R Notebook

Code

Hide

```
library(ggplot2)
```

Hide

```
setwd("C:/Users/Veeru/Desktop/ML/week 2")
getwd()
```

```
[1] "C:/Users/Veeru/Desktop/ML/week 2"
```

Hide

```
df1 <- read.csv("KMeansData_Group1.csv",header=FALSE, sep = ",")
df1 <- na.omit(df1)
str(df1)
```

```
'data.frame':   533 obs. of  2 variables:
 $ V1: num  0.443 0.351 0.498 0.45 0.594 ...
 $ V2: num  0.707 0.443 0.8 0.827 0.3 ...
```

The thumb rule for considering number of clusters: sqrt(2/# observations)

https://www.guru99.com/r-k-means-clustering.html#2 (https://www.guru99.com/r-k-means-clustering.html#2)

Hide

```
k <- round(sqrt(2/nrow(df1)))
k # this does not work.
```

```
[1] 0
```

Initialize a vector of length 10 for the scree plot, “nstart” is the number of times R will restart with different centroids(thumb rule nstart>10). “i” is the number of centers

Hide

```
scree <- rep(0,10) #initializing empty vector
for (i in 1:10){
  df1_kmeans <- kmeans(df1,i,nstart=20)
  scree[i] <- df1_kmeans$tot.withinss/df1_kmeans$totss # the ratio is as per da
tacamp
}
```

Hide

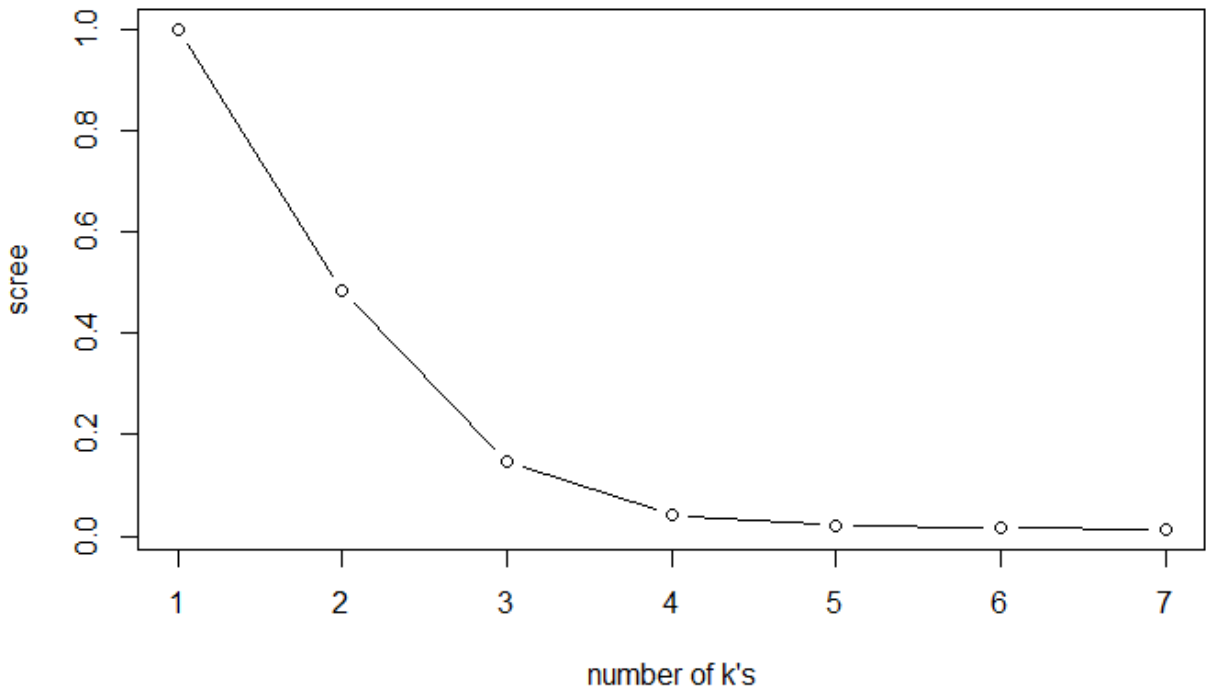
```
scree # 10 possible values for centroids of clusters
```

```
 [1] 1.00000000 0.48297636 0.14583938 0.04195236 0.02054716 0.01714282 0.015554
99
 [8] 0.01215065 0.01156906 0.01043754
```

scree plot, the points below 0.2 or elbow point is best for clustering.

