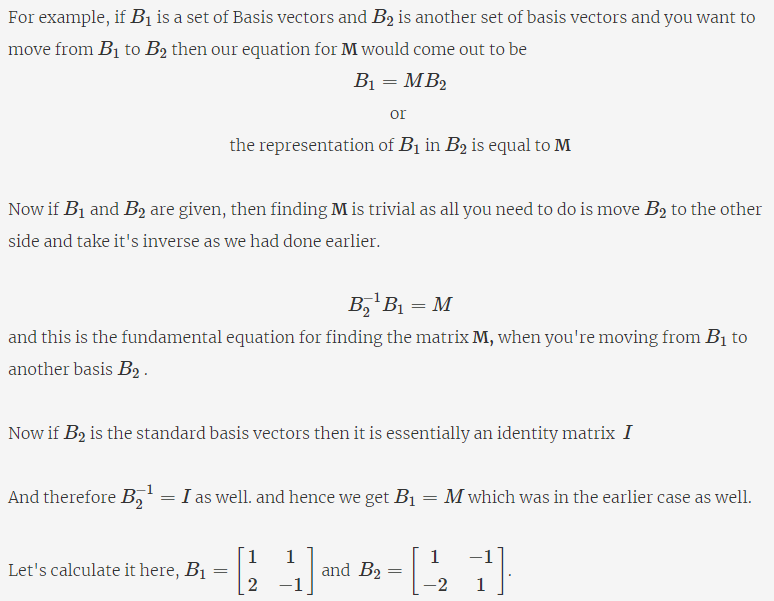
[1 1

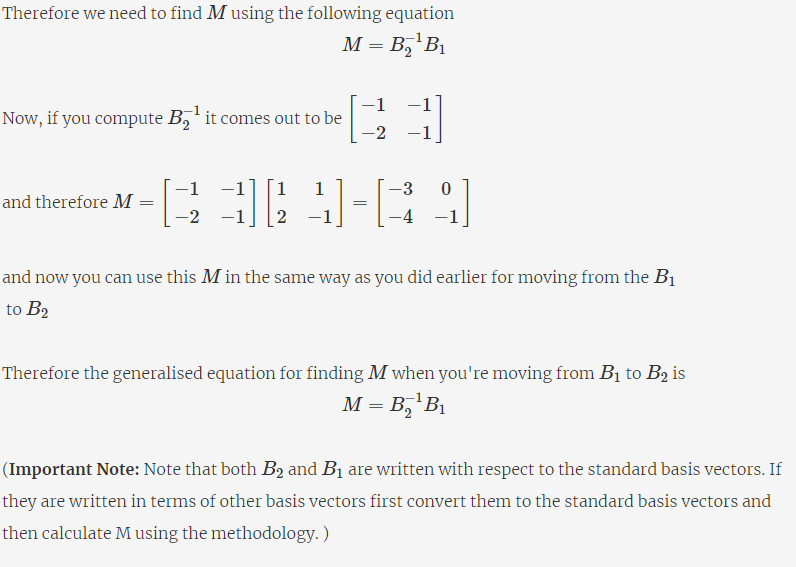
2 −1]

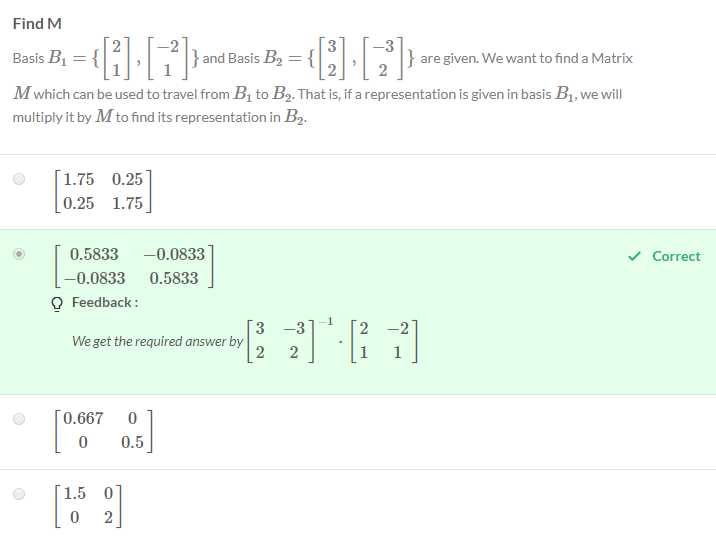
**Feedback :**

*The change of basis matrix M represents the old basis in terms of the new basis. Here, the old basis is given by B and the new basis is the standard basis. Hence, M will be the same as B with the basis vectors written along the columns.*









**Other M**

Now in the previous question, what if you knew a vector V=[11 13] in B2 and you were asked to find its representation in B1?

Top of Form



[5.333 6.667]



[22.5 25.5]

**Feedback :**

*In this case we were traveling from*B2*to*B1*. So we will find out answer by*B−11×B2×V1

**Correct**



**Comprehension: I**

In this segment, we will drive home the concept of Change in Basis. For this imagine you went to a different planet for exploratory reasons, to find intelligent life there. You met an alien there and you started showing it where you came from and location of Earth from that planet.

You realise the standard Basis there is not the same as yours. You use basis B1={i,j}while the alien uses the basis B2={i−2j,i+j}.

You want to show them where the Earth is from that planet. You know the position in your Basis, i.e. [21 12] in B1 . To communicate this location you will have to tell them this same location or vector in their Basis B2.

