Clustering

1 Clustering - It is the task of dividing the unlabeled data or data points into different cluster such that similar data point fall in the same cluster.

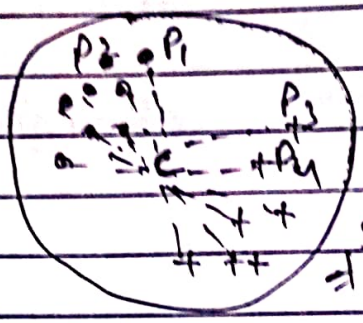
Application of Cluster:- Social network analysis, medical imaging, image Segmentation and anomaly detection.

WCSS → Within in Cluster Submission of Square

$= \sqrt{(x_1 - x_i)^2 + (x_2 - x_i)^2}$

WCSS is to find the no. of cluster in the set

(1)

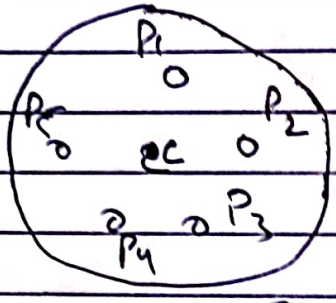


If $k = 1$

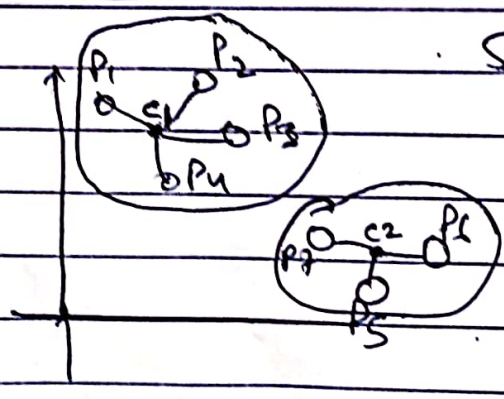
then

SS = Submission of Square

$SS = (C_1 - P_1)^2 + (C_1 - P_2)^2 + (C_1 - P_3)^2 + (C_1 - P_4)^2 + (C_1 - P_5)^2$



(2)



SS_1 for cluster 1 → $(C_1 - P_1)^2 + (C_1 - P_2)^2 + (C_1 - P_3)^2 + (C_1 - P_4)^2$

SS_2 for C_2 → $(C_2 - P_5)^2 + (C_3 - P_6)^2$

inertia = WCSS → $SS_1 + SS_2$

$$WCSS = \Sigma\Sigma (C_i - P_i)^2$$

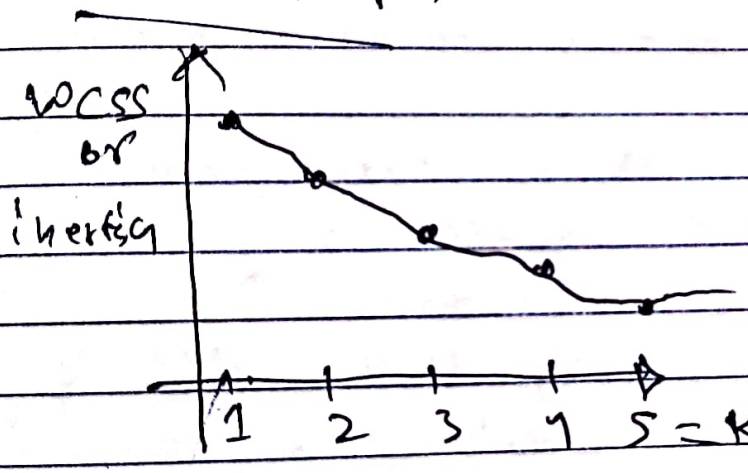
Σ = One Submission for individual cluster &

Σ = One Submission for itself.

Q How and when WCSS will be helpfull.

Ans If we have $K=3$ then we take $K=4$ then $K=n$. In case of $K=n$ then the WCSS will be 0. So when we have a 1 cluster the distance is going to be maximum. Then if $K=1$ then the distance is going to decrease than more for $K=2$

Elbow graph



Note :- for final model we find WCSS for every value of K and we will plot it with the help of Elbow graph and we will check from where there is a no diffrence of K in data and we will select K for a particular model.

