Dt: 3/05/22
3:30 Am

* Un Supervised Machine Learning :

We don't have any output columns, variable /, Feature

↓ (We Group them)

Clustering :-

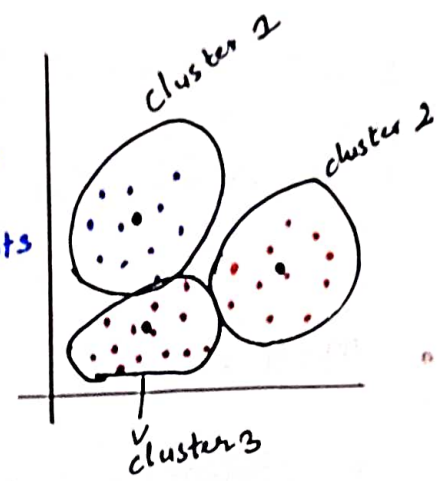
Grouping of Similar objects

1. K-means Clustering
2. Hierachical clustering
3. DB Scan clustering

Un label data

1. What is a cluster ?

A:- A group of objects that are "similar to other objects" (records.) in the cluster, and dissimilar to data points in other clusters.



groups :- segments

2. Clustering applications ?

- Retail / marketing :
  - * Identifying buying patterns of Customers
  - * Recommending new books (or) movies to new Customers.

- Banking :
  - * Fraud detection in credit card use.
  - * Identifying clusters of customers (eg. loyal)

- Insurance :
  - * Fraud detection in claims analysis
  - * Isurance risk of customers.

- Publication :
  - * auto - categorizing news based on their content
  - * Recommending similar news articles

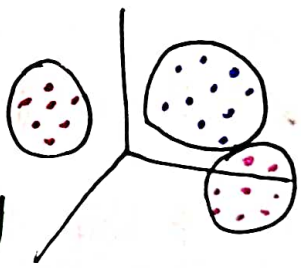
- Medicine :
  - * characterizing patient behaviour

- Biology :
  - * clustering genetic markers to identify family ties.

3. Clustering algorithms

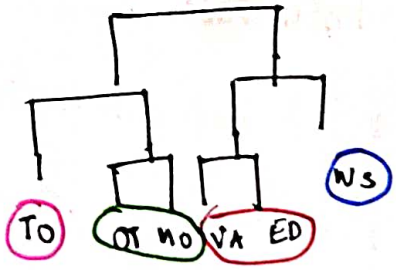
* Partitioned - based clustering
  - Relatively Efficient

Eg:- K-means, K-median, Fuzzy C-means.



* Hierarchical clustering
  - Produces trees of clusters

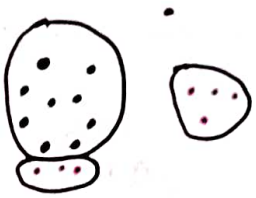
Eg:- Agglomerative, Divisive



* Density - based clustering

- Produces arbitary shaped clusters

eg:- DBSCAN



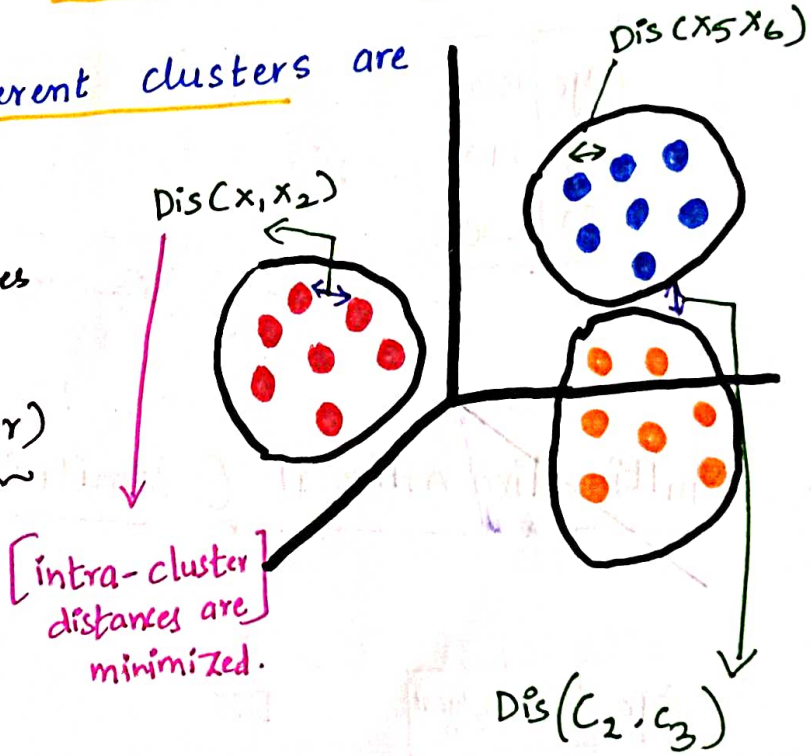
1. K-Means.