Earlier we saw that if the vector is represented as [x y] in a Basis {v1,v2} , it just means the vector  V=x \* v1+y\* v2. So if a vector is represented in Alien Basis as [x y] that is same as saying the vector is x \* [1 −2] + y \* [1 1]. We want this vector to be [21 12]. Which is same as solving the following two equations for **x** and **y**

 1⋅x+1⋅y=21

−2⋅x+1⋅y=12

That is solving the matrix equation

[1 1,−2 1]⋅[x y]=[21 12]

Here we note that the 2×2 matrix is nothing but the representation of Alien's Basis in terms of your Basis. The first column vector of the matrix is the first basis vector of B2, Alien's Basis, and second column is the second vector.

Thus multiplying any vector with this matrix will give us the representation of that vector in your Basis. To go in the other direction, that is, given vector in your Basis you want to find its representation in Alien's Basis B2, you only have to take the inverse of the matrix.

[x y]=[1 1,−2 1]−1⋅[21 12]

This will give us the required [x y].

# **Discussions**

1) You were given B1 and B2, and were asked to find M which can be used to travel from B1 to B2.i.e. B1 = M.B2. You went ahead and calculated M which is nothing but factor for B2 basis. I.e if you have a vector in B1 basis, you can multiply it by M to find corresponding vector in B2 basis.

2) Now, in Q2, you are given vector in B2 basis and you are asked to find corresponding vector in B1 basis. i,e, B2 = M1.B1. (Note - This is M1 is not same as M you calculated in Q1). This is the factor used for converting B2 vector in B1 i.e. travel from B2 to B1.

So to solve this problem you can.

1) Calculate M1 i.e. (inv(B1) . B2) and multiply it by V

OR

2) Calculate inv(M) and multiply it by V.

B1 = np.array([[2,1],

               [-2,1]])

#

B2 = np.array([[3,2],

               [-3,2]])

#

B1 = B1.T

B2 = B2.T

#

# Calculate B2 inverse

B2i = np.linalg.inv(B2)

#

# B1 = M.B2

# Calcualte M

M = B2i @ B1

print("M is: ")

print(M)

#

# V in basis2 is

V = np.array([11,13])

#

# Option1: We can take M inverse and multiple by V to get V in basis1

M1 = np.linalg.inv(M)

print("="\*25)

print("V in B1 using option1")

print(M1 @ V)

#

# Option2: B2 = M.B1

B1i = np.linalg.inv(B1)

M2 = B1i @ B2

print("="\*25)

print("V in B1 using option2")

print(M2 @ V)

**# Output**

M is:

[[ 0.58333333 -0.08333333]

[-0.08333333 0.58333333]]

=========================

V in B1 using option1

[22.5 25.5]

=========================

V in B1 using option2

[22.5 25.5]

Bottom of Form

**Summary: I**

That was end of quite a hectic session! Here's a summary of what you've learnt so far.

First, you came to know about PCA and how it is essentially a dimensionality reduction technique.  You saw the necessity for doing PCA in a couple of situations like

* A predictive model setup where there are a lot of features to eliminate
* A dataset where you needed to perform EDA and Data Visualisation.

You understood how PCA not only helps in resolving the above two issues but has applications in several other areas like noise reduction, finding latent Themes and so on. Then you got a brief understanding of its definition:

It is a statistical procedure that finds principal components or directions that are:

* Linear combination of the original variables
* Are uncorrelated
* Capture Maximum information in the dataset.

After that, you went ahead and learnt some essential linear algebra concepts like vectors and their properties along with their associated operations. Then you studied another tool called matrix multiplication and matrix inverse, both of which proved invaluable in understanding the first fundamental building block of PCA: **Basis**

Basis is essentially the fundamental units in which you express your data. As you saw in the lecture videos, it is similar to how we use units for physical objects to measure things like height, weight, temperature, etc.

In vectors and vector spaces, we use basis vectors to represent the points in space. You understood how every observation in the space can be represented by **scaling** and **adding**the scaled basis vectors. This process is also called**a linear combination**.

Then you learnt one of the key ideas that helped you connect basis vectors and the idea of dimensionality reduction: using different basis vectors to represent the same points.

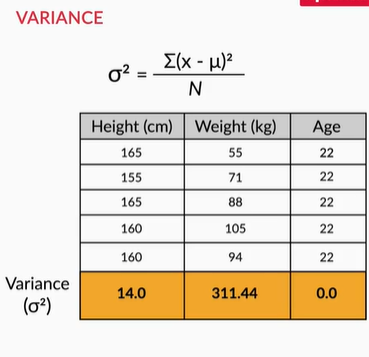
From there, you learnt how to change from one basis space to other using matrices. Here's a list of rules to help you revise the same.

1. If you're moving from a basis space B  to the standard basis, then the change of basis matrix M is the same as the basis vectors of B written as its column vectors. Therefore, if there is a vector v  represented in B and you want to find its representation in the standard basis, then you'd have to perfrom Mv.
2. If you want to go the other way around, where you have v represented in the standard basis and want to find its representation in B  you multiply it by its inverse  - M−1v
3. Finally, if you want to find the change of basis matrix M where you move from two non-standard basis vectors - say from B1 to B2 then you can get that by calculating this value - B−12B1. Note that in all the above cases, the basis vectors should be represented in the same units.