- Here is a problem to allocate the Centroid in the data set in K Mean so to overcome the problem we have K-Mean ++

Note :- whenever we try to build a cluster we either follow
- Agglomerative or
- Divisive (K-mean)

Hierarchical clustering

1 Hierarchial clustering – It is a popular method for grouping objects. It create groups so that object within a are similar to each other

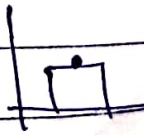
- It is separating data into groups based on some measure of similarity.

Q. What exactly Hierarchial do –

Ans) 1) Every data make an cluster and lie on X-axis of 2D graph.

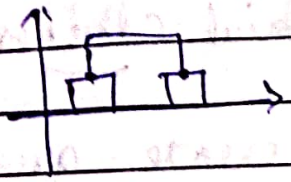
2) After that ~~point~~ data point find similar or near data point to make a dendogram

3) It make vertical line based on euclidian distances and join to data point with similarities

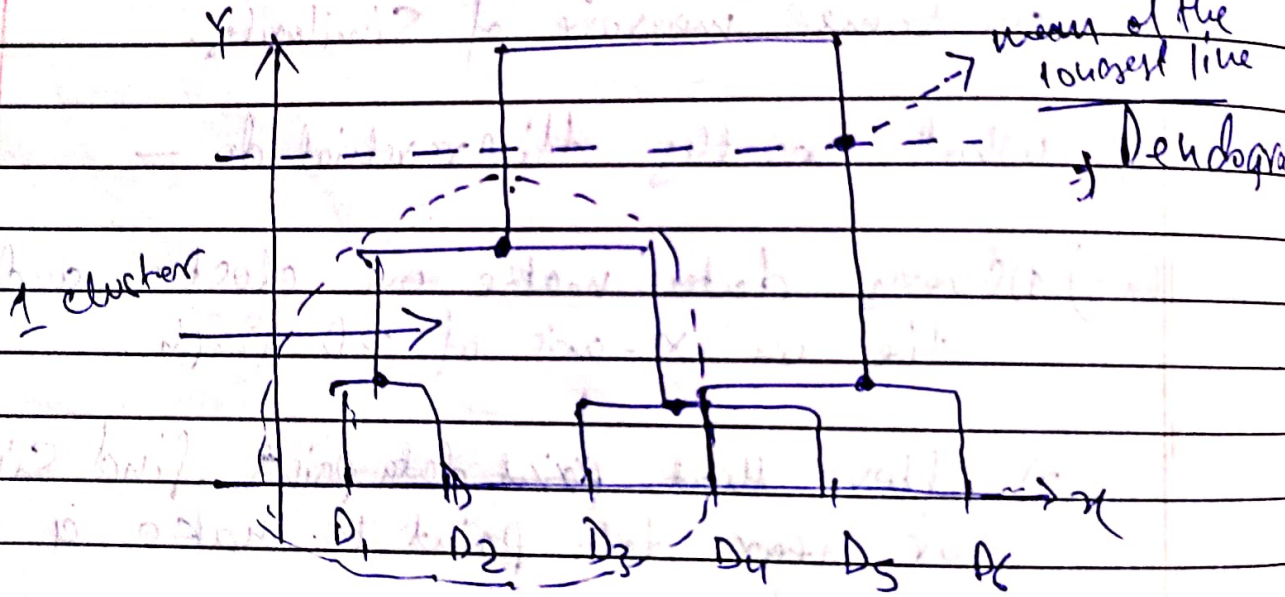


4) After connecting with each data point and make a graph there will be every centroid for every connecting data point

5) and based on the less distance or the similarities we will connect every centroid



6) we will do till every point attach or connect to each other



(7) Now we will check which vertical ↑ distance is more and after choosing the more vertical line we will find a mean of that long vertical line and we will draw a horizontal line

(8) And the no. of line of point it is connected it will created that much cluster for that dataset

(9) This will known as Dendogram approach or we can say Agglomerative clustering.

— D.B SCAN Clustering —

1 DBSCAN :— Density Based Spatial Clustering of Application with Noise.

2 DBSCAN is create cluster by —

i) Epsilon :- representing a distance / radius.

