Expt. No. _____ Date _____ Page No. _____

—: Customer Segmentation :— (partition)

Types of ML :

i) Supervised ML & ii) Unsupervised ML

(i) Supervised :

↳ we train the m/cs using labelled data sets (some of the i/p already mapped to o/p.) & based on training, the machine predicts the output

machine train करते हैं i/p & corresponind o/p से (training data.)
then, ये predict करते हैं test cases के लिए।

Ex. provide data sets of dog & cat
↳ shape, size, color, height etc.
then i/p a picture & ask m/c to identify & predict o/p.

↳ Categories of supervised :—

(i) Classification : o/p variable categorial "yes" or "no"

(ii) Regression : linear relationship b/w i/p & o/p variables.

Teacher's Signature : _____

Unsupervised ML :

↳ group or categories the ~~user~~ unsorted dataset according to the similarities, pattern, and differences.

↳ models are trained with data that is neither labelled and acts on with out supervision.

ex. basket of fruits & images are unknown to model.
task m/c to find patterns & categories
↳ shape, colour, size, height etc.

Categories of Unsupervised ML :

i) Clustering : → स्थायी अंश

↳ find inherent groups from the data.

↳ it is way to group objects in to clusters.
based on similarities, fewer similarities, no-similarities.

↳ algo ~~o~~ used :

~~using~~ ↳ K-Means clustering algo

↳ Mean shift algo

↳ principal component analysis.

etc.

ii) Association :

↳ find interesting relations among variables with in a large data set.

↳ find dependency of one data item on another data item.

Date
Expt. No. Page No.

→ Advantages:

i) used for complicated tasks compared to supervised algo work on unlabelled datasets.

ii) Unlabled data getting easier compare to labelled data.

→ disadv:

i) can be less accurate

ii) algo not trained with the exact op in prior.

K-Means clustering algo:

↳ groups the unlabeled data into different clusters.

↳ K → no. of pre-defined clusters.

↳ unlabled data divided into K- different clusters in such a way that each dataset belongs only one group. that has similer properties.

↳ it allow us to cluster data into different groups and a discovers the categories of group.

↳ Centroid-based algo, each cluster associated with a centroid.

aim is minimize the sum of distance b/w data point & their corresponding clusters.

↳ K should be predetermined in this algo.

Teacher's Signature :

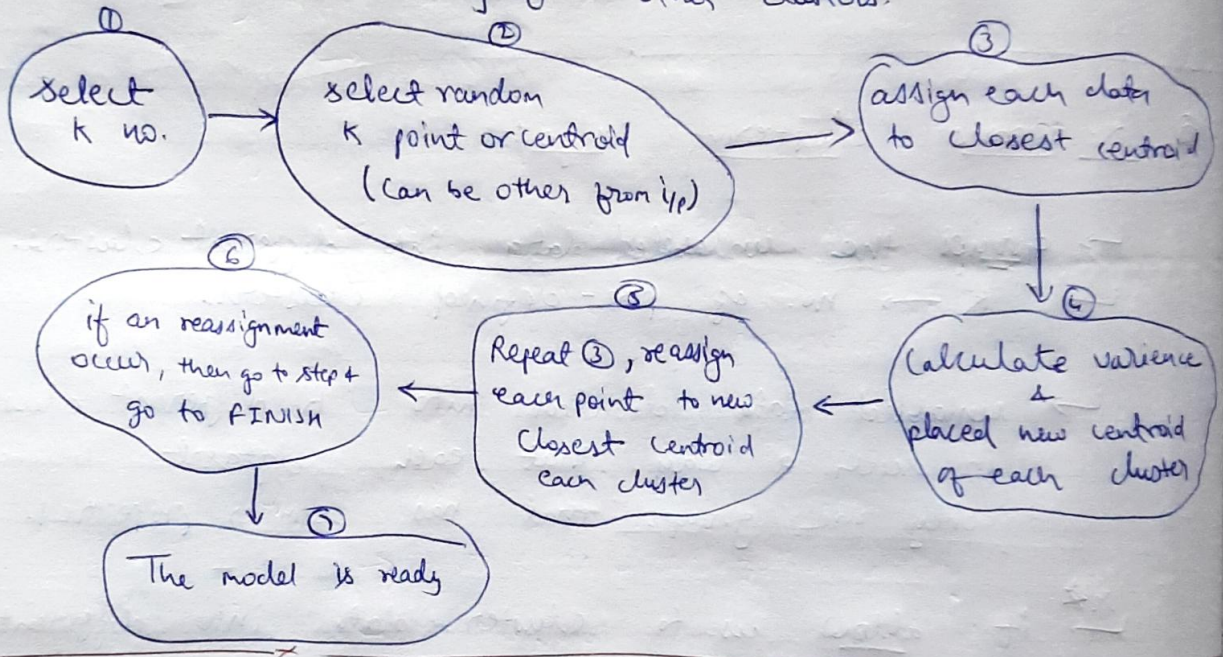
↳ take unlabelled dataset & divided into K- clusters, and repeats process until don't find best cluster.

↳ Two tasks perform by K-Means algo:

(i) Determines the best value for K center points or centroids by an iterative process.

(ii) assign each data point to its closest K-center. which are near to ~~perticular~~ particular K-center, create a cluster.

Hence, each cluster has data points with some commonalities & it is away from other clusters.