**Vector V represented as [ 1 1 ] in basis B = { [ 1 1 ] , [ 2 1 ] } . What is the vector in our standard basis B s t d = { i , j }**

1. We are given a vector V in B basis
2. Then we are also provided with B basis and we know the other basis is standard basis
3. So, to travel from B basis to standard basis i.e. factor to convert from B basis to Std basis B = M.I ( I is identity standard basis). So, B = M i.e. M = B
4. So if we have to convert V given in B basis to standard basis we multiply in by M
5. Please see snippet below:
6. Note - Although I am explicitly taking inverse of Bstd, it is not required. Bstd is Identity matrix.
7. B = np.array([[1,1],
8. [2,1]])
9. #
10. B = B.T
11. #
12. Bstd = np.array([[1,0],
13. [0,1]])
14. #
15. Bstd = Bstd.T
16. #
17. # B=M.Bstd
18. # M = inv(Bstd).B
19. M = np.linalg.inv(Bstd) @ B
20. #
21. V = np.array([1,
22. 1])
23. #
24. print(B @ V)
25. **# Output**
26. [3 2]

# **Variance as Information**

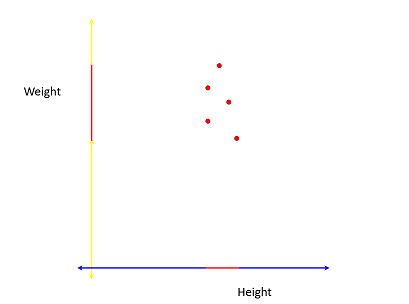


Variance in statistics (sigma variance) is simply sum of the squares deviations from the mean.

So the key takeaway from the above lecture is to measure the importance of a column by checking its variance values. If a column has more variance, then this column will contain more information.

**Geometrically Interpretation of  Variance**

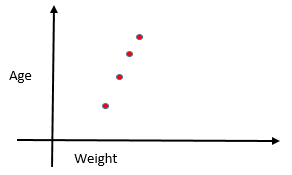
In the above example you saw that the variance of height was only 14, whereas that of weight was 311.14. This gave you an idea that Weight is a more important column than Height. Now, there is another elegant way of looking at variance geometrically. Take a look at the following image.



The red line on the Height and Weight axes show the**spread** of the projections of the vectors on those axes. As you can see here, the spread of the line is quite good on the Weight axis as compared to the Height axis. Hence you can say that Weight has more variance than Height. This idea of the spread of the data being equivalent to the variance is quite an elegant way to distinguish the important directions from the non-important ones.

**PCA: Variance**

Which axis captures more variance in the following plot between Age and Weight?



  Top of Form



**Age**

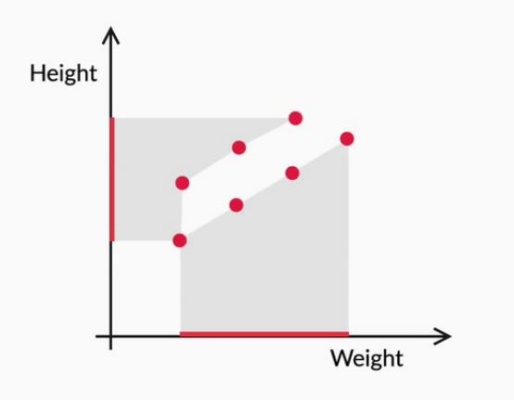
**Feedback :**

*When the farthest points of the data are projected on the axis, the length of the projection becomes proportional to the variance. In the given image, the length of the projection of the farthest points of data on the Y-axis is more than the length of the projection on the X-axis.*

# **Directions of Maximum Variance**

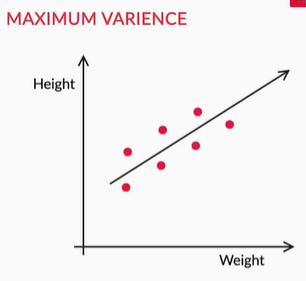
So you saw that when the variances are unequally distributed among the original features or columns i.e. some columns have much less variance than others, it is easier to remove those columns and do dimensionality reduction.

But what about the scenario when the variances are pretty similar? For example, take a look at the following image containing the height and weight information of a different group of patients.



As you can see, the spread along both the axes is quite comparable and therefore, you can't directly go and say that one direction is more useful than the other. What to do now?

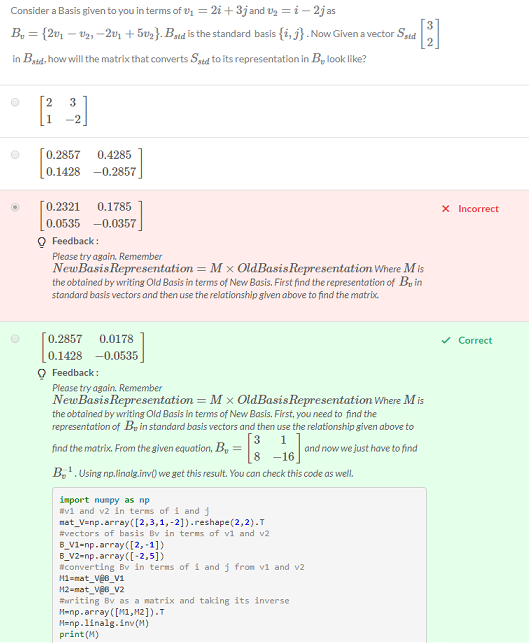
We can capture different direction where we can capture maximum variance. So, for example below

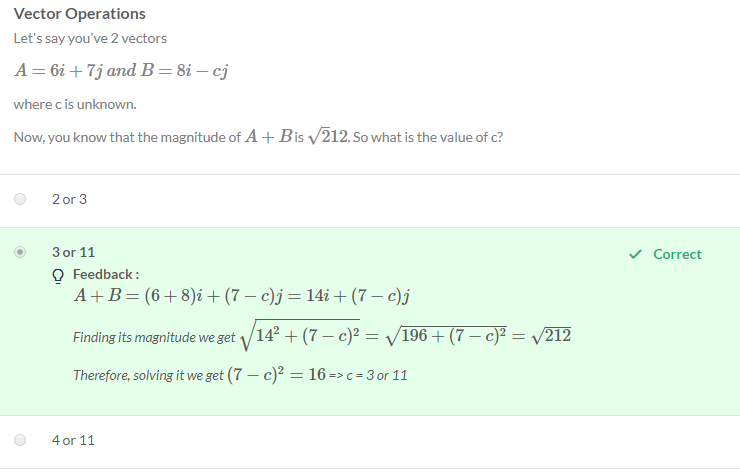


This is what PCA does. It figures out the best direction which provides the maximum variance in the data set. It changes the basis vectors in such a way that the new basis vectors capture the maximum variance or information.