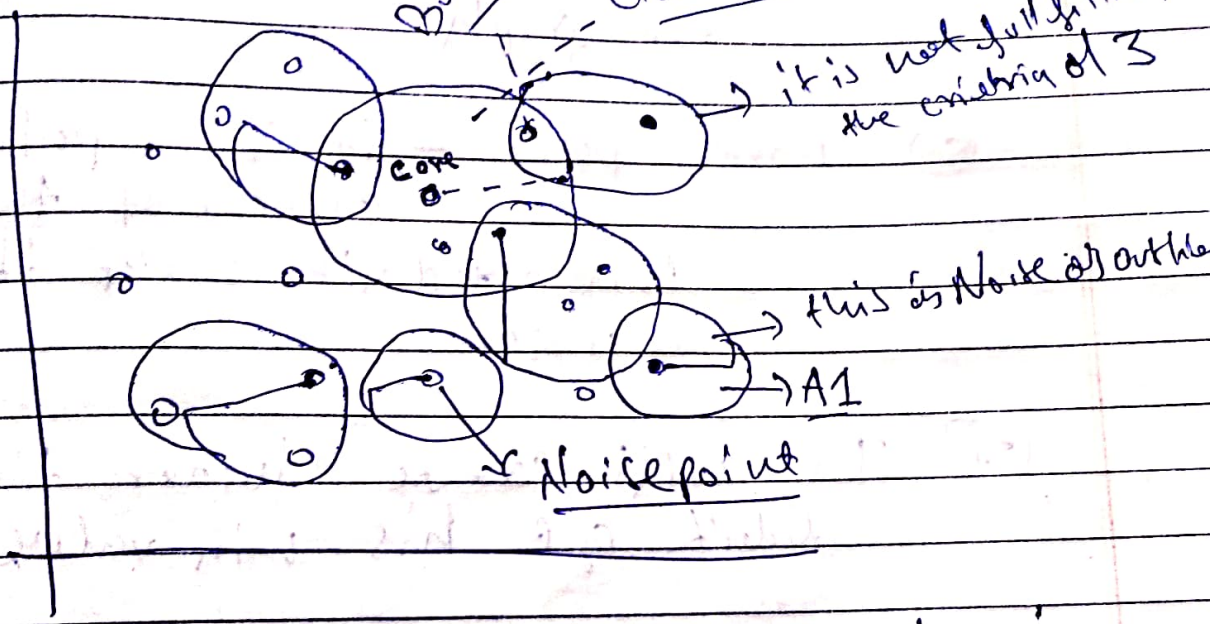
ii) Min-point :— Minimum no. of point requires to make a cluster.

iii) Core point :- Satisfy the minimum point.

iv) Border-point :— point is a part of some circle but not form its own cluster.

v) Noise :— outlier and which is not a part of core point as well as a cluster.

3. How to make cluster or group in DBSCAN if Epsilon is 1 & min points is 3



Ans 1) We will choose a random data in the graph and draw a circle whose radius will be 1 and minimum data point will be 3 or greater than 3 and the random point is taken is basically a core point

2) Now i will take all the point of the circle of the cluster 1 will is form.

3) Now we will do the same thing with the data just inside the circle or cluster 1. but it should have Epsilon 1 & min point 3.

(4) If any point doesn't a part of any core point or cluster then it will be Noise / outlier point for eg - A1 in the graph

(5) Border point - it is a point of cluster of the core point but didn't make it any cluster

(7) It will create cluster till it didn't get this min value

Metrics of Cluster —

1. How to evaluated the cluster — (Performance Matrix) or Performance Matrix of cluster.
   i) Jaccard coefficient
   ii) Entropy
   iii) Purity
   iv) Silhouette coefficient
   v) Seperation
   vi) Rand Index

2. Rand Index — It is one of the Metrics by which we will able to observe or monitor that how good our cluster creation is.

$$\text{Rand Index} = \frac{\text{Total Agree}}{\text{Total Agree} + \text{Total Disagree}}$$

SS → It means that both the point belong to a same cluster for both our algorithm and Ground Truth (ground truth means that base which i will able to compare)

DD → Both point don't belong to the same cluster but both our algorithm and Ground Truth.