0→ choose K (no. of clusters):

Elbow method:

↳ concept WCSS (within cluster sum of square

$$WCSS = \sum_{P_i \text{ in } Cl_1} dis(P_i C_1)^2 + \sum_{P_i \text{ in } Cl_2} dis(P_i C_2)^2 + \sum_{P_i \text{ in } Cl_3} dis(P_i C_3)^2$$

↳ sum of squares of the dis btw each point & its centroid with in cluster 3.

same for other 2.

Date

Expt. No. Page No.

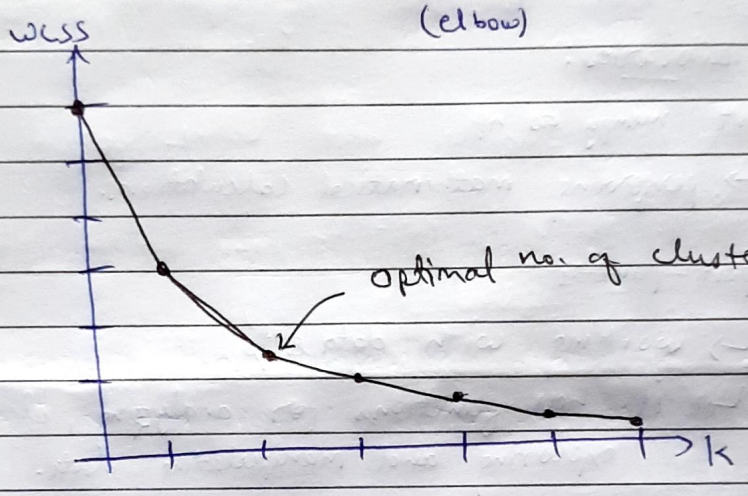
To find optimal value of k.

i) execute k-means clustering on given dataset for different k values (range 1-10).

ii) for each k, calculate WCSS.

iii) plot curve WCSS values & no. of clusters k.

iv) sharp point of bend or a point of plot looks like arm. (elbow) Consider that best value for k.



Note: We can choose no. of cluster equal to data points then WCSS becomes zero, that end point of plot.

Teacher's Signature :

—: Python implementation of K-Means :—

Steps to be:

i) Data pre-processing
ii) finding optimal no. of clusters using elbow method
iii) Training K-means algo on training dataset
iv) visualizing the clusters.

① Data pre-processing:—

(i) Importing Libraries:

a) import numpy as np
↳ performing mathematical calculations.

b) import pandas
↳ working with data sets
↳ it has functions for analyzing, cleaning, exploring and manipulating data.
↳ make readable data.

c) matplotlib.pyplot
↳ serve as visualization utility
↳ graphs, bubble chart, pie chart etc.

(ii) Importing data:

we need to deal with huge datasets while analyzing the data, which usually get in csv file.

using: read_csv( )
↳ location of file.

Expt. No. Date
Page No.

a) information of dataset
↳ data.info()
↳ Print information about Data frame
Contains:
no. of columns, no. of rows, Dtype and memory uses.

↳ head() & describe
↳ discription of first 5 & last 5 rows of data frame.

↳ isnull() & sum()
↳ checking missing values in data frame if any present then give no. of vacant or missing values.

(iii) Extracting independent variables:
↳ we want group based upon spending score & spending score data is important for mall owner.
↳ we neglect other arguments.

use: iloc[: , [3,4]].values
(row: empty because we want all rows; columns: starting col. 3, last column 4)

Teacher's Signature :

(2) Finding the Optimal number of clusters using elbow method:

↳ use the KMeans class of sklearn.cluster library to form clusters

↳ we store WCSS value in a list for 1-10 clusters.
then we draw the plot by plotting WCSS & clusters. ⊖

↳ sharp bend consider best value for K (no. of clusters.

```
KMeans(n_clusters=i, init='K-Means++', random_state=42)
```
↳ no. of clusters (default is 8)

(Y)
```
kmeans.fit(X)
list.append(kmeans.inertia)
```

init: { K-means++ }

↳ select initial cluster centroids using sampling based on an empirical probability distribution of the points contribution to overall inertia.

↳ making several trials at each sampling step and choosing best centroid.

random state:

↳ Determines random no. generation for centroid initialization. use an int to make the randomness.

inertia_ :

↳ Sum of squared distances of samples to their closest cluster center.

fit():

↳ Compute K-means clustering.

③ Training K-means algo on data:

```
KMeans(n_clusters=5, init='k-means++', random_state=42)
y_predict = Kmeans.fit_predict(data)
```

fit_predict: Compute cluster centers & predict cluster index for each sample.

④ Visualizing the clusters:

↳ To visualize the clusters will use scatter plot using scatter() function of matplotlib

```
Scatter(x_axis, y_axis, marker="d", c=" ", label=" ")
```

marker: design for points; c: colour; label: labelin particular cluster.

x_axis:

↳ we have different values for different clusters ~~as well as~~

```
X[y_predict==0, 0]
```

(thats for 1st cluster) → first column from data set

y_axis:

```
X[y_predict==0, 1]
```

for 1st cluster. → 2nd column from data set

↳ scatter(kMeans.cluster_centers_[:,0], kMeans.cluster_centers_[:,1], s="", c="", label="centroid")
(co-ordinates of cluster center; 0 → col.)

↳ legend():
↳ place legend on the axes.

⑤ Result:
↳ different clusters with different colours.
↳ five different clusters
i) rich, spending low
ii) rich, spending high
iii) medium, spending medium
iv) poor, spending low
v) poor, —"— high.

⑥ printing data clusterwise
↳ we add new column in dataset then give condition for different cluster & print for all clusters.