* Partitoning clustering
* K-means divides the data into non-Overlapping subsets (clusters) without any cluster internal structure
* records within a cluster are very similar
* records across different clusters are very different

Ex :- covid zones

*Determine the similarity (or) dissimilarity

- intra - within
- inter - outside



⇒Aim:-

* Between two clusters, the distance should be maximum - Better the model
* within cluster, the data point should be closely to each other - Better the model

* One-dimensionality (similarity / distance)

	Age
1	54
2	50

We, calculate Distance.

$$\sqrt{(54-50)^2} = 4.$$

$$Dis(X, X_2) = \sqrt{\sum_{i=0}^{n}(x_{1i}-x_{2i})^2}$$ distance formula.

#3 dimensional - 3 columns

$$\sqrt{(x_2-x_1)^2+(y_2-y_1)^2+z_2-z_1)^2}$$ (Euclidean)

* two-dimensionality (similarity / distance)

Age	Income
54	190
50	200

$$Dis(X, X_2 = \sqrt{(x_1-x_2)^2}$$

$$= \sqrt{(54-50)^2+(190-200)^2}$$

$$= 10.77$$

* Multi-dimentional (similarity / distance)

Age	income	Education.
54	190	3
50	200	8

$$Dis(X, X_2) = \sqrt{(x_{1i}-x_{2i})^2}$$

$$= \sqrt{(54-50)^2+(190-200)^2+(3-8)^2}$$

$$= 11.87$$

Q:- what is the difference b/w "K means" & KNN ?

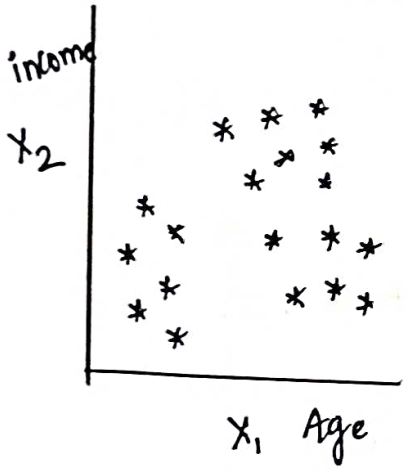
A: Both use Distance Formulas to group them.

but, K-means is clustering, KNN- classification.

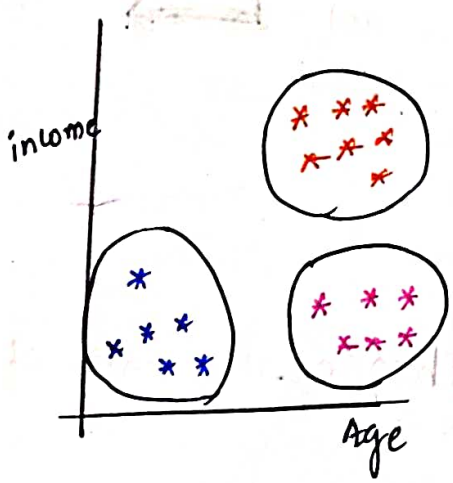
Non supervised (only group)

supervised (o/p) predict

* K-means



(normal Scatterplot)



(# K-means) Clustering.