SD →

Ground Truth	S 1	D 0	→ Prediction clustering
S 1	SS	SD	
D 0	DS	DD	

So, Rand Index will be = $\frac{SS + DD}{SS + DS + SD + DD}$

* Ground Truth → is our Actual observation or equivalent to Y it provided by us.

3 Jaccard Coefficient :- A total dataset which our cluster is able to group or seprate from a ground truth Some values we are able to get

$$JC = \frac{SC}{SS + SD + DS}$$

4 Entropy = $-P_i \cdot \log(P_i)$ is always try to tell that how random my distribution are.

$$= -\sum P_i \log(P_i)$$

Entropy tell us the randomness (scatterness) in the cluster and we always try to or looking for less randomness into our data distribution. So we can compare in our data which cluster has a less randomness.

5 Purity :- Total percentage of data set or a data point clustered directly.

Purity of cluster i = P_i = $\max(P_{ij})$

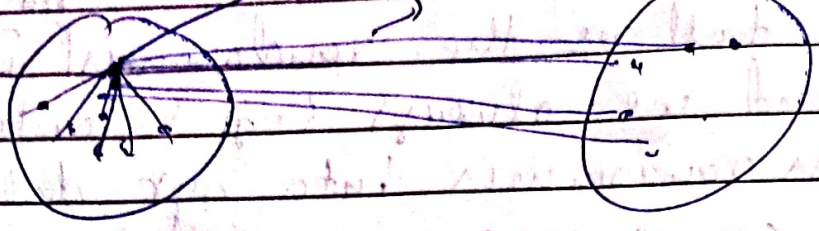
Purity of whole cluster = $P(C) = \left(\sum \frac{m_i}{n} P_i\right)$

V.V.I 6 Silhouette — to find accuracy of the clusters

$$s(x) = \frac{b(x) - a(x)}{\max\{a(x), b(x)\}}$$

$a(x)$ = average distance of x from all the other points in the same cluster.

$b(x)$ = average distance of x from all the point in another cluster.



Note - In other metric we have to prepare a ground Truth. then only we can able to calculate accuracy or stability of model in-term of clustering but in case of silhouette it not required because it following distance based approaches to evaluated whether my cluster is ~~god~~ good or bad.

$$\text{Silhouette} = \frac{1}{N} \sum s(x)$$

PCA

— Principal Component Analysis —

→ PCA is not a approach where we can train a model, It is a data pre-processing Stage.

→ PCA help to reduce a dimension.

1 Step of PCA —

i) PCA Say that always take dataset who has mean = 0 & Standard Deviation = 1. (Standardization, Z-Score Normalization).

→ PCA Never work in a Normal Distributed dataset, it work with Standard Normal Distri.

ii) Draw a best fit line or a Straight line as we do in Linear Regression.

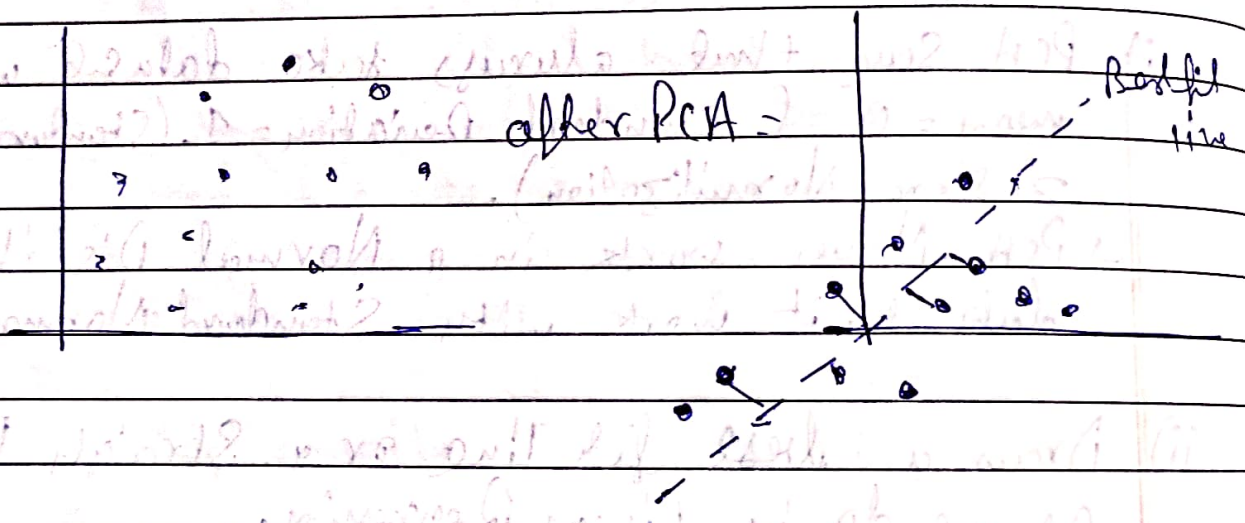
iii) The line draw is PC1 which is a Straight line and after that PC2 will be create which is perpendicular line of PC1.

