ASSIGNMENT NO 2

Implement K-means algorithm using R programming

Objective:

To identify the optimized k value using WSS

Theory

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how thealgorithm works, along with the Python implementation of k-means clustering.

K-Means Algorithm

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into differentclusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that eachdataset belongs to only one group that has similar properties.

The k-means clustering algorithm mainly performs two tasks:

Determines the best value for K center points or centroids by an iterative process.
Assigns each data point to its closest k-center. Those data points which are near to the particular k-center,create a cluster.

Hence each cluster has data points with some

commonalities, and it is away from other clusters.K-

Means algorithm:

Step-1: Select the number K to decide the number of clusters.
Step-2: Select random K points or centroids. (It can be other from the input dataset).
Step-3: Assign each data point to their closest centroid,
which will form the predefined K clusters.Step-4: Calculate
the variance and place a new centroid of each cluster.
Step-5: Repeat the third steps, which means reassign each data point to the new closest centroid of eachcluster.

Step-6: If any reassignment occurs,
then go to step-4 else go to FINISH.
Step-7: The model is ready.

Elbow Method:

In cluster analysis, the elbow method is a heuristic used in determining the number of clusters in a data set. Themethod consists of plotting the explained variation as a function of the number of clusters, and picking the elbow of the curve as the number of clusters to use.

Using the "elbow" or "knee of a curve" as a cutoff point is a common heuristic in mathematical optimization to choose a point where diminishing returns are no longer worth the additional cost. In clustering, this means oneshould choose a number of clusters so that adding another cluster doesn't give much better modeling of the data.

The Elbow method looks at the total WSS as a function of the number of clusters: One should choose a numberof clusters so that adding another cluster doesn't improve much better the total WSS. The optimal number of clusters can be defined

Calculate the ***Within-Cluster-Sum of Squared*** *Errors (WSS) for* ***different values of k****, and choose the k for which WSS becomes first starts to diminish. In the plot of WSS-versus-k, this is visible as an* ***elbow.***

Within-Cluster-Sum of Squared Errors sounds a bit complex.
Let's break it down:
The Squared Error for each point is the square of the distance of the point from its representation i.e. its predicted cluster center.

- The WSS score is the sum of these Squared Errors for all the points.
- Any distance metric like the Euclidean Distance or the Manhattan Distance can be used.

To find the optimal value of clusters, the elbow method follows the below steps:

- executes the K-means clustering on a given dataset for different K Valuesfor each value of K, calculates the WSS value.
- Plots a curve between calculated WSS values and the number of clusters K.
- The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the bestvalue of K.

R Program CODE:

(There are some missing lines in the code... Fill it and then execute)

Implementation

```
# install.packages("ggplot2")#   install.packages("dplyr")#
install.packages("ggpubr")
# install.packages("factoextra")
library(ggpubr) library(factoextra) library(dplyr) library(ggplot2)

PATH <-"data.csv"

df <- read.csv(PATH) %>%

select(-c(X, cd, multi, premium)) glimpse(df)

summary(df)

rescale_df <- df %>% mutate(price_scal = scale(price),

hd_scal = scale(hd), ram_scal = scale(ram), screen_scal = scale(screen), ads_scal
= scale(ads),

trend_scal = scale(trend)) %>%

select(-c(price, speed, hd, ram, screen, ads, trend))


#WSS
kmean_withinss <- function(k) { cluster <- kmeans(rescale_df, k)
return (cluster$tot.withinss)
}


# Set maximum cluster max_k <-20
# Run algorithm over a range of k
wss <- sapply(2:max_k, kmean_withinss)


# Create a data frame to plot the graph elbow <-data.frame(2:max_k,
wss)


# Plot the graph with gglop
p <- ggplot(elbow, aes(x = X2.max_k, y = wss)) + geom_point() +
geom_line() +
scale_x_continuous(breaks = seq(1, 20, by = 1))
```

```
# print graph print(p)

k <- 7
res <- kmeans(df, k)

#nstart will try 25 different random starting assignments and then
select the best results.
# res <- kmeans(df, k, nstart = 25)
print(fviz_cluster(res, data = df,
palette = c("#2E9FDF", "#00AFBB", "#E7B800","#1E9FDF",
"#20AFBB", "#E 3B800","#5E9FDF", "#05AFBB","#2A9FDF",
"#50AFBB", "#EEB800"),
geom = "point", ellipse.type = "convex", ggtheme = theme_bw()
))
```

Sample Output

Experimentation

Download any dataset from Kaggle and apply k means to the data set and give your insights on the dataset

Justify why the k value that is selected for the application is the correct value