* Steps to K-means :

STep : 1 (Choose The no. of "K" of clusters)

↓

STep : 2 (Select at random "K" points, The Centroids) (not necessarily from Your dataset)

↓

STep : 3 (assign Each data point to closest Centroid → (that forms "K" clusters)

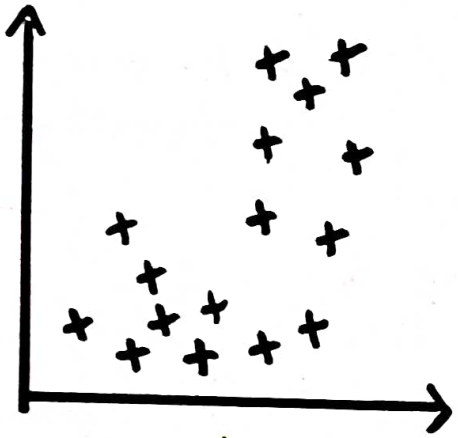
STep: 4 (Compute & place new Centroid of Each cluster)

STep: 5 (Re assign Each data point to the new closet Centroid. if any reassignment took place. go to Step 4. Otherwise go to FIN

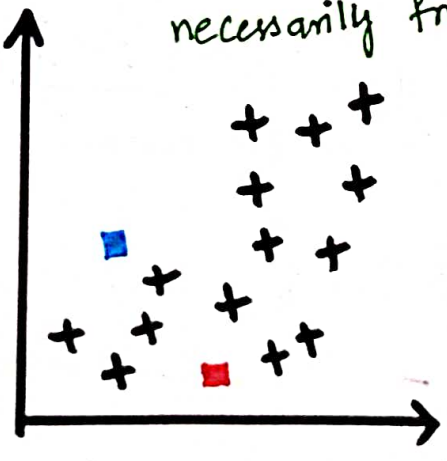
* K-value is not Fixed...

→ Your Model is Ready

⇒ STep :1 :- Choose the no. of K-clusters : 2

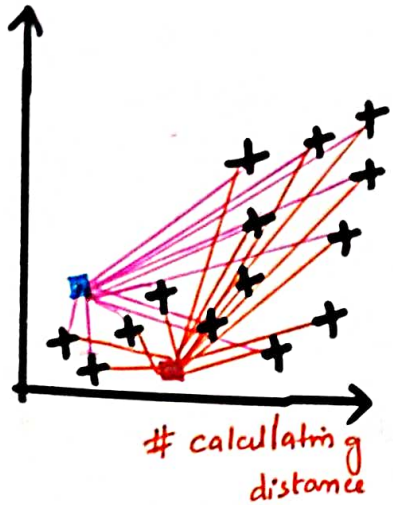


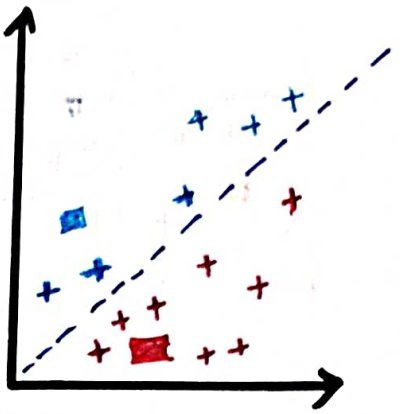
⇒ Step: 2 Select at random K-points, The Centroids (not necessarily from your dataset)



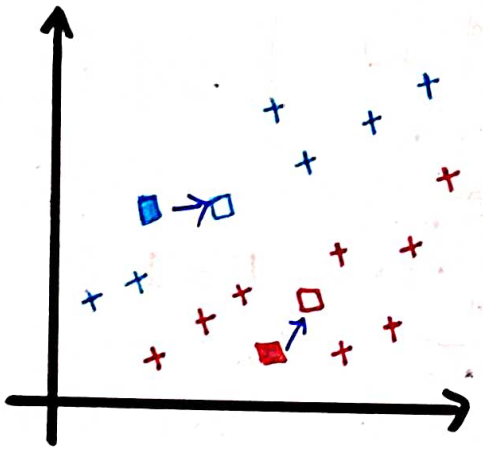
Centroid. Based on #assumptions

STep:-3 (assign Each data point to the closest Centroid that forms K-clusters.)