Hide

```
plot(scree, type="b", xlab="number of k's")
```



choosing arbitary number of centroid/clusters: 3 (from scree plot)

Hide

```
k=3
```

Empty vector called cluster_group is created. The motive is to store the assignment values in it.Each observation's distance from the 3 centroids will be calculated,the one which is the minimum is selected and that centroid is assigned to the observation.

Hide

```
df <- read.csv("KMeansData_Group1.csv",header=FALSE, sep = ",")
df <- na.omit(df)
#df <- scale(df) # scaling is standardizing the data
df$cluster_group <- NA
str(df)
```

```
'data.frame':   533 obs. of  3 variables:
 $ V1           : num  0.443 0.351 0.498 0.45 0.594 ...
 $ V2           : num  0.707 0.443 0.8 0.827 0.3 ...
 $ cluster_group: logi  NA NA NA NA NA NA ...
```

3 of the cluster centers are choosed randomly and will be used as centroids. Euclidean distance formula will be used to compute the distance of each observation from three of these centroids

Hide

```
xcen <- sample(df$V1,k)
xcen
```

```
[1] 2.127337 6.279980 9.615745
```

Hide

```
ycen <- sample(df$V2,k)
ycen
```

```
[1] 0.8721779 9.0285042 7.6167140
```

In the above chunk, xcen and ycen are the vectors, which has randomly chosen values from the variables in dataframe df. Three values are chosen which represents the number of centroids from which the distance from rest of the observations will be calculated using euclidean distance formula and the same observations will be assigned to the centroid which is nearest(smaller distance) to them.

create random cluster centers

Hide

```
# this is just for reference  or future use.
#xcen <- runif(k, min = min(df$V1), max = max(df$V1))
#ycen <- runif(k, min = min(df$V2), max = max(df$V2))
```

cluster centroids choosed randomly from df variables are stored in seperate data-frame

Hide

```
#xcen <- runif(k, min = min(df$V1), max = max(df$V1))
#ycen <- runif(k, min = min(df$V2), max = max(df$V2))
```

assign cluster with minimum distance to each observation

Hide

```
for(i in 1:nrow(df)) {
  dist <- sqrt((df$V1[i]-centroid_df$xcen)^2 + (df$V2[i]-centroid_df$ycen)^2)#e
uclidean
  df$cluster_group[i] <- which.min(dist)
#filling the cluster_group variable with the assignments, i.e observations assi
ned to the one of the three cluster.
}
```

(df$V1[i] - centroid_df$xcen)^2 this produces vector of 3 elements

[1] 71.701515767 9.889352276 0.004103784

The first observation of df$V1 and df$ V2 subtracting the vector of the sample selected i.e 3 centroids. The distance which is the minimum is selected and is assigned with the respective cluster number.

Hide

```
head(df)# each obervation is now assigned with their cluster numbers
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.4429445	0.7071066	1
2	0.3514473	0.4426592	1
3	0.4975926	0.7999334	1
4	0.4501654	0.8270090	1
5	0.5940498	0.3003777	1
6	0.6268217	0.7995863	1

6 rows

Hide

```
# checking for the number of observation in each cluster
nrow(subset(df, df$cluster_group == 1))
```

```
[1] 226
```

Hide

```
nrow(subset(df, df$cluster_group == 2))
```

```
[1] 128
```

Hide

```
nrow(subset(df, df$cluster_group == 3))
```

```
[1] 179
```

making another data frame ready for another set of centroids from the same dataframe

Hide

```
x <- rep(NA,k)
y <- rep(NA,k)
centroid_upt <- data.frame(x,y)
centroid_upt
```

x <lgl>	y <lgl>
NA	*NA*
NA	*NA*
NA	*NA*

3 rows

updating the centroids values

Hide

```
for(i in 1:k) {
  centroid_upt[i,1] <- mean(subset(df$V1, df$cluster_group == i))#xcen obs upda
te
  centroid_upt[i,2] <- mean(subset(df$V2, df$cluster_group == i))#ycen obs upda
te
  }
```

when i = 1, for 1st loop, the subset of observations which are assigned to cluster 1 are chosen from both the variables/columns i.e V1 and V2. mean for each variable's observations is calculated and is now the new centroid.Three such centroids are computed since we chose k=3 and stored in the new dataframe centroid_upt. This process will be continued untill there no change in the mean or in the centroids co-ordinates.

Hide

```
centroid_upt
```

x <dbl>	y <dbl>
1.805469	1.207068
4.650801	7.839537
8.594576	2.297604

3 rows

Combining all together

Hide

