Lab 3

Group 1: Laxman Panthi & Zubin Shah

Part 1

Overview:

This is part 1 of Lab 3 and here we are going to consider the Wholesale Customer dataset and perform unsupervised learning technique KNN (K- Nearest Neighbor) to determine the optimum clusters for our customers.

Let's start with importing all the libraries that we would require for performing the analysis.

```
# load libraries
library(tidyverse)

library(cluster)

library(NbClust)

library(rpart.plot)

library(RColorBrewer)
library(rpart)
library(rattle)
```

Data Preparation:

Now we will load the dataset and pre-process the data. Since the focus is to perform cluster analysis based on items bought by the customers; our top customers are not that useful for providing the insights so we will remove those customers from the list.

Wholesale Customers Dataset: R Code

Step 1:

Load the dataset and create a function to remove top n customers.

```
data = data.frame(read_csv("wholesale.csv"))

## Parsed with column specification:
## cols(
##   Channel = col_double(),
##   Region = col_double(),
##   Fresh = col_double(),
```

```
##   Milk = col_double(),
##   Grocery = col_double(),
##   Frozen = col_double(),
##   Detergents_Paper = col_double(),
##   Delicassen = col_double()
## )

top.n.custs <- function (data, cols, n = 5) {
  #Initialize a vector to hold customers being removed
  idx.to.remove <- integer(0)
  for (c in cols) {
    # For every column in the data we passed to this function
    #Sort column "c" in descending order (bigger on top)
    #Order returns the sorted index (e.g. row 15, 3, 7, 1, ...) rather than t
he actual values sorted.
    col.order <- order(data[, c], decreasing = T)
    #Take the first n of the sorted column C to
    #combine and de-duplicate the row ids that need to be removed
    idx <- head(col.order, n)
    idx.to.remove <- union(idx.to.remove, idx)
  }
  #Return the indexes of customers to be removed
  return(idx.to.remove)
}
```

Step 2:

Evaluate the number of customers removed from the list, evaluate and summarize the dataset.

```
#How Many Customers to be Removed?
top.custs <-top.n.custs(data, cols = 1:5,n=5)
length(top.custs)

## [1] 18

#Examine the customers
data[top.custs,]

##     Channel Region  Fresh  Milk Grocery Frozen Detergents_Paper Delicassen
## 1         2      3  12669  9656    7561    214             2674       1338
## 2         2      3   7057  9810    9568   1762             3293       1776
## 3         2      3   6353  8808    7684   2405             3516       7844
## 5         2      3  22615  5410    7198   3915             1777       5185
## 6         2      3   9413  8259    5126    666             1795       1451
## 4         1      3  13265  1196    4221   6404              507       1788
## 182       1      3 112151 29627   18148  16745             4948       8550
## 126       1      3  76237  3473    7102  16538              778        918
## 285       1      3  68951  4411   12609   8692              751       2406
## 40        1      3  56159   555     902  10002              212       2916
## 259       1      1  56083  4563    2124   6422              730       3321
```

```
## 87        2      3  22925 73498   32114    987             20070        903
## 48        2      3  44466 54259   55571   7782             24171       6465
## 86        2      3  16117 46197   92780   1026             40827       2944
## 184       1      3  36847 43950   20170  36534               239      47943
## 62        2      3  35942 38369   59598   3254             26701       2017
## 334       2      2   8565  4980   67298    131             38102       1215
## 66        2      3     85 20959   45828     36             24231       1423
```

```
#Remove the Customers
data.rm.top<-data[-c(top.custs),]
#Examine summary stats for the remaining data
print(summary(data.rm.top))
```

```
##     Channel         Region          Fresh            Milk
##  Min.   :1.00   Min.   :1.000   Min.   :    3   Min.   :   55
##  1st Qu.:1.00   1st Qu.:2.000   1st Qu.: 3072   1st Qu.: 1497
##  Median :1.00   Median :3.000   Median : 8130   Median : 3582
##  Mean   :1.31   Mean   :2.531   Mean   :11076   Mean   : 5172
##  3rd Qu.:2.00   3rd Qu.:3.000   3rd Qu.:16251   3rd Qu.: 6962
##  Max.   :2.00   Max.   :3.000   Max.   :56082   Max.   :36423
##     Grocery          Frozen        Detergents_Paper    Delicassen
##  Min.   :    3   Min.   :   25.0   Min.   :    3.0   Min.   :    3.0
##  1st Qu.: 2132   1st Qu.:  738.8   1st Qu.:  255.2   1st Qu.:  398.0
##  Median : 4603   Median : 1487.5   Median :  799.5   Median :  904.5
##  Mean   : 7211   Mean   : 2910.3   Mean   : 2541.6   Mean   : 1352.0
##  3rd Qu.:10391   3rd Qu.: 3428.0   3rd Qu.: 3879.2   3rd Qu.: 1752.2
##  Max.   :39694   Max.   :60869.0   Max.   :19410.0   Max.   :16523.0
```

From the above it can be seen that we are removing top 18 customers from the list.