and

If PC3 created that it will be perpendicular line of PC1 & PC2 and it is Z-axis.

Note - In PCA we don't eliminate the columns we reduces the columns.

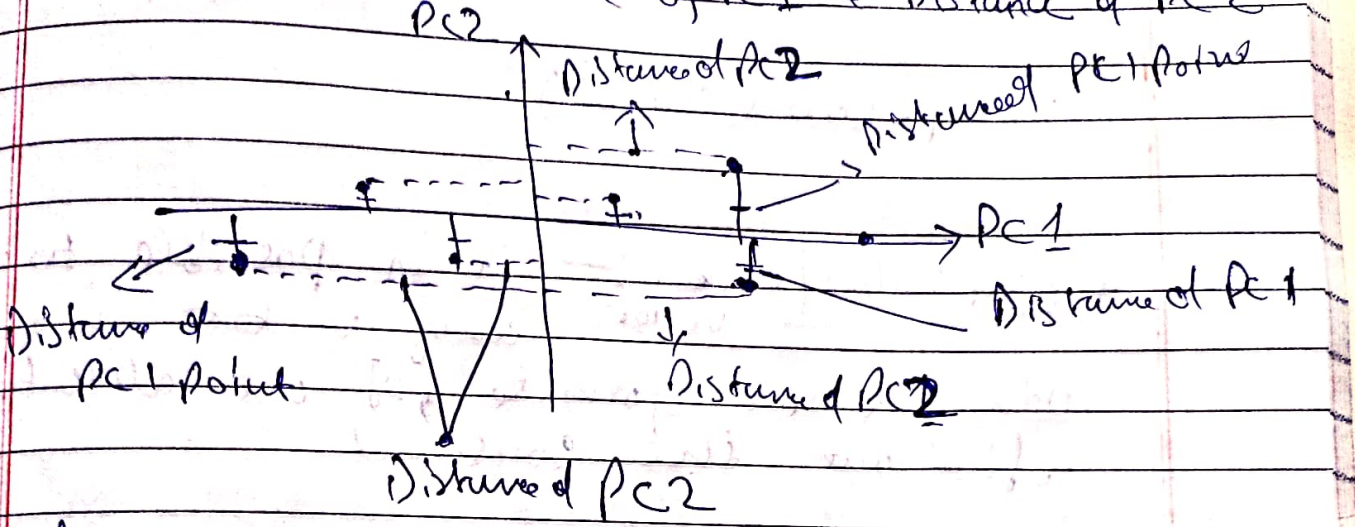
Note :- PC 1 (Principal component 1) is the most contributing component that other PC2 & PC3 ... PCn will have less contribution. Because PC 1 is a kind of a straight line which has been drawn base out of the relation. And rest PC line are the perpendicular on top of the first one.