The ideal basis vectors that we wanted has the following properties:

* They explain the directions of maximum variance
* When used as the new set of basis vectors, the transformed dataset is now suitable for dimensionality reduction.
* These directions explaining the maximum variance are called the **Principal Components** of our data.





**Graded Question**

Which of the following preprocessing steps is the most crucial before performing PCA?

Top of Form



**Standardisation of data**

**Feedback :**

*If variables are on a different scale (e.g. fractions and millions), then PCA (while trying to maximise the variance) will give higher importance to the variables with high variance simply because of scale. For example, if you change one variable from km to cm (increasing its variance), it may go from having little impact to dominating the first.*

Bottom of Form

# **The Algorithm of PCA**

This session will introduce you to the following topics:

* Covariance Matrix
* Eigendecomposition
* Algorithm of PCA

**Covariance Matrix**

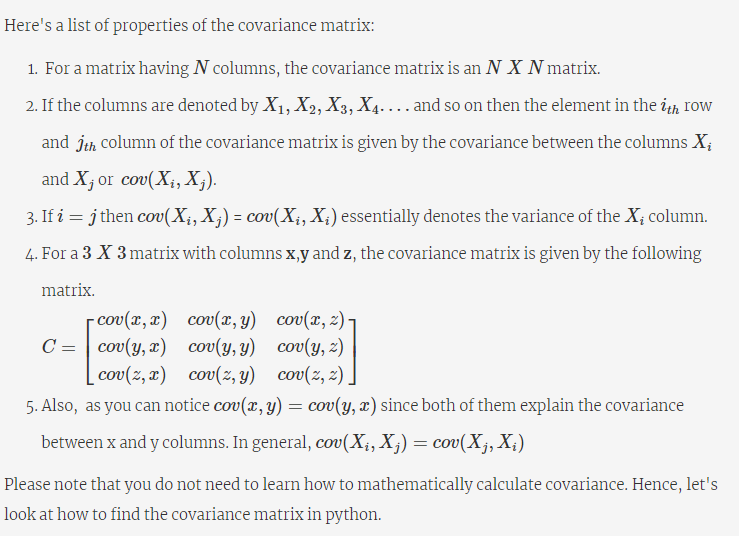
In the previous session, you learnt about how to find the variance of a particular column to measure its importance. However, this does not capture the inter column relationship or the correlations between the variables. This information is essential since we also want to make sure that the multicollinearity in our dataset is minimum. In order to capture the covariance or the correlations between the columns, we use the covariance matrix.

The covariance matrix is one such matrix that not only captures the variance but also the covariance information as well.

The covariance  between 2 columns X and Y is given by the following formula

cov(X,Y)=∑(Xi−¯X)(Yi−¯Y)/N

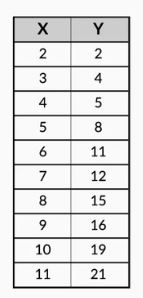
You need not calculate the values mathematically as there is a functionality in Python already available which shall be covered shortly. Nevertheless, you can check this [link](https://corporatefinanceinstitute.com/resources/knowledge/finance/covariance/) to further understand how this value is calculated.



As mentioned earlier, the reason you study the covariance matrix is to capture all the information in the dataset - variance along the columns as well as the inter-column covariance. Now using this matrix, we'll be able to deduce the directions of maximum variance and we'll be learning how to do that in the upcoming segment.

**Transforming the Covariance Matrix**

The given dataset A is as follows



and the associated covariance matrix is given by

cov(A)=[[9.16, 19.61] [19.61, 42.23]]

In this case, we've cov(x,x)=9.16 , cov(y,y)=42.23 and cov(x,y)=cov(y,x)=19.61

From here, we can observe that there is some correlation between the X and Y columns. Also we can see Y has little higher variance among the variables in column.

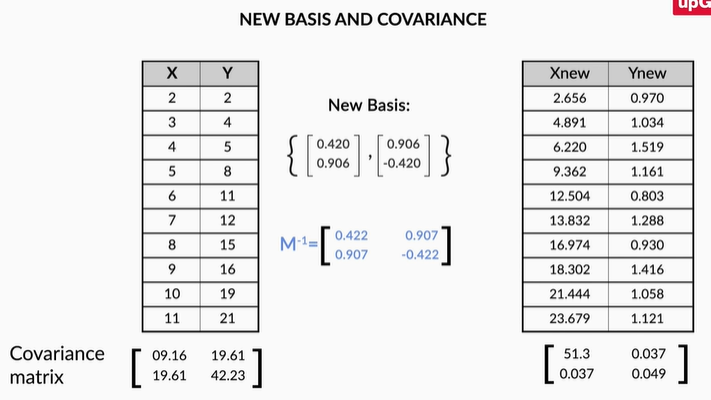
Let's see what happens to the covariance matrix when we take another basis to represent the same points

**Covariance Matrix in New Basis**

Let's say that the same friend, who earlier gave you the smart suggestion in the roadmap example comes around again and asks you to represent your data in the following basis

*New Basis=[[0.420,0.906],[0.906,−0.420]]*

He tells you that you'll observe something interesting with the covariance matrix when the data is represented in this basis. Let's check out your friend's claims.



Take an inverse of covariance matrix M, and multiply with our data to represent the data in the new basis.