Cluster Analysis: Using KNN Technique

Now since data has been pre-processed, lets run KNN upto 20 clusters for 100 trials and review what should the optimal clusters be. Also, we will finalize our k-means analysis using within and between sum of squares.

Step 3:

Create seed for reproducibility and perform 100 trials for k ranging from 2 to 20 and evaluate the means with between and within.

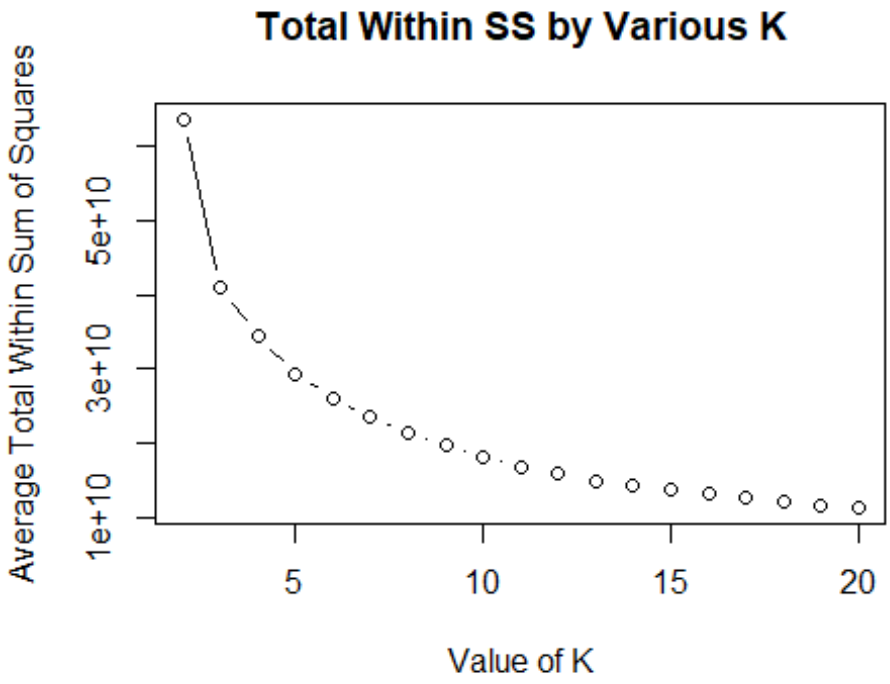
```
#Set the seed for reproducibility
set.seed(76964057)
#Try K from 2 to 20
rng<-2:20
#Number of times to run the K Means algorithm
tries <-100
```

```r
#Set up an empty vector to hold all of points
avg.totw.ss <-integer(length(rng))
avg.totb.ss <- integer(length(rng))
avg.tot.ss <- integer(length(rng))

# For each value of the range variable
for (v in rng) {
  #Set up an empty vectors to hold the tries
  v.totw.ss <- integer(tries)
  b.totb.ss <- integer(tries)
  tot.ss <- integer(tries)
  #Run kmeans
  for (i in 1:tries) {
    k.temp <- kmeans(data.rm.top, centers = v)
    #Store the total withinss
    v.totw.ss[i] <- k.temp$tot.withinss
    #Store the betweenss
    b.totb.ss[i] <- k.temp$betweenss
    #Store the total sum of squares
    tot.ss[i] <- k.temp$totss
  }
  #Average the withinss and betweenss
  avg.totw.ss[v - 1] <- mean(v.totw.ss)
  avg.totb.ss[v - 1] <- mean(b.totb.ss)
  avg.tot.ss[v - 1] <- mean(tot.ss)
}
```