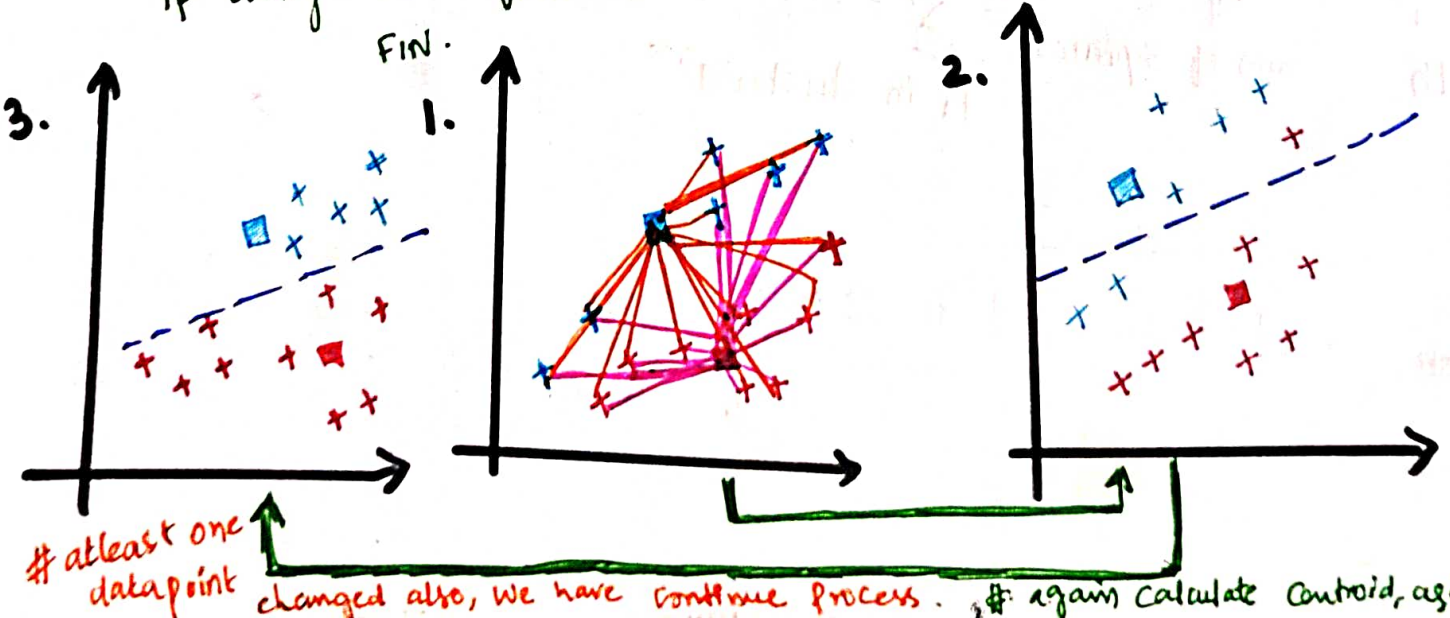
STep:4 Compute and replace the new centroid of Each other.



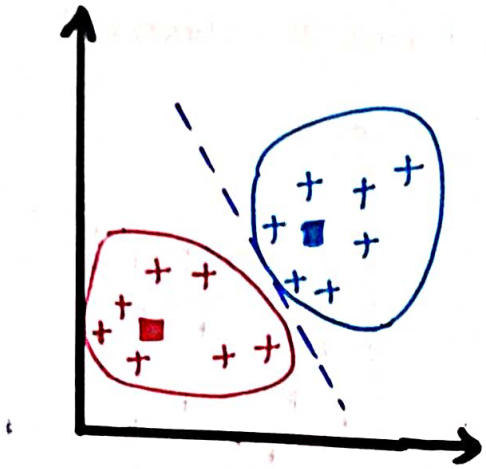
Calculate the original Centroid
Avg = $\bar{x}$ $\bar{y}$
# replace with avg.