```
# Kmeans function
create_clusters <- function(df,k) {

  xcen <- sample(df$V1,k)
  ycen <- sample(df$V2,k)
  centroid_df <- data.frame(xcen = xcen, ycen = ycen)
  stop_criteria <- FALSE
    while(stop_criteria == FALSE) {
#filling the cluster_group variable with the assignments, i.e observations assi
ned to the one of the three cluster.
      for(i in 1:nrow(df)) {
        dist <- sqrt((df$V1[i]-centroid_df$xcen)^2 + (df$V2[i]-centroid_df$yce
n)^2)
        df$cluster_group[i] <- which.min(dist)# which will give index number
        }
# storing the sample values computed for centroids, since we will be using to s
top the iteration at a point.
        xcen_old <- centroid_df$xcen
        ycen_old <- centroid_df$ycen
        # updating the centroids values
        for(i in 1:k) {
          centroid_df[i,1] <- mean(subset(df$V1, df$cluster_group == i))
          centroid_df[i,2] <- mean(subset(df$V2, df$cluster_group == i))
          }
        # stop the loop if there is no change in cluster coordinates
        if(identical(xcen_old, centroid_df$xcen) & identical(ycen_old, centroid
_df$ycen))
          stop_criteria <- TRUE
    } # while loop ends here
    the_list <- list(df, centroid_df)
    return(the_list)
}# function ends here
```

Hide

```
df <- read.csv("KMeansData_Group1.csv",header=FALSE, sep = ",")
df <- na.omit(df)
#df <- scale(df)
df$cluster_group <- NA
str(df)
```

```
'data.frame':   533 obs. of  3 variables:
 $ V1           : num  0.443 0.351 0.498 0.45 0.594 ...
 $ V2           : num  0.707 0.443 0.8 0.827 0.3 ...
 $ cluster_group: logi  NA NA NA NA NA NA ...
```

Hide

```
k=3
```

Hide

```
df
```

	V1 <dbl>	V2 <dbl>	cluster_group <lgl>
1	0.44294453	0.707106557	*NA*
2	0.35144735	0.442659199	*NA*
3	0.49759257	0.799933391	*NA*
4	0.45016541	0.827008997	*NA*
5	0.59404983	0.300377731	*NA*
6	0.62682167	0.799586314	*NA*
7	0.57155692	0.229410031	*NA*
8	0.57468328	0.570192935	*NA*
9	0.41352855	0.198721332	*NA*
10	0.21185378	0.698238836	*NA*

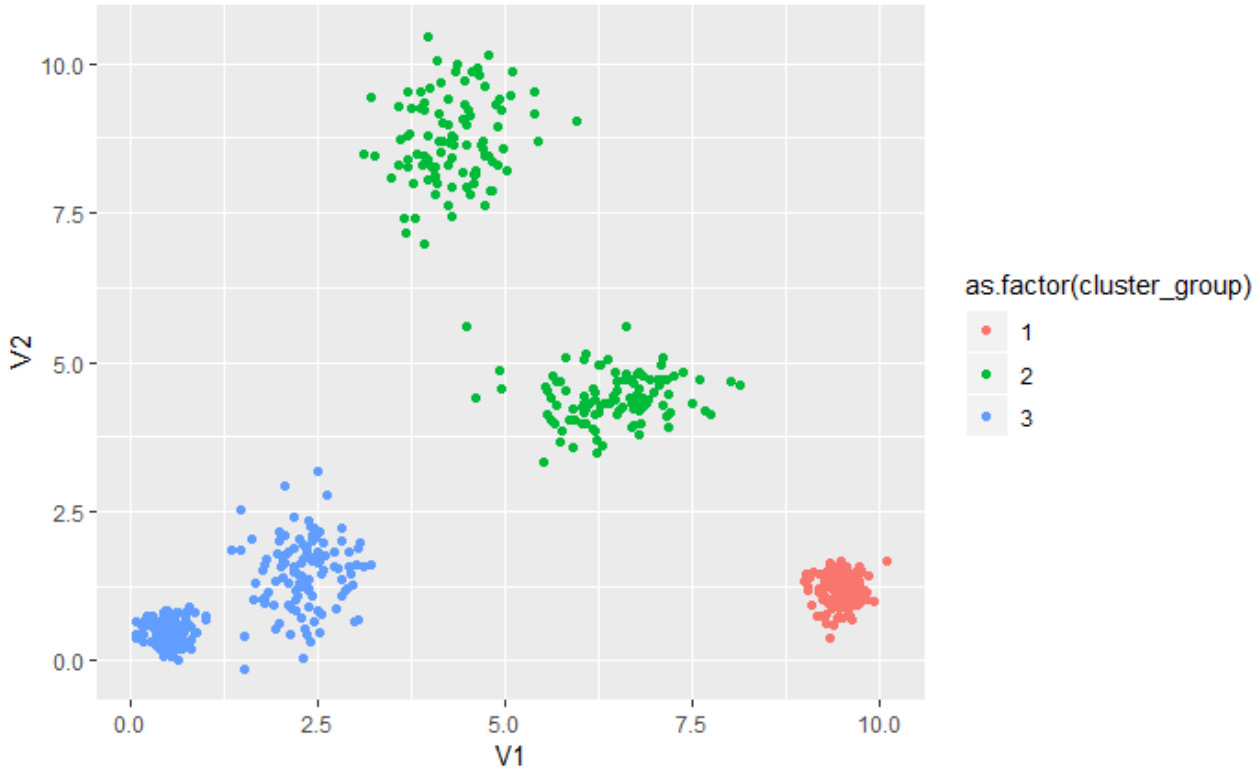
1-10 of 533 rows Previous **1** 2 3 4 5 6 … 54 Next

Hide