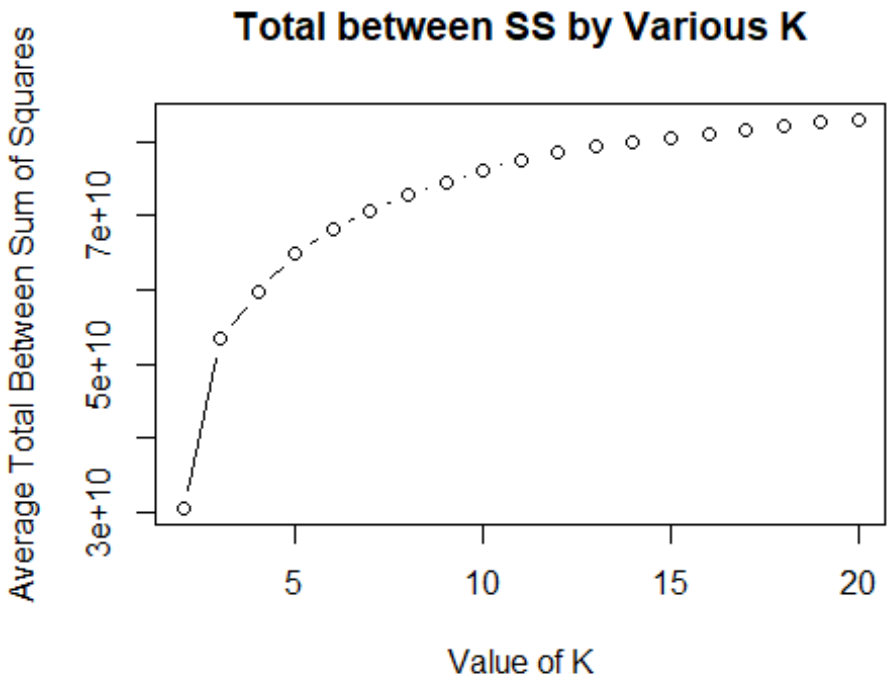
Step 4:

Plot the total within and between sum of squares by various k values along with their ratios.

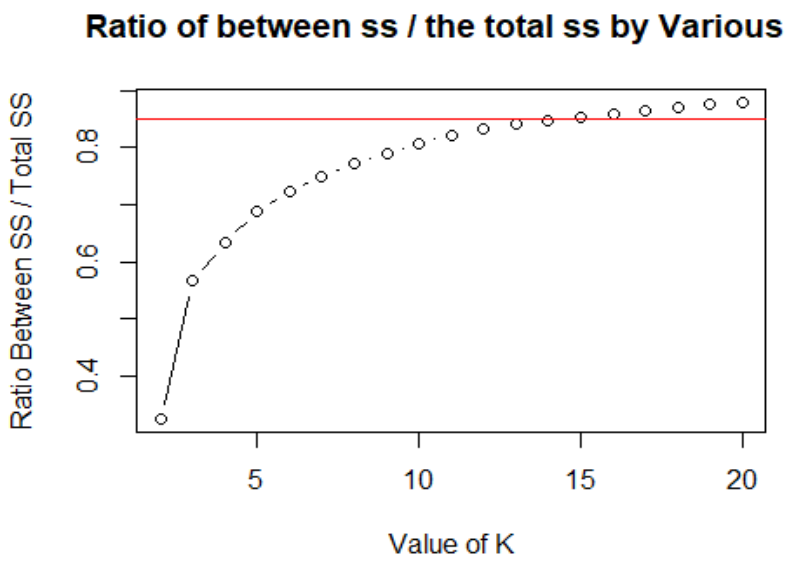
```r
plot(rng,avg.totw.ss,type="b", main="Total Within SS by Various K",
ylab="Average Total Within Sum of Squares",
xlab="Value of K")
```