So as per your friend's suggestion, you went ahead and transformed the dataset and computed the covariance matrix for the new dataset A′. The new covariance matrix is as follows:

*cov(A′)=[[51.3,0.03],[0.03,0.049]]*

As you can see now the covariance matrix has changed a lot. *cov(xnew,xnew)* is much higher than *cov(ynew,ynew)*.  One important thing that has happened here is that *cov(xnew,ynew)* is now ~0. What does that mean? This means that x and y columns have very less correlation.

Well, now that the 2 features are**nearly uncorrelated**, there is no dependence or correlation of one direction on the other. This process of converting the covariance matrix with only non-zero diagonal elements and 0 values everywhere else is also known as **diagonalisation**.

**Creating Uncorrelated Features**

Now you must be wondering "How does finding new basis vectors where the covariance matrix only has non-zero values along the diagonal and 0 elsewhere help us?"

Well now,

* Your new basis vectors are all uncorrelated and independent of each other.
* Since variance is now explained only by the new basis vectors themselves, you can find the directions of maximum variance by checking which value is higher than the rest numerically. There is no correlation to take into account this time. All the information in the dataset is explained by the columns themselves.

So now, your new basis vectors are **uncorrelated**, **hence linearly independent**, and **explain the directions of maximum variance.**

Do the above terms ring a[bell](https://learn.upgrad.com/v/course/376/session/49806/segment/273467)?

As a matter of fact, these basis vectors are the **Principal Components** of the original matrix. The algorithm of PCA seeks to find those new basis vectors that diagonalise the covariance when the same data is represented on this new basis. And then these vectors would have all the above properties that we require and therefore would help us in the process of dimensionality reduction.

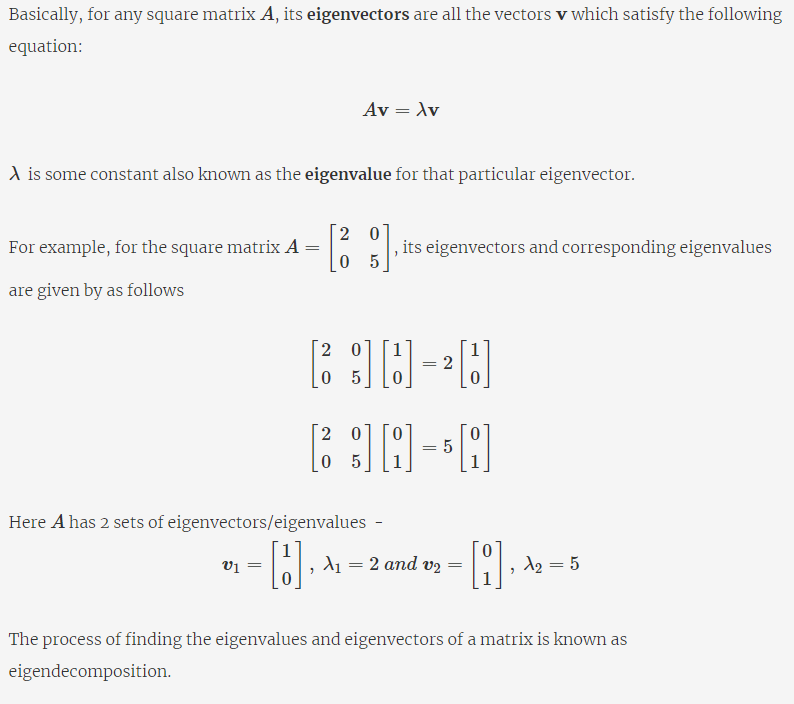
# **Eigendecomposition**

So, algorithm of PCA has three major steps

1. Construct Covariance Matrix to represent all the information of the data set
2. Perform eigen decomposition of the Covariance matrix
3. New Basis is the eigen vectors of Covariance matrix obtained in step 1

You need to do find something known as **eigenvectors** of the covariance matrix using a process called **Eigendecomposition**. These eigenvectors would be the new set of basis vectors in whose representation, the covariance matrix will be diagonalized. Therefore these would be the new principal components.

Here's a brief overview of what eigendecomposition is and how it is applied on the covariance matrix to give us the eigenvectors. Please note that though it is recommended that you understand the text below you can perform the eigendecomposition in a couple of steps in Python.



In general, if the size of matrix A is n ( i.e. it is a n x n matrix) then, there will be at most n eigenvectors that can be formed. As you saw in the above case, A is a 2 x 2 matrix and hence it had 2 eigenvectors.

**Note**: This is a brief overview of what eigenvectors and eigenvalues are and it is sufficient for you right now in the context of PCA. If you want to learn further about eigenvectors and how they're found out, please go through the additional link mentioned [here](https://learn.upgrad.com/v/course/376/session/49813/segment/273522).

**Eigendecomposition of Covariance Matrix**

Now, continuing from the previous segment, we wanted to find a new set of basis vectors where the covariance matrix gets diagonalised. It turns out that these new set of basis vectors are in fact the eigenvectors of the Covariance Matrix. And therefore these eigenvectors are the Principal Components of our original dataset. In other words, these eigenvectors are the directions that capture maximum variance.

But what about the eigenvalues? What do they signify?

Well, the **eigenvalues** are indicators of **the variance explained by that particular direction** or eigenvector. So higher is the eigenvalue, higher is the variance explained by that eigenvector and hence that direction is more important for us.

Therefore to summarise,

* The eigendecomposition of the covariance matrix C  yields us the eigenvectors v1,v2,v3....  with their corresponding eigenvalues as λ1,λ2,λ3.... ...
* When you use the eigenvectors as the new set of basis vectors and transform the original dataset to this new basis, your covariance matrix will now be diagonalised.
* These **eigenvectors** are the **Principal Components** of the original dataset. v1 is **Principal Component 1**, v2 is **Principal Component 2**and so on.
* The eigenvectors are ordered on the basis of their eigenvalues, to signify the variance explained by them. Or you can say λ1>λ2>λ3..... Higher the eigenvalue, higher is the amount of variance captured by the eigenvector. Hence the maximum variance is explained by Principal Component 1, the second-highest variance is explained by Principal Component 2 and so on.