```
plot_cluster <- create_clusters(df,3)
data <- data.frame(plot_cluster[1])
centers <- data.frame(plot_cluster[2])
```

Hide

```
ggplot(data, aes(V1, V2, color = as.factor(cluster_group))) + geom_point()
```



The above plot seems to be of 5 clusters not 3

Hide

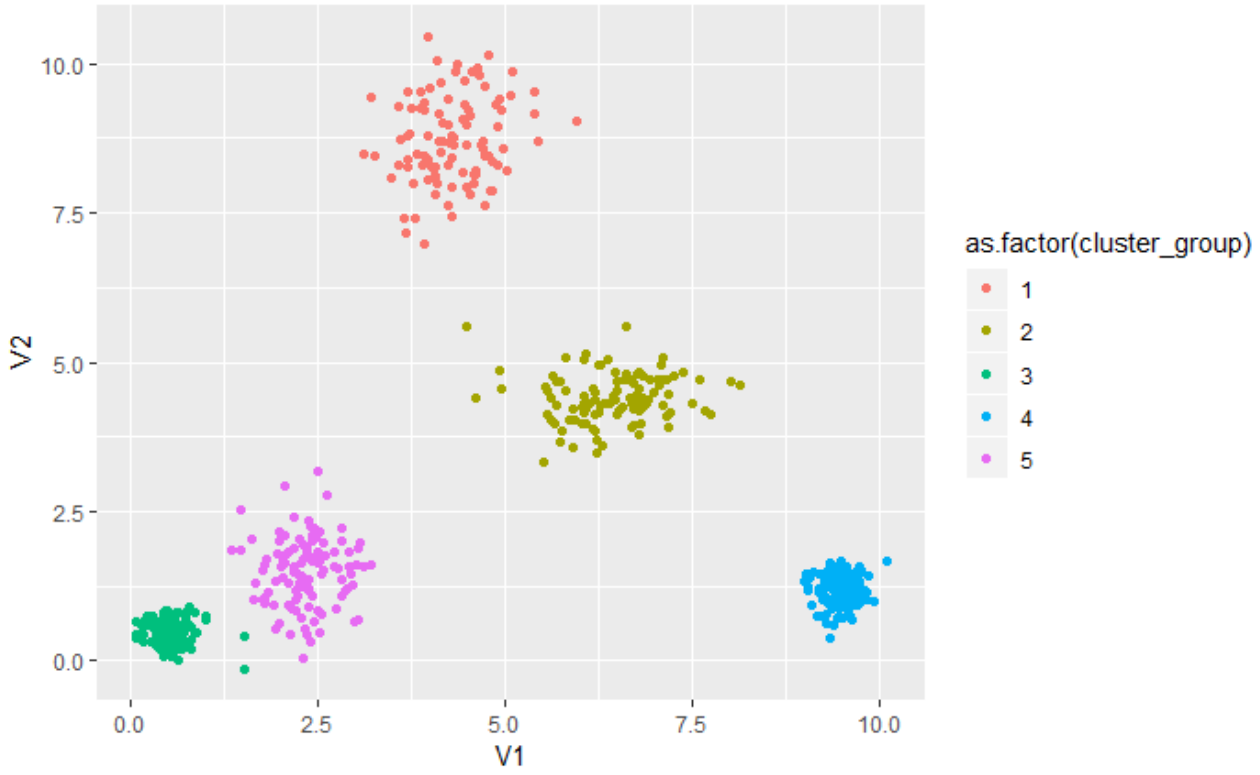
```
pc <- create_clusters(df,5)
data <- data.frame(pc[1])
centers <- data.frame(pc[2])
```

Hide