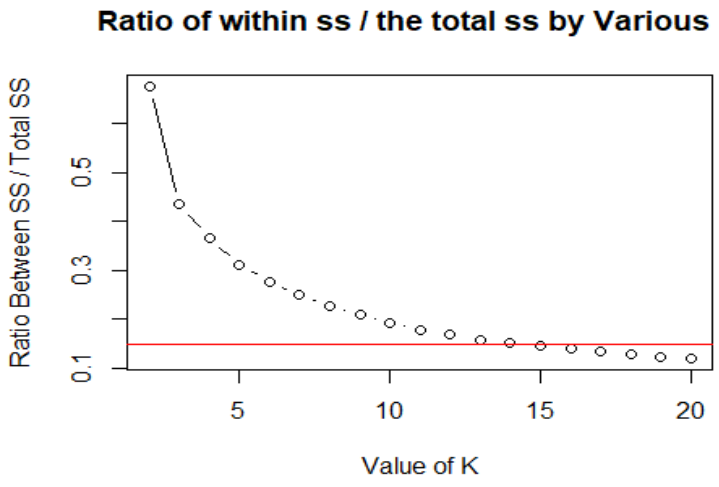

```
plot(rng,avg.totb.ss,type="b", main="Total between SS by Various K",
ylab="Average Total Between Sum of Squares",
xlab="Value of K")
```



```r
#Plot the ratio of betweenss/total ss and withinss / total ss for evaluation
plot(rng,avg.totb.ss/avg.tot.ss,type="b", main="Ratio of between ss / the tot
al ss by Various K",
ylab="Ratio Between SS / Total SS",
xlab="Value of K")
abline(h=0.85, col="red")
```




```r
plot(rng,avg.totw.ss/avg.tot.ss,type="b", main="Ratio of within ss / the tota
l ss by Various K",
ylab="Ratio Between SS / Total SS",
xlab="Value of K")
abline(h=0.15, col="red")
```



Step 5:

From the above graphs, decide the optimal k value (k=5) to create k clusters of the given data. Evaluate the clusters and plot it.

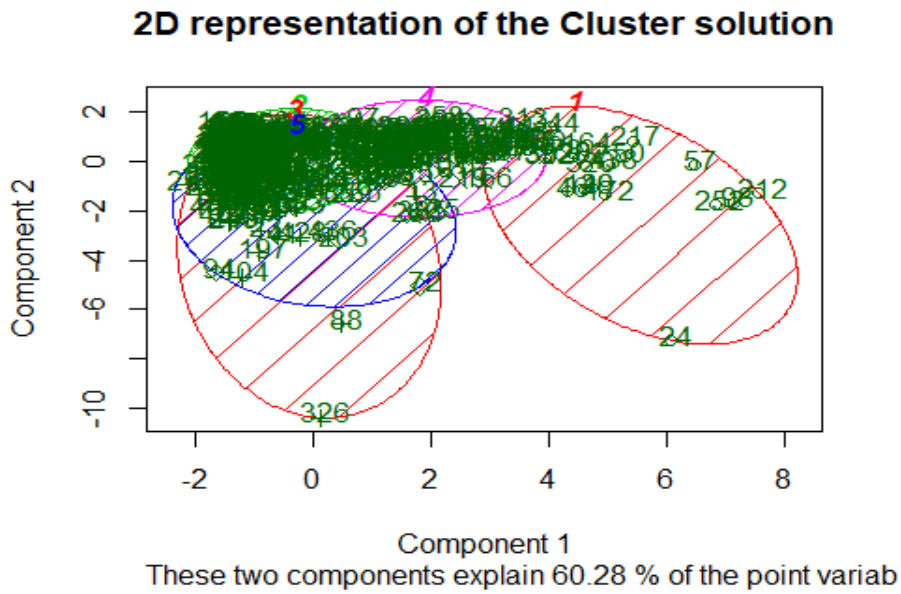
```
#Create the best number of clusters, Remove columns 1 and 2
#n <- readline(prompt = "Enter the best number of clusters: ")
n <- as.integer(5)
k <-kmeans(data.rm.top[,-c(1,2)], centers=n)
#Display cluster centers
print(k$centers)
```

```
##       Fresh      Milk   Grocery   Frozen Detergents_Paper Delicassen
## 1  7944.286 19517.476 27404.619 2176.714       12311.0000  2982.7619
## 2  5585.596  2742.142  3272.519 2669.568         880.9891   887.2678
## 3 37275.906  5219.844  5850.094 6883.125         824.8125  2169.3438
## 4  4391.893  9045.202 14218.262 1397.452        6083.0238  1408.0714
## 5 18857.735  3371.235  4775.049 3492.618        1131.5196  1547.4020
```

```
#Give a count of data points in each cluster
print(table(k$cluster))
```

```
##
##   1   2   3   4   5
##  21 183  32  84 102
```

```
clusplot(data.rm.top, k$cluster, main='2D representation of the Cluster solut
ion',
color=TRUE, shade=TRUE,
labels=2, lines=0)
```



Conclusion:

From the above, it is clearly observed that k=5 is the optimal number of clusters suggested from the within and between sum of squares.