STep:5 Reassign each data point to the new closest Centroid.
if any reassignment took place, go to step 4. Orther wise got

FIN.

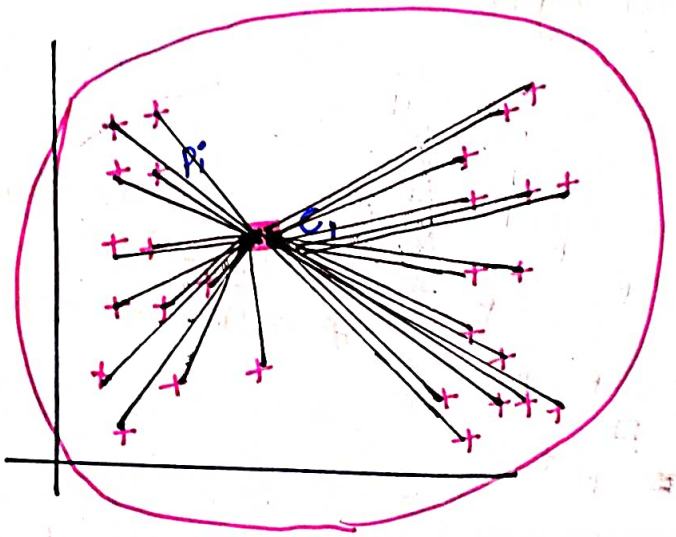


FINAL : Model is Ready.



Que :- How to choose right number of clusters ?

A :-



cluster 1

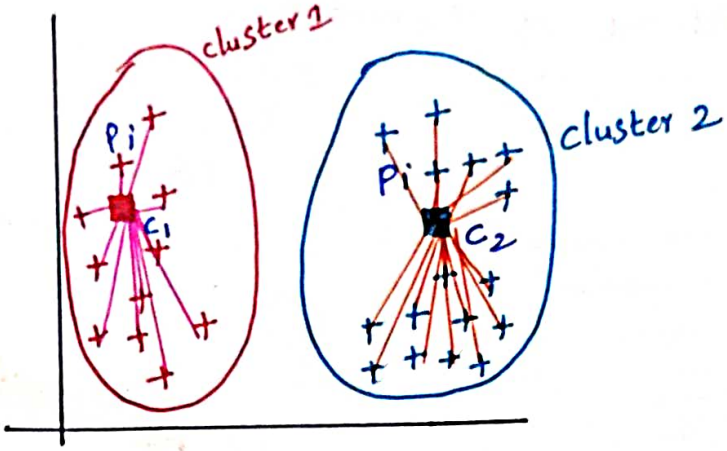
$K = 1$

* Wcss (cluster) ⇒ (with cluster Sum of Squares)

with — Sum of squares $= \sum_{P_i \text{ in cluster } 1} \text{distance}(P_i, C_1)^2$

$$* \text{Wcss} = \text{distance}(P_i, C_1)^2$$

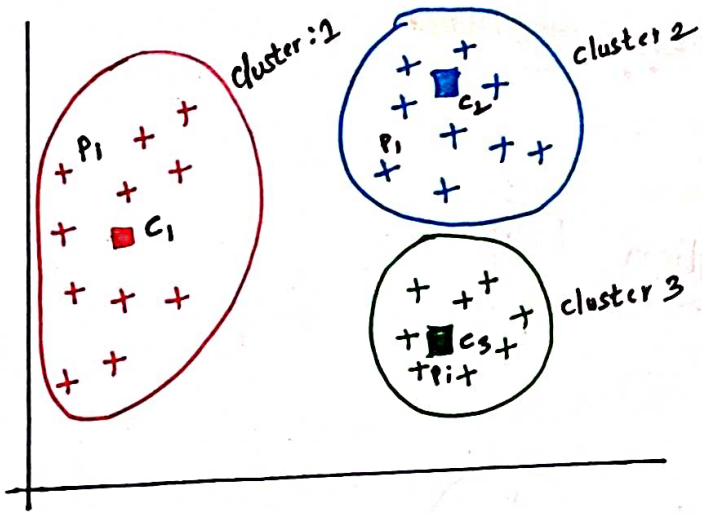
$K = 1$



#Calculate SSE

K = 2

* Wcss = $\sum_{P \text{ in cluster } 1}$ distance $(P_i, C_i)^2$ + $\sum_{P_i \text{ in cluster } 2}$ distance $(P_i, C_2)^2$ (SSE) + ... (SSE)



K = 3

Apply loop.