```
distinct_cluster <- ggplot(data, aes(V1, V2, color = as.factor(cluster_group)))
+ geom_point()
```

Hide

```
distinct_cluster
```



Within sum of square is sum of summed up square distances between each point and the corresponding center of the cluster

“df” is the data-frame which has the observations assigned to their respective centroids

“centroid_df” is the data-frame which have centers of each cluster

Hide

```
wss <- function(df,centroid_df) {
  wss_tot = 0
  for (i in 1:nrow(centroid_df)) {
    sub_set <- subset(df, df$cluster_group == i)
     for (s in 1:nrow(sub_set)) {
       sum_obs <- (sub_set$V1[s] - centroid_df$xcen[i])^2 + (sub_set$V2[s] - ce
ntroid_df$ycen[i])
       wss_tot = wss_tot + sum_obs
     }
  }
  wss_tot
}
```

for loop for clusters values varying from 1 to 10, so that wss is computed and by scree plot optimal clusters can be selected.

Hide

```
scree <- rep(0,10)
for (k in 1:10){
  clusters <- create_clusters(df,k)
  d <- data.frame(clusters[1])
  c <- data.frame(clusters[2])
  scree[k] <- wss(d, c)
}
```

```
Error in scree[k] <- wss(d, c) : replacement has length zero
```

wss for 1 to 7 clusters are calculated, but from 8 to 10 are not.

Hide

```
scree
```

```
 [1] 5517.09006 1754.49328  503.67285  269.57164  269.56429   98.15944   69.461
41
 [8]    0.00000    0.00000    0.00000
```

manually doing the for loop to check the error occurred

Hide

```
wss_df <- rep(0,10) # i will be adding wss values from 1 to 10 clusters manuall
y
```

Only 1 cluster

Hide

```
clusters <- create_clusters(df,1)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	1
2	0.35144735	0.442659199	1
3	0.49759257	0.799933391	1
4	0.45016541	0.827008997	1
5	0.59404983	0.300377731	1

	V1 <dbl>	V2 <dbl>	cluster_group <int>
6	0.62682167	0.799586314	1
7	0.57155692	0.229410031	1
8	0.57468328	0.570192935	1
9	0.41352855	0.198721332	1
10	0.21185378	0.698238836	1

1-10 of 533 rows Previous **1** 2 3 4 5 6 ... 54 Next

Hide

```
c
```

xcen <dbl>	ycen <dbl>
4.768795	3.166096

1 row

wss for one cluster

Hide

```
w <- wss(d,c)
w
```

```
[1] 5517.09
```

Hide

```
wss_df[1] <- w
wss_df
```

```
 [1] 5517.09    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
0.00
```

2 clusters

Hide

```
clusters <- create_clusters(df,2)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	1
2	0.35144735	0.442659199	1
3	0.49759257	0.799933391	1
4	0.45016541	0.827008997	1
5	0.59404983	0.300377731	1
6	0.62682167	0.799586314	1
7	0.57155692	0.229410031	1
8	0.57468328	0.570192935	1
9	0.41352855	0.198721332	1
10	0.21185378	0.698238836	1

1-10 of 533 rows Previous **1** 2 3 4 5 6 … 54 Next

Hide

```
c
```

xcen <dbl>	ycen <dbl>
2.413287	3.523559
8.120040	2.657524

2 rows

wss for 2 clusters

Hide

```
wss_df[2] <- wss(d,c)
wss_df
```

```
 [1] 5517.090 1309.649    0.000    0.000    0.000    0.000    0.000    0.000
0.000
[10]    0.000
```

3 clusters

Hide

```
clusters <- create_clusters(df,3)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	1
2	0.35144735	0.442659199	1
3	0.49759257	0.799933391	1
4	0.45016541	0.827008997	1
5	0.59404983	0.300377731	1
6	0.62682167	0.799586314	1
7	0.57155692	0.229410031	1
8	0.57468328	0.570192935	1
9	0.41352855	0.198721332	1
10	0.21185378	0.698238836	1