Summary:

Wholesale Customer dataset was taken and pre-processed to exclude the top customers from the study. Then we evaluated the model with a range of k-means and based on the statistical analysis, we conclude that 5 (k=5) clusters are optimum to divide our customer data.

Part 2

Overview:

This is part 2 of Lab 3 and here we are going to consider the Wine dataset and perform unsupervised learning technique KNN (K- Nearest Neighbor) to determine the optimum clusters based on the contents of the wine. The dataset has 13 chemical measurements on 178 observations of Italian wine.

Let's start with importing all the libraries that we would require for performing the analysis.

```
# load libraries
library(tidyverse)

library(cluster)

library(NbClust)

library(rpart.plot)

library(RColorBrewer)
library(rpart)
library(rattle)
```

Data Pre-Processing: R Code

Step 1:

In this data, we do not need much pre-processing. We will separate our dependent variable and just standardize our data using the scale function in R.

```
#Load data into R/RStudio and view it
wine <- read.csv("wine.csv")
df <- scale(wine[-1])
#Examine the data frame and plot the within sum of squares
head(df)
```

```
##        Alcohol  Malic.acid        Ash        Acl         Mg   Phenols
## [1,] 1.5143408 -0.56066822  0.2313998 -1.1663032 1.90852151 0.8067217
## [2,] 0.2455968 -0.49800856 -0.8256672 -2.4838405 0.01809398 0.5670481
## [3,] 0.1963252  0.02117152  1.1062139 -0.2679823 0.08810981 0.8067217
## [4,] 1.6867914 -0.34583508  0.4865539 -0.8069748 0.92829983 2.4844372
## [5,] 0.2948684  0.22705328  1.8352256  0.4506745 1.27837900 0.8067217
## [6,] 1.4773871 -0.51591132  0.3043010 -1.2860793 0.85828399 1.5576991
##      Flavanoids Nonflavanoid.phenols    Proanth  Color.int        Hue
## [1,]  1.0319081           -0.6577078  1.2214385  0.2510088  0.3611585
## [2,]  0.7315653           -0.8184106 -0.5431887 -0.2924962  0.4049085
## [3,]  1.2121137           -0.4970050  2.1299594  0.2682629  0.3174085
## [4,]  1.4623994           -0.9791134  1.0292513  1.1827317 -0.4263410
## [5,]  0.6614853            0.2261576  0.4002753 -0.3183774  0.3611585
## [6,]  1.3622851           -0.1755994  0.6623487  0.7298108  0.4049085
```

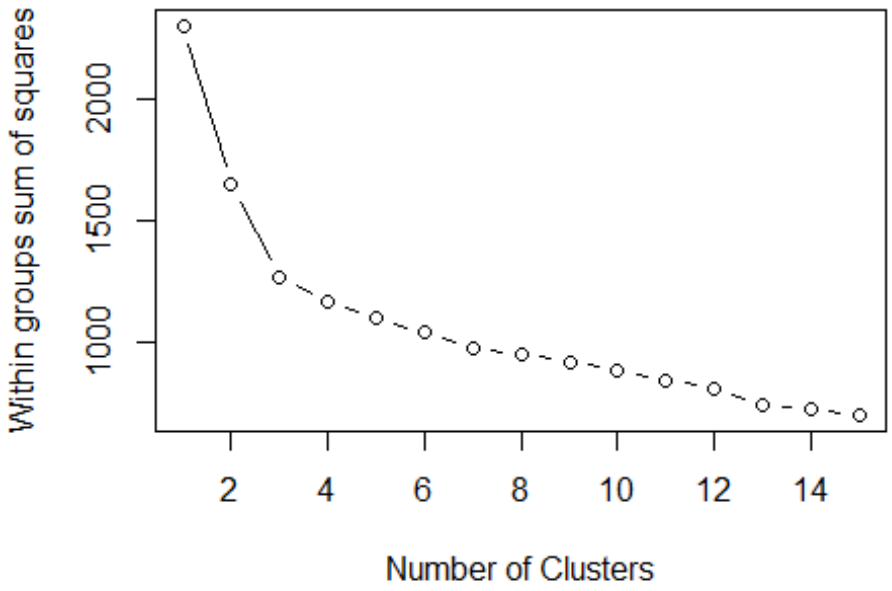
```
##              OD     Proline
## [1,] 1.8427215  1.01015939
## [2,] 1.1103172  0.96252635
## [3,] 0.7863692  1.39122370
## [4,] 1.1807407  2.32800680
## [5,] 0.4483365 -0.03776747
## [6,] 0.3356589  2.23274072
```