2 How to know that how many PC line to created

Ans EVR (Explained Variance Ratio) :- One PC have a capacity that one single line or how much relation this one single line is to explain is called (EVR) value

EVR of PC1 = Distance of PC1 Point / Distance of PC1 + Distance of PC2

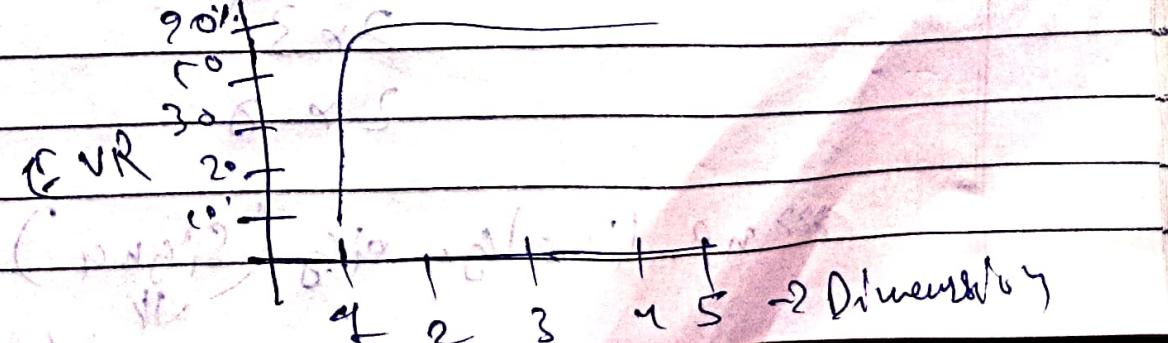


for ex,

$$EVR \text{ of } PC1 = \frac{50}{50+5} = 0.91$$

∴ It means that PC1 line contain 91% of data information.

3 Draw Screen plot to Know to Know how many dimension to take with the help of EVR



$$\text{Covariance}_{(C_1,C_2)} = \frac{\sum_{m-1}^{m-1} (C_1 - \bar{C}_1)(C_2 - \bar{C}_2)}{m-1}$$

>>> np.cov(a0, a1)

* eigen value/eigen vector :- it is possible to represent any vector in any other vector formate, if we able to find out the constant.

$$T(y = dv)$$

for ex

x_1	x_2
2	4
4	8
3	6
8	16

x_1

2 x 2

2 x 3

2 x 8

>>> np.linalg.eig(Sigma) or

Sigma = variable of cov.

>>> np.linalg.eig(np.cov(a0, a1)) - will give PC

5 ⇒ Eigen value always tell the one component & the other PC.

⇒ The higest eigen value will be considered the PC1.

Code of PCA -

```
>>> from sklearn.decomposition import PCA
>>> pca = PCA()
>>> principal_component = pca.fit_transform(data)
>>> pca.explained_variance_ratio_
>>> plt.figure()
>>> plt.plot(np.cumsum(pca.explained_variance_ratio_))
```

Note :- Here data should be standardization.

PCA - Code

i) Use Standard Scaling for mean = 0 & S.D = 1
ii) Find the Covariance Matrics.
iii) find the eginen eginen vector & Eginen Value.
iv) Draw Screen plot and check n_compent or no of PC.
v)

>>> i) Using glass data &
ii) droping the label colums

code1 >>> :. first do Standard Scaling to the data
>>> from sklearn.preprocessing import StandardScaler
>>> scaler = StandardScaler()
>>> scaler.fit_transform(data)
>>> scale_data = scaler.fit_transform(data)

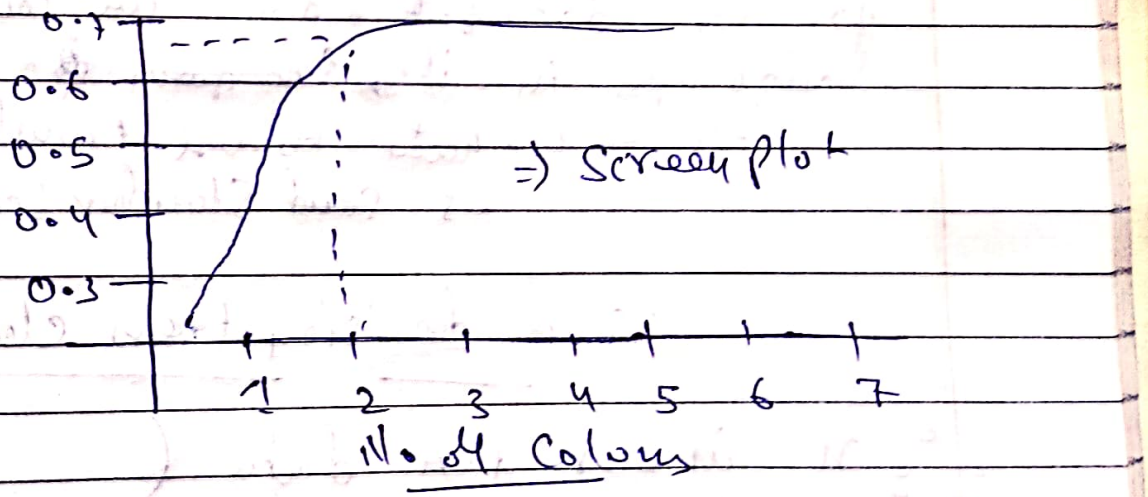
code2 :. to change in data frame
>>> df = pd.DataFrame(scale_data, columns = data.columns)