1-10 of 533 rows Previous **1** 2 3 4 5 6 ... 54 Next

Hide

```
c
```

xcen <dbl>	ycen <dbl>
1.460688	1.008216
9.481592	1.190304
5.405729	6.513005

3 rows

wss for 3 clusters

Hide

```
wss_df[3] <- wss(d,c)
wss_df
```

```
 [1] 5517.0901 1309.6491  503.6728    0.0000    0.0000    0.0000    0.0000
0.0000
 [9]    0.0000    0.0000
```

4 clusters

Hide

```
clusters <- create_clusters(df,4)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	2
2	0.35144735	0.442659199	2
3	0.49759257	0.799933391	2
4	0.45016541	0.827008997	2
5	0.59404983	0.300377731	2
6	0.62682167	0.799586314	2
7	0.57155692	0.229410031	2
8	0.57468328	0.570192935	2
9	0.41352855	0.198721332	2
10	0.21185378	0.698238836	2

1-10 of 533 rows Previous **1** 2 3 4 5 6 … 54 Next

Hide

```
c
```

	xcen <dbl>	ycen <dbl>

xcen <dbl>	ycen <dbl>
9.481592	1.190304
1.460688	1.008216
4.310716	8.727085
6.448599	4.404357

4 rows

wss for 4 clusters

Hide

```
wss_df[4] <- wss(d,c)
wss_df
```

```
 [1] 5517.0901 1309.6491  503.6728  269.5716    0.0000    0.0000    0.0000
0.0000
 [9]    0.0000    0.0000
```

5 clusters

Hide

```
clusters <- create_clusters(df,5)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	5
2	0.35144735	0.442659199	5
3	0.49759257	0.799933391	5
4	0.45016541	0.827008997	5
5	0.59404983	0.300377731	5
6	0.62682167	0.799586314	5
7	0.57155692	0.229410031	5
8	0.57468328	0.570192935	5

	V1 <dbl>	V2 <dbl>	cluster_group <int>
9	0.41352855	0.198721332	5
10	0.21185378	0.698238836	5

1-10 of 533 rows Previous **1** 2 3 4 5 6 ... 54 Next

Hide

```
c
```

xcen <dbl>	ycen <dbl>
2.3436601	1.4911035
6.4485995	4.4043567
9.4815923	1.1903036
4.3107156	8.7270851
0.5344319	0.5016579

5 rows

wss for 5 clusters

Hide

```
wss_df[5] <- wss(d,c)
wss_df
```

```
 [1] 5517.09006 1309.64915  503.67285  269.57164   98.63926    0.00000    0.000
00    0.00000
 [9]    0.00000    0.00000
```

6 clusters

Hide

```
clusters <- create_clusters(df,6)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

V1 <dbl>	V2 <dbl>	cluster_group <int>

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	3
2	0.35144735	0.442659199	3
3	0.49759257	0.799933391	3
4	0.45016541	0.827008997	3
5	0.59404983	0.300377731	3
6	0.62682167	0.799586314	3
7	0.57155692	0.229410031	3
8	0.57468328	0.570192935	3
9	0.41352855	0.198721332	3
10	0.21185378	0.698238836	3

1-10 of 533 rows Previous **1** 2 3 4 5 6 ... 54 Next

Hide

```
c
```

xcen <dbl>	ycen <dbl>
4.3107156	8.7270851
9.4815923	1.1903036
0.5344319	0.5016579
5.8626116	4.2814506
6.9054713	4.5001817
2.3436601	1.4911035

6 rows

wss for 6 clusters

Hide

```
wss_df[6] <- wss(d,c)
wss_df
```

```
 [1] 5517.09006 1309.64915  503.67285  269.57164   98.63926   98.63926    0.000
00    0.00000
 [9]    0.00000    0.00000
```

7 clusters gives out NAN as centoids, this is the reason the for loop above was producing error

Hide

```
clusters <- create_clusters(df,7)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	6
2	0.35144735	0.442659199	6
3	0.49759257	0.799933391	6
4	0.45016541	0.827008997	6
5	0.59404983	0.300377731	6
6	0.62682167	0.799586314	6
7	0.57155692	0.229410031	6
8	0.57468328	0.570192935	6
9	0.41352855	0.198721332	6
10	0.21185378	0.698238836	6

1-10 of 533 rows Previous **1** 2 3 4 5 6 … 54 Next

Hide

```
c
```

xcen <dbl>	ycen <dbl>
NaN	NaN
4.310716	8.7270851
9.474428	1.3849314

xcen <dbl>	ycen <dbl>
6.448599	4.4043567
9.490216	0.9560295
1.460688	1.0082162
NaN	NaN

7 rows

The chunk of code is repeatedly re-runned for NAN to be removed.