Clustering: Using KNN Technique

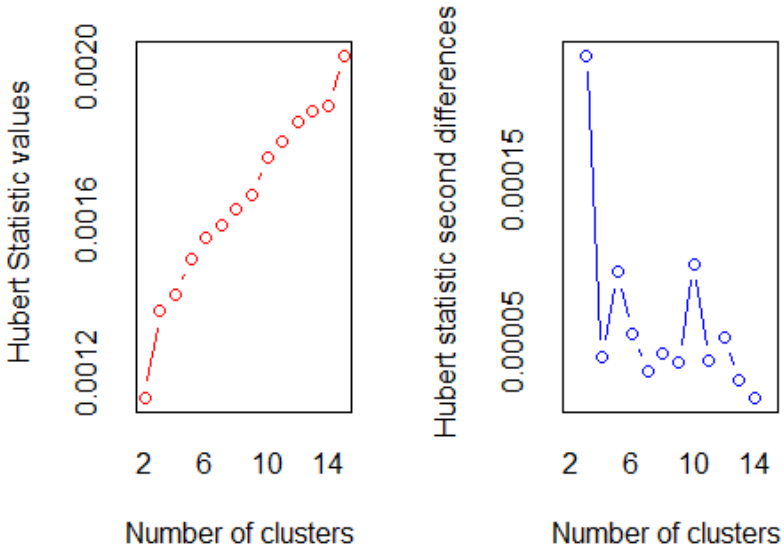
Step 2:

Here we will first plot the within sum of squares to find the initial value of k and then we would finalize the number of clusters based on table and barplot.

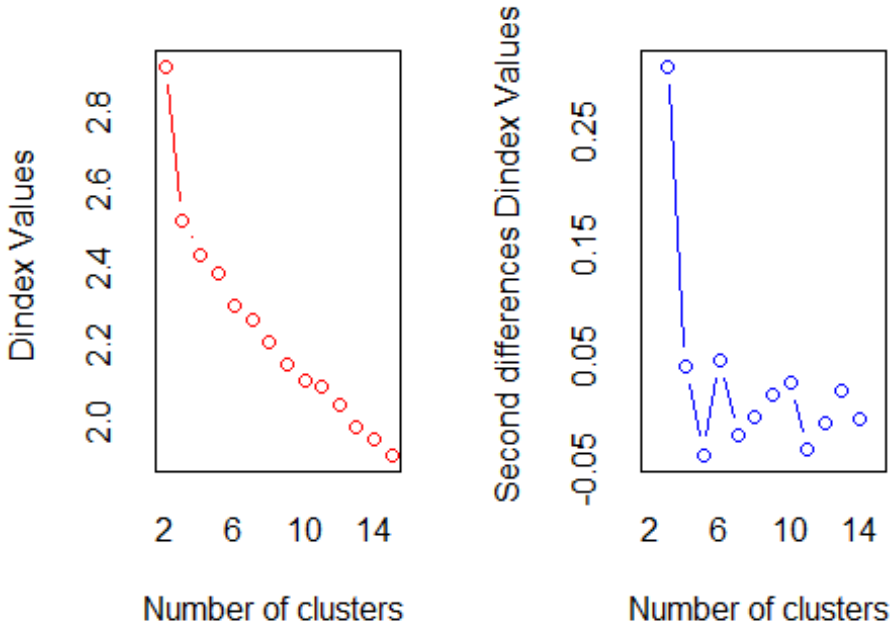
```
#Plot the within (cluster) sum of squares to determine the initial value for
"k"
wssplot <- function(data, nc = 15, seed = 1234) {
  wss <- (nrow(data) - 1) * sum(apply(data, 2, var))
  for (i in 2:nc) {
    set.seed(seed)
    wss[i] <- sum(kmeans(data, centers = i)$withinss)
  }
  plot(1:nc,
       wss,
       type = "b",
       xlab = "Number of Clusters",
       ylab = "Within groups sum of squares")
}

wssplot(df)
```




```
#Start the k-Means analysis using the variable "nc" for the number of cluster
s
set.seed(1234)
nc <- NbClust(df, min.nc=2, max.nc = 15, method = "kmeans")
```



```
## *** : The Hubert index is a graphical method of determining the number of
clusters.
##                 In the plot of Hubert index, we seek a significant knee th
at corresponds to a
##                 significant increase of the value of the measure i.e the s
ignificant peak in Hubert
##                 index second differences plot.
##
```