code3 >>> df.describe() (# to know the whether mean=0 & S.D = 1 or not)

Code 4 >>> from sklearn.decomposition import PCA
>>> pca = PCA()
>>> pca.fit_transform(df)

∴ Now we have to the EVR or the no. of PC to take in the PCA with the help of Screen plot with the help EVR

Code 5 >>> plt.figure()
>>> plt.plot(np.cumsum(pca.explained_variance_ratio_))
>>> plt.xlabel("No. of Columns")
>>> plt.ylabel("EVR")



⇒ Screen plot

∴ this tell the No. of Principal component (PC) with the presenting of data lie.

No. 2 lies - 70% of data so we will take n-component = 2

Code6 >>> pca1 = PCA(n_components = 2)

Code7 >>> new_data = pca1.fit_transform(df)

Code8 >>> # to create new data into Data Frame

>>> x = pd.Dataframe(new_data, columns=["PC1", "PC2"])

//

Now we have use PCA to reduce our Dimension or columns for better performance for our Data Set. Now we can use any algorithm in it.

Note — we have to use transform as we have used Standard Scaler

Using decision tree classifier

x is our PCA data

y = data.Class our label or outcome

Code9 >>> from sklearn.tree import DecisionTreeClassifier

>>> dt_model = DecisionTreeClassifier()

>>> dt_model.fit(x, y)

Note :- Now we can use for predict but the problem is that after PCA our columns has change to 2 and in the original data set there 9 colums. So, unknown person will give 9 colums. And we also uses standard scaler.

So, we have to used pca1.transform to reduce the colums and scalar.transform to change the value for the prediction purpose. Shown in the given code in 10

code 10 :- db_model.predict(pca1.transform(scalar.transform([[0.12, 0.34, 0.44 ..... 9]]))

— Clustering Code —

>>> Using Mall dataset

KMeans —

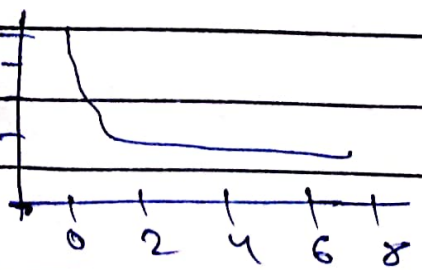
Code1 >>> from sklearn.cluster import KMeans

Code2 >>> wcss = [ ]
>>> for i in range (1,15):
>>> [Kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 30)
>>> Kmeans.fit(x)
>>> wcss.append(Kmeans.inertia_)

∴ Now we will do elbow graph with the help of WCSS so we could know that the no. of cluster should we take.

WCSS = interia = help to know or find the no. of cluster that we should use

Code 3 >>> plt.plot(range(1,15), wcss)



code 4 >>> Kmean1 = KMeans(n_cluster = 5, init = "k-means++", random_state = 30)
>>> Kmean1.fit_predict(x)

code 5 >>> we will a column of predict in our dataset
>>> x["cluster number"] = Kmean1.fit_predict(x)

code 6 >>> Kmean1.predict([[25,3277]])

DBSCAN

>>> from sklearn.cluster import DBSCAN
>>> dbscan = DBSCAN(eps=1, min_sample=3)
>>> dbscan.fit(x)