Hide

```
clusters <- create_clusters(df,7)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	5
2	0.35144735	0.442659199	5
3	0.49759257	0.799933391	5
4	0.45016541	0.827008997	5
5	0.59404983	0.300377731	5
6	0.62682167	0.799586314	5
7	0.57155692	0.229410031	5
8	0.57468328	0.570192935	5
9	0.41352855	0.198721332	5
10	0.21185378	0.698238836	5

1-10 of 533 rows Previous **1** 2 3 4 5 6 … 54 Next

Hide

```
c
```

xcen <dbl>	ycen <dbl>

xcen <dbl>	ycen <dbl>
2.3436601	1.4911035
4.2123514	8.2519355
6.9369279	4.4873758
9.4815923	1.1903036
0.5344319	0.5016579
5.8905098	4.3094776
4.4582620	9.4398095

7 rows

wss for 7 clusters

8 clusters after number of time re-running the chunk

Hide

```
clusters <- create_clusters(df,8)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	8
2	0.35144735	0.442659199	8
3	0.49759257	0.799933391	8
4	0.45016541	0.827008997	8
5	0.59404983	0.300377731	8
6	0.62682167	0.799586314	8
7	0.57155692	0.229410031	8
8	0.57468328	0.570192935	8
9	0.41352855	0.198721332	8
10	0.21185378	0.698238836	8

1-10 of 533 rows Previous **1** 2 3 4 5 6 ... 54 Next

Hide

```
c
```

xcen <dbl>	ycen <dbl>
9.4815923	1.1903036
2.2144108	0.8894687
5.8905098	4.3094776
2.6975626	1.6045346
6.9369279	4.4873758
2.1205750	2.0203725
4.3107156	8.7270851
0.5245982	0.5025811

8 rows

wss for 8 clusters

Hide

```
wss_df[8] <- wss(d,c)
wss_df
```

```
 [1] 5517.09006 1309.64915  503.67285  269.57164   98.63926   98.63926   68.572
17   63.04142
 [9]    0.00000    0.00000
```

9 clusters(re-run chunk)

Hide

```
clusters <- create_clusters(df,9)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	6
2	0.35144735	0.442659199	6

	V1 <dbl>	V2 <dbl>	cluster_group <int>
3	0.49759257	0.799933391	6
4	0.45016541	0.827008997	6
5	0.59404983	0.300377731	6
6	0.62682167	0.799586314	6
7	0.57155692	0.229410031	6
8	0.57468328	0.570192935	6
9	0.41352855	0.198721332	6
10	0.21185378	0.698238836	6

1-10 of 533 rows Previous **1** 2 3 4 5 6 … 54 Next

Hide

```
c
```

xcen <dbl>	ycen <dbl>
5.862612	4.2814506
4.212351	8.2519355
2.100616	0.7881664
4.458262	9.4398095
6.905471	4.5001817
0.514492	0.5089155
2.695289	1.4366917
2.168817	1.9767393
9.481592	1.1903036

9 rows

wss for 9 clusters

Hide

```
wss_df[9] <- wss(d,c)
wss_df
```

```
 [1] 5517.09006 1309.64915  503.67285  269.57164   98.63926   98.63926   68.572
17   63.04142
 [9]   60.73801    0.00000
```

10 clusters

Hide

```
clusters <- create_clusters(df,10)
d <- data.frame(clusters[1])
c <- data.frame(clusters[2])
d
```

	V1 <dbl>	V2 <dbl>	cluster_group <int>
1	0.44294453	0.707106557	7
2	0.35144735	0.442659199	7
3	0.49759257	0.799933391	7
4	0.45016541	0.827008997	7
5	0.59404983	0.300377731	7
6	0.62682167	0.799586314	7
7	0.57155692	0.229410031	7
8	0.57468328	0.570192935	7
9	0.41352855	0.198721332	7
10	0.21185378	0.698238836	7

1-10 of 533 rows Previous **1** 2 3 4 5 6 ... 54 Next

Hide