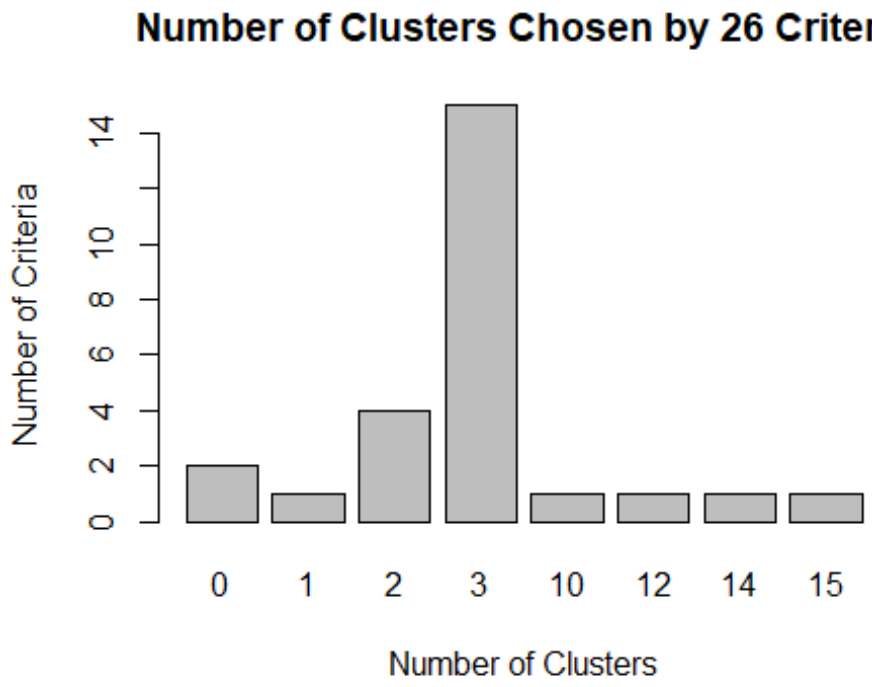

```
## *** : The D index is a graphical method of determining the number of clust
ers.
##                 In the plot of D index, we seek a significant knee (the si
gnificant peak in Dindex
##                 second differences plot) that corresponds to a significant
increase of the value of
##                 the measure.
##
## *****************************************************************
## * Among all indices:
## * 4 proposed 2 as the best number of clusters
## * 15 proposed 3 as the best number of clusters
## * 1 proposed 10 as the best number of clusters
## * 1 proposed 12 as the best number of clusters
## * 1 proposed 14 as the best number of clusters
## * 1 proposed 15 as the best number of clusters
##
##                    ***** Conclusion *****
```

```
##
## * According to the majority rule, the best number of clusters is  3
##
##
## *******************************************************************
```

```
print(table(nc$Best.n[1,]))
```

```
##
##  0  1  2  3 10 12 14 15
##  2  1  4 15  1  1  1  1
```

```
barplot(table(nc$Best.n[1,]), xlab = "Number of Clusters", ylab = "Number of
Criteria", main = "Number of Clusters Chosen by 26 Criteria")
```



Step 3:

From the above results, decide the optimal k value (k=3), conduct k means analysis and evaluate the centers of the clusters.