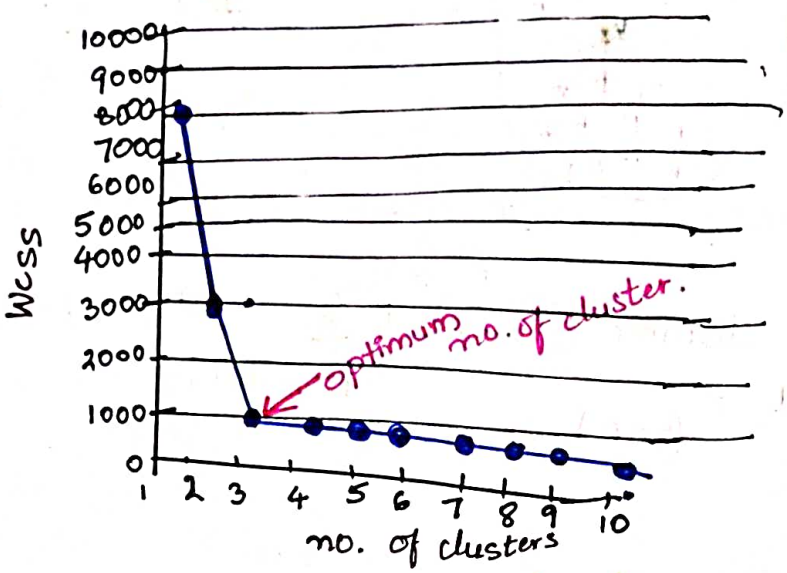
Wcss = $\sum_{P \text{ in cluster } 1}$ distance$(P_i, C_1)^2$ + $\sum_{P_i \text{ in cluster}}$ distance $(P_i, C_2)^2$ + $\sum_{P \text{ in cluster } 3}$ distance $(P, C_3)^2$

03/05/22

6:00 Am.

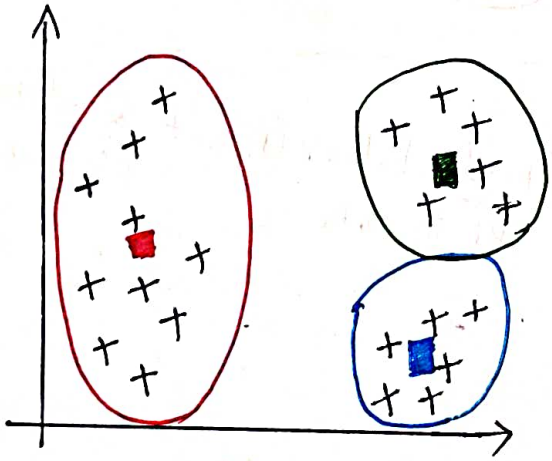
* choosing The right no. of clusters

* Elbow Technique



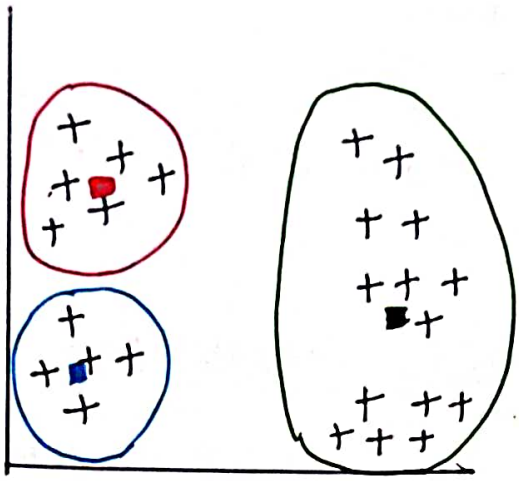
* Elbow method

* Random initialization trap



randomly

Que :- What would happen if we had a bad random initialisation ?



different types of Clusters But, Data set is common.