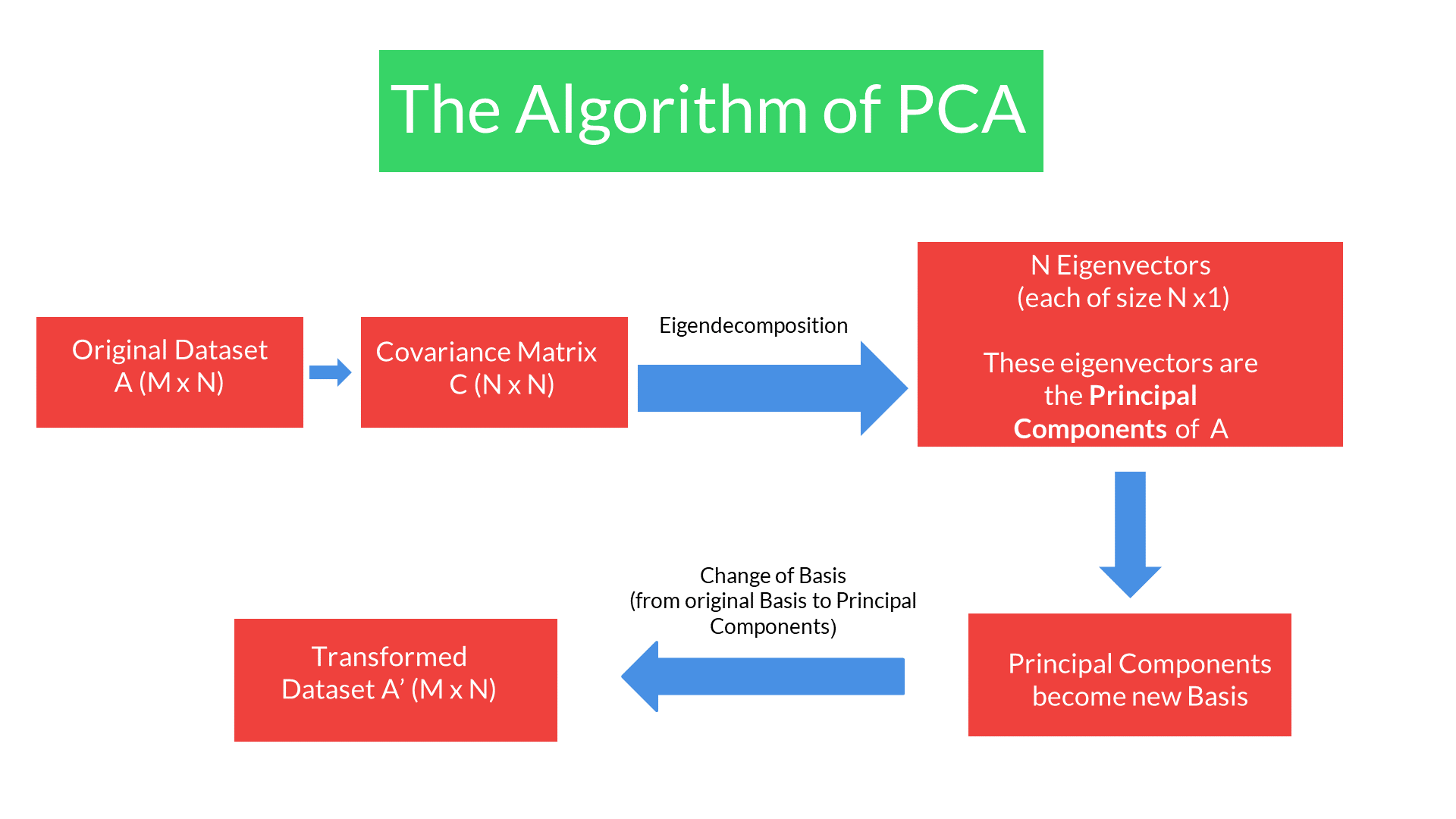
Let's go ahead and formalise the algorithm of PCA in the next segment and also check out a demonstration that puts it into use.

**Additional Links**

* Note that we haven't covered why doing an eigendecomposition on the covariance matrix yields us the Principal Components. This part is beyond the scope of the current module and therefore if you're curious about learning why this method works, you can take a look at this [link](https://math.stackexchange.com/questions/23596/why-is-the-eigenvector-of-a-covariance-matrix-equal-to-a-principal-component).
* Apart from eigendecomposition, there are more generalised algorithms that are used to perform PCA right now such as the **Singular Value Decomposition,**about which you can read from this [link](https://medium.com/data-science-group-iitr/singular-value-decomposition-elucidated-e97005fb82fa).