Output (Sample)

```
Rows: 6,259
Columns: 7
$ price     1499, 1795, 1595, 1849, 3295, 3695, 1720, 1995, 2225, 2575,
219…
$ speed     25, 33, 25, 25, 33, 66, 25, 50, 50, 50, 33, 66, 50, 25, 50, 50,…

$ hd      80, 85, 170, 170, 340,      340, 170, 85, 210, 210, 170, 210, 130, 2…
$ ram     4, 2, 4, 8, 16, 16, 4,      2, 8, 4, 8, 8, 4, 8, 8, 4, 2, 4, 4, 8, 4…
$ screen  14, 14, 15,   14,   14,   14,  14, 14,   14,  15, 15,   14,   14, 14,   14,  14,…
$ ads     94, 94, 94,   94,   94,   94,  94, 94,   94,  94, 94,   94,   94, 94,   94,  94,…
$ trend   1,   1,   1, 1,   1,    1, 1,   1,   1, 1, 1,    1,  1,   1, 1,   1,  1, 1, 1, 1,   1, …
```

price	speed		hd		ram	
Min. : 949	Min. :	25.00	Min. :	80.0	Min. :	2.000
1st Qu.:1794	1st Qu.:	33.00	1st Qu.:	214.0	1st Qu.:	4.000
Median :2144	Median :	50.00	Median :	340.0	Median :	8.000
Mean 2220	Mean :	52.01	Mean :	416.6	Mean :	8.287
3rd Qu.:2595	3rd Qu.:	66.00	3rd Qu.:	528.0	3rd Qu.:	8.000

```
Max.        :5399    Max.   :100.00    Max.   :2100.0    Max.   :32.000
```

```
     screen            ads              trend
 Min.   :14.00   Min.   : 39.0   Min.   : 1.00
 1st Qu.:14.00   1st Qu.:162.5   1st Qu.:10.00
 Median :14.00   Median :246.0   Median :16.00
 Mean   :14.61   Mean   :221.3   Mean   :15.93
 3rd Qu.:15.00   3rd Qu.:275.0   3rd Qu.:21.50
 Max.   :17.00   Max.   :339.0   Max.   :35.00
```

K-means clustering with 7 clusters of sizes 470, 1161, 1236, 822, 1367, 959, 244

Cluster means:

	price	speed	hd	ram	screen	ads	trend
1	2782.130	63.58085	1008.5106	20.493617	15.17660	156.9064	25.23404
2	1950.003	44.42636	203.5668	4.298019	14.36003	240.2756	11.71576
3	1502.227	40.64482	251.4061	4.092233	14.23625	221.9426	17.31149
4	2911.227	58.53771	420.0073	11.698297	14.83333	244.2336	11.31265
5	2408.809	52.60497	394.2473	8.523775	14.63350	240.1661	13.31383
6	1952.808	60.04171	583.3326	8.129301	14.71533	179.4056	23.59020
7	3710.697	66.51230	585.3811	12.803279	15.27049	233.5205	11.09016

Within sum of squares by cluster:
[1] 57005095 32617124 53328948 37386498 61142606 63803514 49314504
(between_SS / total_SS = 86.2 %)

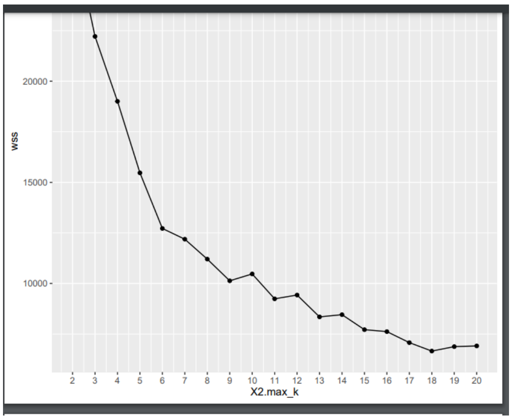
Screenshots (sample)

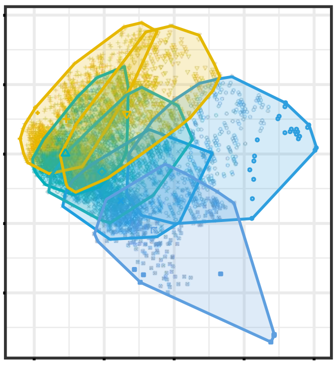
Of WSS vs K

Visualization of clusters for selected K value.

Comment on the clusters formed and what insight it provides

In the below graph at k = 9 the graph there are less changes in the WSS value.





Conclusion:

WSS helps to identify the K value. ***Write in your own words***