```
c
```

xcen <dbl>	ycen <dbl>
5.7886076	4.2224537
9.4815923	1.1903036
4.2123514	8.2519355
7.3329007	4.5311109

xcen <dbl>	ycen <dbl>
6.5901073	4.4967582
2.1269214	1.9956098
0.5245982	0.5025811
2.7837245	1.6376072
2.2246254	0.9364170
4.4582620	9.4398095

1-10 of 10 rows

wss for 10 clusters.

Hide

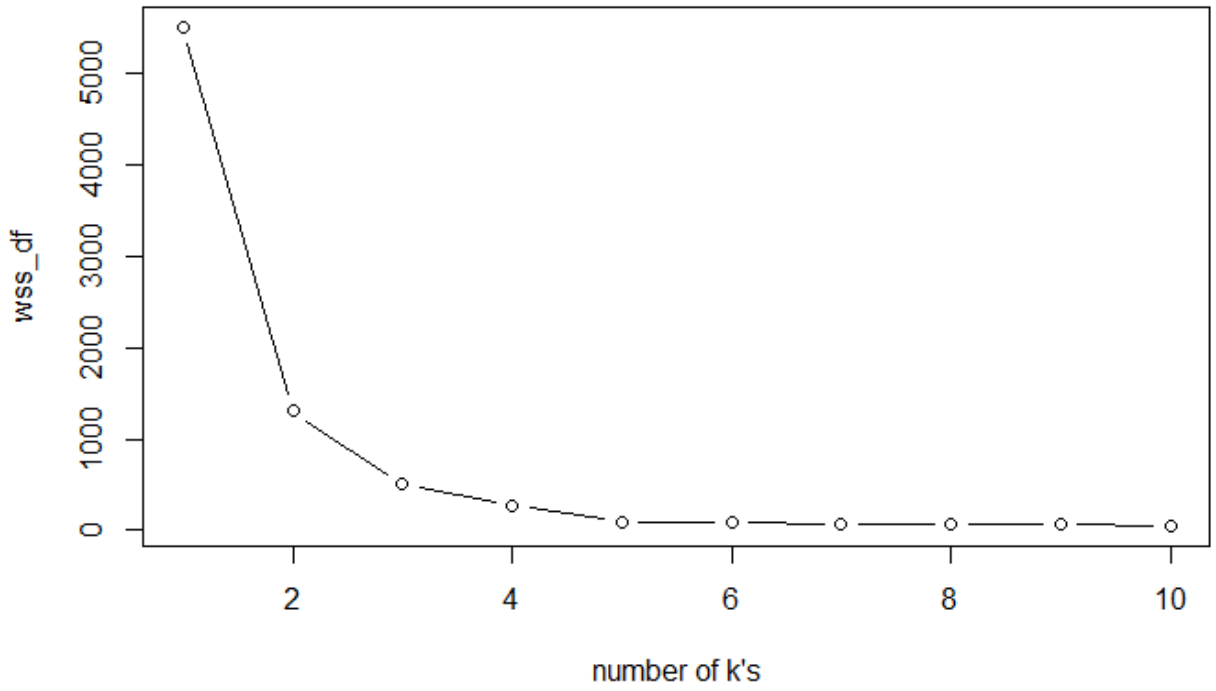
```
wss_df[10] <- wss(d,c)
wss_df
```

```
 [1] 5517.09006 1309.64915  503.67285  269.57164   98.63926   98.63926   68.572
17   63.04142
 [9]   60.73801   55.19255
```

elbow/scree plot is plotted for wss vs number of clusters and as per the elbow point which is 3 in this case, 3 should be choosed for computing kmeans for 3 number of clusters. In the scatter plot above we checked for both 3 and 5 clusters individually

Hide

```
number_of_clusters <- 1:10
plot(number_of_clusters,wss_df, type="b", xlab="number of k's")
```



links referred:

http://dni-institute.in/blogs/k-means-clustering-algorithm-explained/ (http://dni-institute.in/blogs/k-means-clustering-algorithm-explained/) https://uc-r.github.io/kmeans_clustering#distance (https://uc-r.github.io/kmeans_clustering#distance)
http://enhancedatascience.com/2017/10/24/machine-learning-explained-kmeans/ (http://enhancedatascience.com/2017/10/24/machine-learning-explained-kmeans/)
https://github.com/mehdimo/K-Means/blob/master/kmeans_mehdi.R (https://github.com/mehdimo/K-Means/blob/master/kmeans_mehdi.R)
https://stackoverflow.com/questions/41912875/writing-own-kmeans-algorithm-in-r (https://stackoverflow.com/questions/41912875/writing-own-kmeans-algorithm-in-r)
https://www.youtube.com/watch?v=j9ZPMlVHJVs (https://www.youtube.com/watch?v=j9ZPMlVHJVs)
within sum of square