# **Algorithm of PCA**

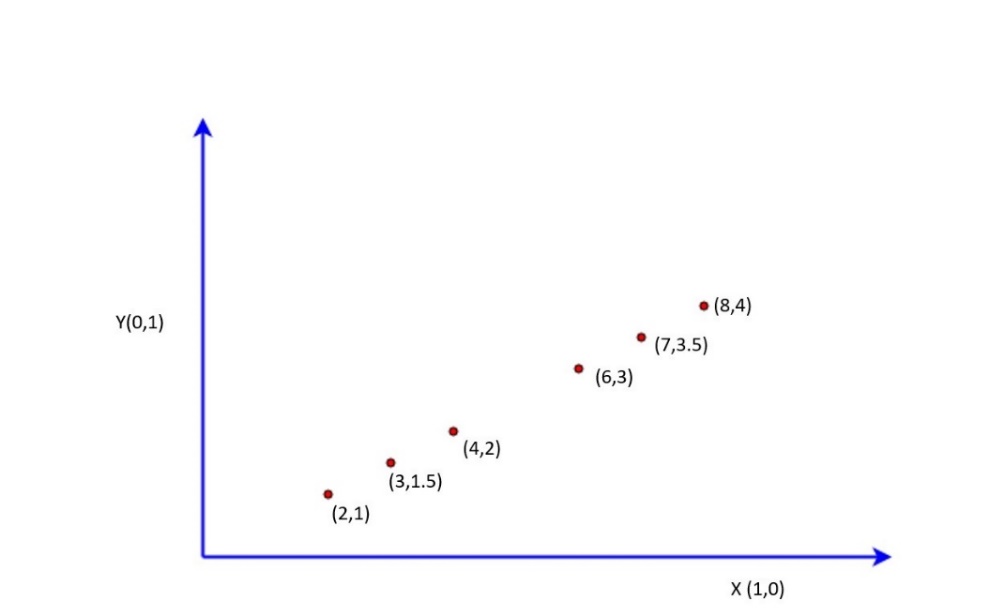
So, algorithm of PCA has three major steps

1. Construct Covariance Matrix to represent all the information of the data set
2. Perform eigen decomposition of the Covariance matrix
3. New Basis is the eigen vectors of Covariance matrix obtained in step 1
4. So formally, you now understand what the algorithm of PCA does. Here's a flowchart that summarises the important steps.
5. 
6. **The Algorithm of PCA**

**Demonstration**

Here's a short demonstration that shows how these directions are found. We'll be using the same roadmap example that you saw earlier. Earlier your friend found the new directions visually, without any additional computations. Let's see how PCA does the same using the algorithm that you just saw above.

Here we've the coordinates of the various locations in the original basis as shown in the image below:



The data points in matrix is A = [[2,1],[3,1.5],[4,2],[6,3],[7,3.5],[8,4]]

Now let's go ahead and perform the steps of the PCA algorithm on this dataset.

* Step 1: Finding the covariance matrix

           First, we need to compute the covariance matrix of A. This comes out to be the following.

           (You can use numpy.cov to find the covariance matrix as well)

 cov(A)=C=[[5.6,2.8],[2.8,1.4]]

 Step 2: Eigendecomposition of the covariance matrix

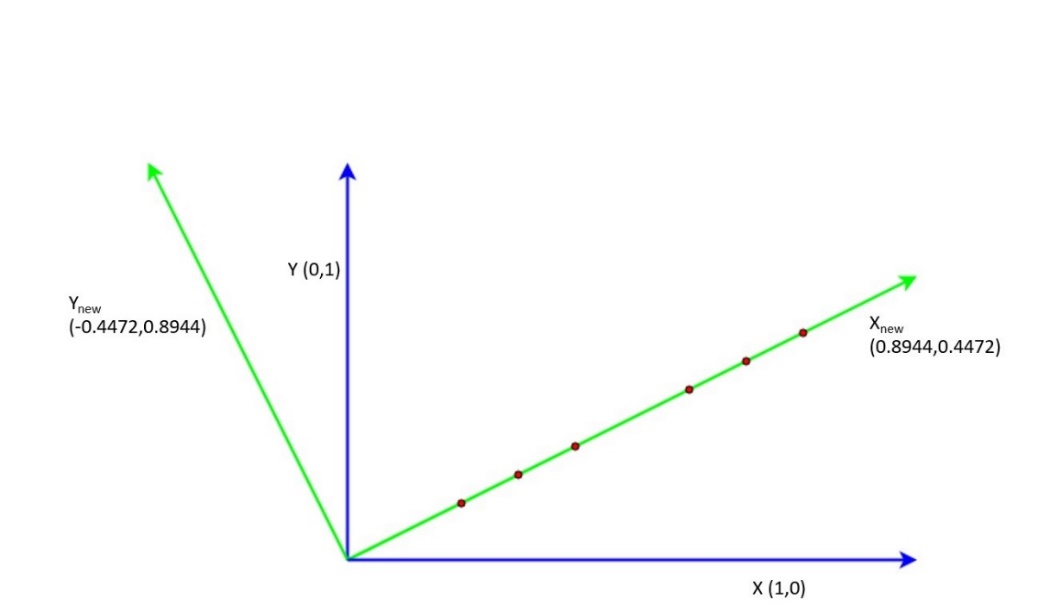
           Next, we need to find the eigenvectors of the covariance matrix C which would be **Principal Components** of the original dataset A. They come out to be the following:

           (You can use the np.linalg.eig function to compute them)

 eigvec1=[0.8944,0.4472] and eigvec2=[−0.4472,0.8944]

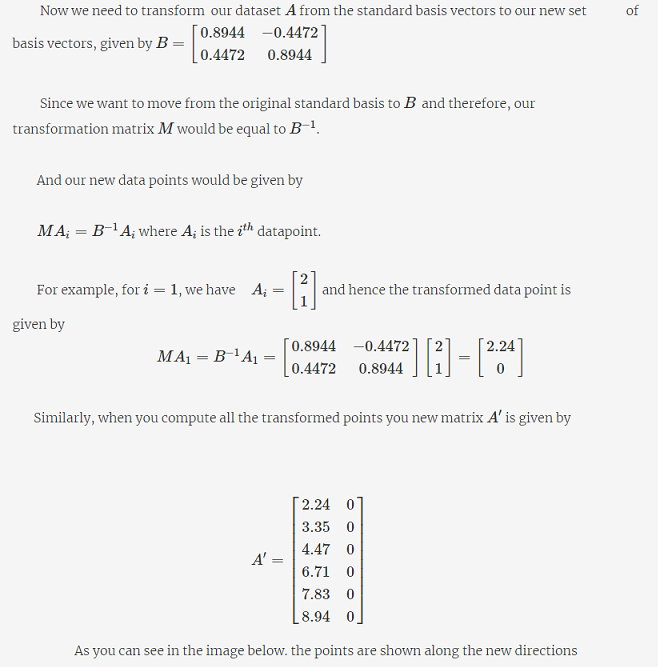
           These eigenvectors form the new set of basis vectors.