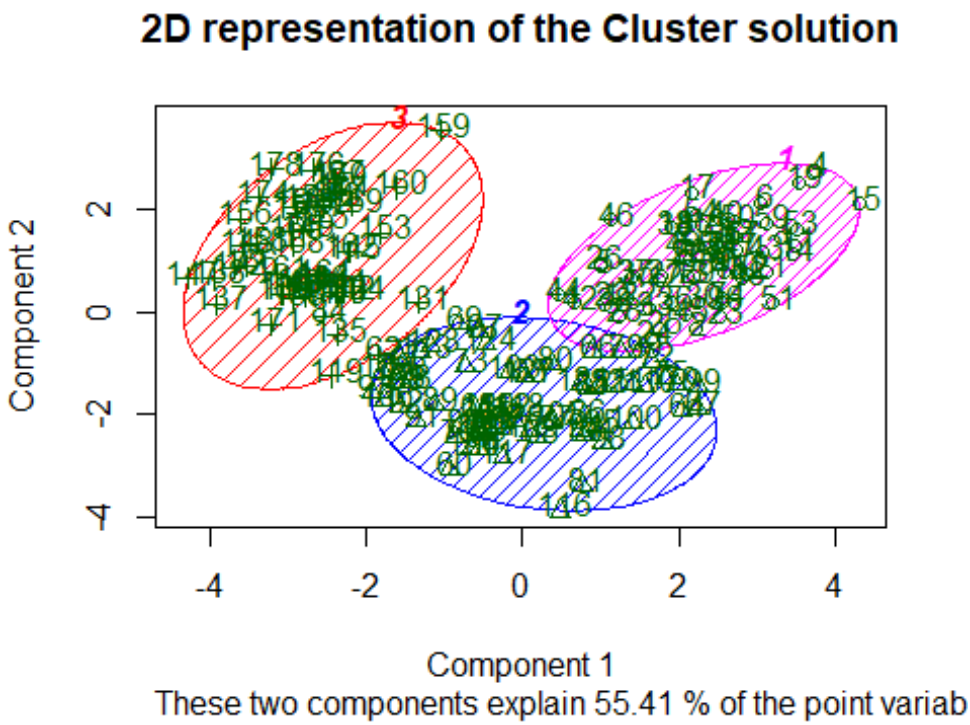
```
#Enter the best number of clusters based on the information in the table and
barplot
n <- 3

#Conduct the k-Means analysis using the best number of clusters
set.seed(1234)
fit.km <- kmeans(df, n, nstart=25)
print(fit.km$size)
```

```
## [1] 62 65 51

print(fit.km$centers)

##      Alcohol Malic.acid        Ash        Acl          Mg     Phenols
## 1  0.8328826 -0.3029551  0.3636801 -0.6084749  0.57596208  0.88274724
## 2 -0.9234669 -0.3929331 -0.4931257  0.1701220 -0.49032869 -0.07576891
## 3  0.1644436  0.8690954  0.1863726  0.5228924 -0.07526047 -0.97657548
##    Flavanoids Nonflavanoid.phenols     Proanth  Color.int        Hue
## 1  0.97506900          -0.56050853  0.57865427  0.1705823  0.4726504
## 2  0.02075402          -0.03343924  0.05810161 -0.8993770  0.4605046
## 3 -1.21182921           0.72402116 -0.77751312  0.9388902 -1.1615122
##           OD    Proline
## 1  0.7770551  1.1220202
## 2  0.2700025 -0.7517257
## 3 -1.2887761 -0.4059428

print(aggregate(wine[-1], by=list(cluster=fit.km$cluster), mean))

##   cluster  Alcohol Malic.acid      Ash      Acl        Mg  Phenols
## 1       1 13.67677   1.997903 2.466290 17.46290 107.96774 2.847581
## 2       2 12.25092   1.897385 2.231231 20.06308  92.73846 2.247692
## 3       3 13.13412   3.307255 2.417647 21.24118  98.66667 1.683922
##   Flavanoids Nonflavanoid.phenols  Proanth Color.int       Hue       OD
## 1  3.0032258            0.2920968 1.922097  5.453548 1.0654839 3.163387
## 2  2.0500000            0.3576923 1.624154  2.973077 1.0627077 2.803385
## 3  0.8188235            0.4519608 1.145882  7.234706 0.6919608 1.696667
##     Proline
## 1 1100.2258
## 2  510.1692
## 3  619.0588
```

Step 4:

Creating a confusion matrix to evaluate the prediction accuracy and plot the clusters.

```
#Use a confusion or truth table to evaluate how well the k-Means analysis per
formed
ct.km <- table(wine$Wine, fit.km$cluster)
print(ct.km)

##
##      1  2  3
##   1 59  0  0
##   2  3 65  3
##   3  0  0 48

#Generate a plot of the clusters
clusplot(df, fit.km$cluster, main='2D representation of the Cluster solution'
,
color=TRUE, shade=TRUE,
labels=2, lines=0)
```



Conclusion:

From the above, it is clearly observed that k=3 is the optimal number of clusters suggested from the sum of squares plots and table. Also from the confusion matrix we can see that the model has high accuracy (96.63%).

Summary:

Wine dataset was taken and pre-processed prior to clustering. Then we evaluated the model to find the optimal number of clusters which was found to be 3 and evaluated the model for its accuracy (96.63%).