Based on random initalization. it is going to change clusters

If you choose wrong initalization. Total problem Gets wrong. it is the problem in K-means.

A:- Solution for this is K-means ++

⇒ The Random initialisation Trap

One major drawback of K-means clustering is Random initialisation of Centroids. The formation of clusters is closely bound by the initial position of a centroid. The random positioning of Centroids can Completely alter clusters and can result in random formation.

* The Solution is K-means ++
* K-means ++ is an algorithm that is used to initialise The K-means algorithm. *** instead of "arthematic mean" it Take "Weighted mean". For Calculating Centroid.

. K-means ++

1. Choose one Centroid uniformly at random from among the data points.
2. For each data point say x, Compute $D(x)$, which is distance between x and nearest Centroid that has already been choosen.
3. Choose one new data point at random as a new Centroid, using weighted probability distribution where a point "x" is chosen with probability proportional to $D(x)^2$.
4. Repeat steps 2 and 3 until K Centres have been choosen.
5. Proceed with Standard K-means clustering.

Ex:-

Age	Edu	Exp
21	B.Tech	0
24	B.Tech	2
22	B.Tech	0
28	MTech	3

group

→ grouping same Element (or) similar

Multiple feature over all

(calculate distance formula)

Encoding

CODE :-

```
# df = pd.read_csv("Mall_Customers.csv")
# df.head()
```

Out :-

CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
1	Male	19	15	39
2	Male	21	15	81
3	Female	20	16	6
4	Female	23	16	77
5	Female	21	17	40

```
# df.shape
```

Out: (200, 5)

```
# df.info()
```

Out:

200 entries, 0 to 199

Column	Non null Count	Dtype
CustomerID	200 non null	int64
Genre	200 non null	object
Age	"	int64
Annual Income	"	"
Spending Score	"	"

```
# df.isnull().sum()
```

Out 0.

X # to Understanding, we take only two columns.

```
# x = df.iloc[:,[3,4]].values
```

MODEL

```
from sklearn.cluster import KMeans.

# using the elbow method to find the optimal no. of clusters.

# wcss = []

for k in range (1, 21):

    kmeans = KMeans (n_clusters = k, init = "k-mean++")

    k means.fit (x)

    wcss.append (kmeans.inertia_)
```

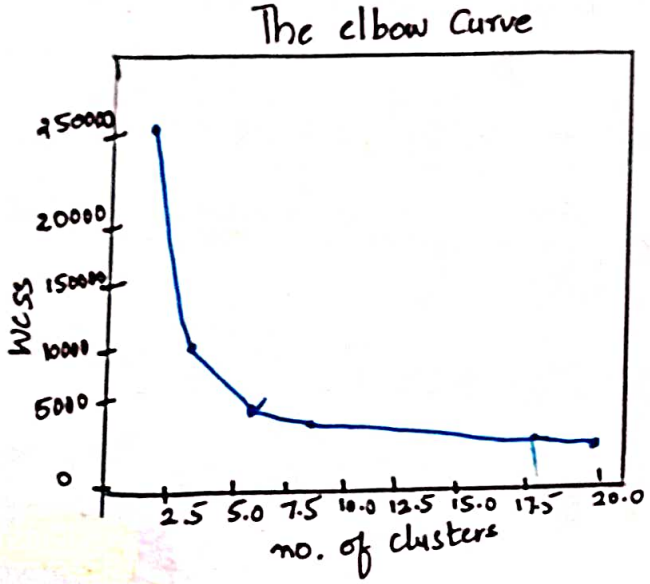
- #wcss values is called inertia

```
# WCSS

out: [269981.2800 ... # k=1
      1811363.5959 , # k=2
      106348.373062,
      73679.789039
       .
       .
       .
       .
       .
       .
      8504.241372467]
```