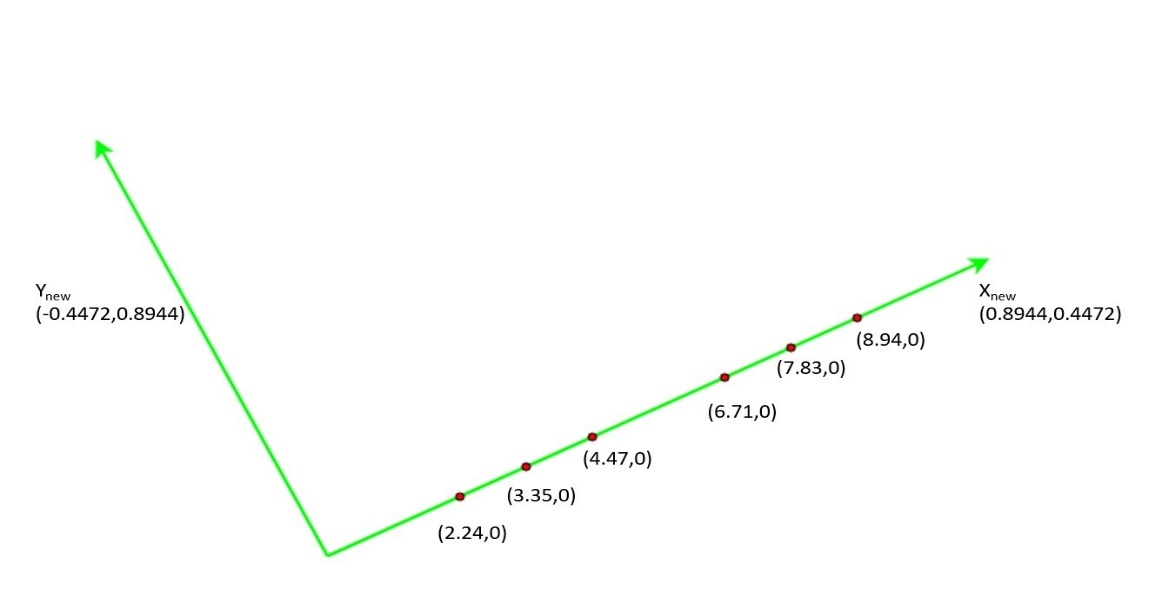
           The eigenvectors represent the new set of basis vectors or directions along which points need to be represented now. The new directions can be visualised below along the green lines:



* Step 3: Transforming the original data to the new set of vectors

         Now we need to transform  our dataset A from the standard basis vectors to our new set of basis vectors, given by B=[[0.8944,−0.4472],[0.4472,0.8944]]





Now it is easy for you to decide that Ynew is completely redundant and only Xnew is the only direction that you require to represent your points. This is exactly what your friend suggested as well.

 In conclusion, you saw that, by using PCA you're able to convert your original data to a new basis and then take a decision on which directions/features to keep and therefore perform dimensionality reduction.

 You can take a look at the following code to understand how the above calculations were executed in Python.

**[Demonstration: Algorithm of PCA](https://cdn.upgrad.com/UpGrad/temp/7315711b-0acb-4eaf-bce8-2df1d2e21e04/Algorithm+of+PCA+Demonstration.ipynb" \o "Algorithm of PCA Demonstration.ipynb" \t "_blank)**

[file\_download](https://cdn.upgrad.com/UpGrad/temp/7315711b-0acb-4eaf-bce8-2df1d2e21e04/Algorithm+of+PCA+Demonstration.ipynb" \o "Algorithm of PCA Demonstration.ipynb" \t "_blank)**[Download](https://cdn.upgrad.com/UpGrad/temp/7315711b-0acb-4eaf-bce8-2df1d2e21e04/Algorithm+of+PCA+Demonstration.ipynb" \o "Algorithm of PCA Demonstration.ipynb" \t "_blank)**

# **Practical Considerations and Alternatives**

Those were some important points to remember while using PCA. To summarise:

* Most software packages use SVD to compute the principal components and assume that the data is **scaled and centred,**so it is important to do standardisation/normalisation.
* PCA is a**linear transformation method** and works well in tandem with linear models such as linear regression, logistic regression etc., though it can be used for computational efficiency with non-linear models as well.
* It should **not be used forcefully to reduce dimensionality**(when the features are not correlated).

You learnt some important shortcomings of PCA:

* PCA is limited to linearity, though we can use **non-linear techniques such as t-SNE**as well (you can read more about t-SNE in the optional reading material below).
* PCA needs the components to be perpendicular, though in some cases, that may not be the best solution. The alternative technique is to use **Independent Components Analysis.**
* PCA assumes that columns with low variance are not useful, which might not be true in prediction setups (especially classification problem with a high class imbalance).

If you are interested in reading about t-SNE (t-Distributed Stochastic Neighbor Embedding) or ICA, you can go through the additional reading provided below.

This brings us to the end of this segment.

**Additional Reading**

**t-SNE**

* [Laurens van der Maaten's (creator of t-SNE) website](https://lvdmaaten.github.io/tsne/)
* [Visualising data using t-SNE: Journal of Machine Learning Research](http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf)
* [How to use t-SNE effectively](https://distill.pub/2016/misread-tsne/)

**Independent Components Analysis**

* [Stanford notes on ICA](http://cs229.stanford.edu/notes/cs229-notes11.pdf)