Elbow curve

```
# plt.plot (range(1,21), wcss)
# plt.title ("The elbow method")
# plt.xlabel ("no. of clusters")
# plt.ylabel ("wcss")
# plt.show()
```



fit the K-means Model on the data.

```
# Kmeans = KMeans (n_clusters = 5, init = "K-means++",
                   random_state = 42)
```

predict

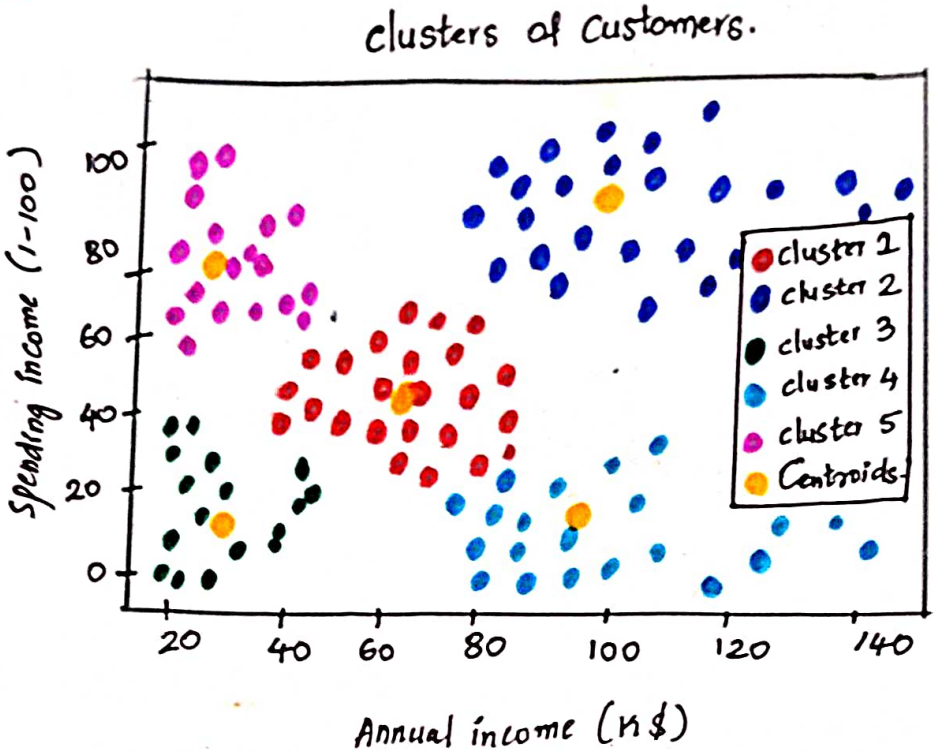
```
# y_Kmeans = Kmeans.fit_predict (x)
```

Visualising The clusters

```
# cluster 1
# plt.scatter(x[y_kmeans == 0,0], x[y_kmeans == 0,1],
                s=100, c="red", label="cluster 1")
# cluster 2
# plt.scatter(x[y_kmeans == 1,0], x[y_kmeans == 1,1],
                s=100, c="blue", label="cluster 2")
# cluster 3
# plt.scatter(x[y_kmeans == 2,0], x[y_kmeans == 2,1],
                s=100, c="green", label="cluster 3")
# cluster 4
# plt.scatter(x[y_kmeans == 3,0], x[y_kmeans == 3,1],
                s=100, c="cyan", label="cluster 4")
# cluster 5
# plt.scatter(x[y_kmeans == 4,0], x[y_kmeans == 4,1],
                s=100, c="magenta", label="cluster 5")
# Centroid
# plt.scatter(kmeans.cluster_centers_[:,0],
                kmeans.cluster_centers_[:,1],
                s=100, c="yellow", label="centroids")

# plt.title("clusters of customers")
# plt.xlabel("Annual Income ('k $)")
# plt.ylabel("spending score (1-100)")
# plt.legend()
# plt.show()
```

Out :-



#orginal data

```
# a = df.iloc[:,3]
# b = df.iloc[:,4]
# plt.scatter(a,b)
```

Out :