# Topic: Dimension Reduction (PCA)

**Instructions**

Please share your answers filled inline in the word document. Submit Python code and R code files wherever applicable.

Please ensure you update all the details:

**Name: Anandakrishnan k v ;;;; Batch Id: 19042021 ;;;; Topic: Principal Component Analysis**

* 1. **Business Problem or Objective:**

Perform Principal component analysis and perform clustering using first 3 principal component scores (both Hierarchical & K-Mean clustering). Use Scree plot or elbow curve and obtain optimum number of clusters and check whether we have obtained same number of clusters with the original data

* 1. **Constraints (if any):**

**Maximize: Enhance the performance by minimizinnng the process time**

**Maximize: maximize the accuracy of the analysis results**

1. **Work on each feature of the dataset to create a data dictionary as displayed in the below image:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of the feature** | **Description** | **Type** | **Relevance** |
| Type | **Type of the liquid** | **ordinal** | **Output variable(irrelevant while do analysis)** |
| **Alcohol** | **Quantity of alcohol content** | **Quantitative** | **relevant** |
| **Malic** | **Quantity of Malic content** | **Quantitative** | **relevant** |
| **Ash** | **Quantity of Ash content** | **Quantitative** | **relevant** |
| **Alcalinity** | **Quantity of alcality content** | **Quantitative** | **relevant** |
| **magnesium** | **Quantity of magnesium content** | **Quantitative** | **relevant** |
| **phenols** | **Quantity of phenols content** | **Quantitative** | **relevant** |
| **flavanoids** | **Quantity of flavanoida content** | **Quantitative** | **relevant** |
| **Nonflavanoids** | **Quantity of nonflavanoids content** | **Quantitative** | **relevant** |
| **Proanthocyanins** | **Quantity of Proanthocyanins content** | **Quantitative** | **relevant** |
| **Color** | **Quantity of color** | **Quantitative** | **relevant** |
| **Hue** | **Quantity of Hue** | **Quantitative** | **relevant** |
| **Dilution** | **Dilution capacity** | **Quantitative** | **relevant** |
| **Proline** | **Quantity of proline** | **Quantitative** | **relevant** |

**2.1 Make a table as shown above and provide information about the features such as its Data type and its relevance to the model building, if not relevant provide reasons and provide description of the feature.**

**Using R and Python codes perform:**

1. **Data Pre-processing**

**3.1 Data Cleaning, Feature Engineering, etc.**

**R code:-**

# Loading wine data

library(readxl)

library(readr)

input <- read\_csv(file.choose())

mydata <- input

## the first column in mydata eliminating as it has Type information treating as output

data <- mydata[, -1]

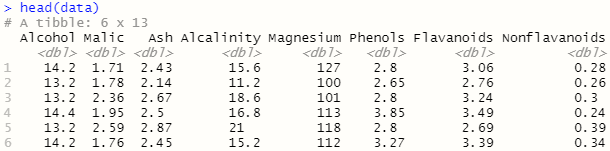
attach(data)

**## DATA CLEANING AND EDA BEGINS**

## missing data checking

sum(is.na(data)) ## no null values

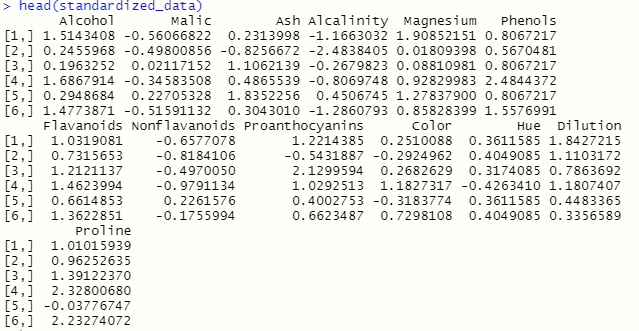
head(data)

****

### Standardize the data ###

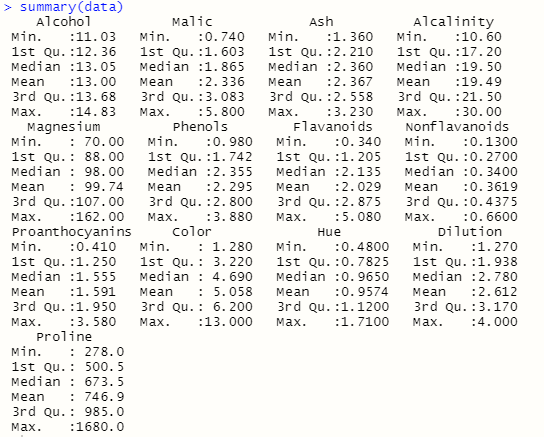
standardized\_data <- scale(data) # Excluding the university name

head(standardized\_data)

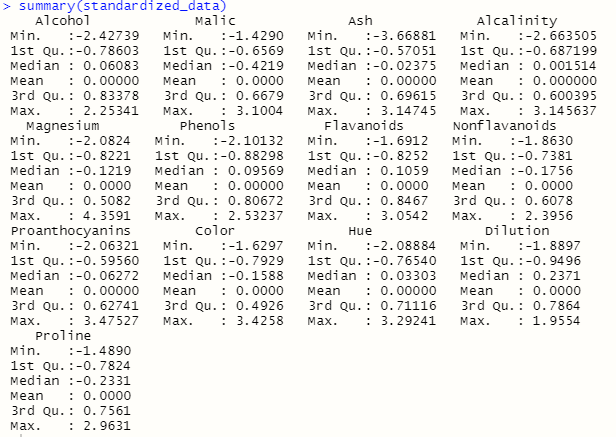
****

1. **Exploratory Data Analysis (EDA):**
   1. **Summary**

summary(data)

****

summary(standardized\_data)

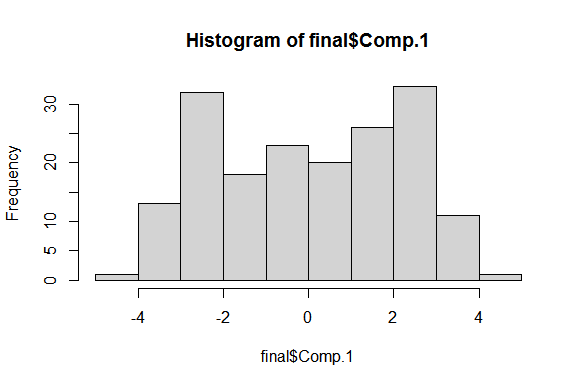
****

* 1. **Univariate analysis**

### Plots && Analysis after applying PCA####

## HISTOGRAM

hist(final$Comp.1)



## calling the univariate analysis custom function

univariate\_analysis(final$Comp.1)

variance= 4.73243697758359"

std. deviation= 2.17541650669098"

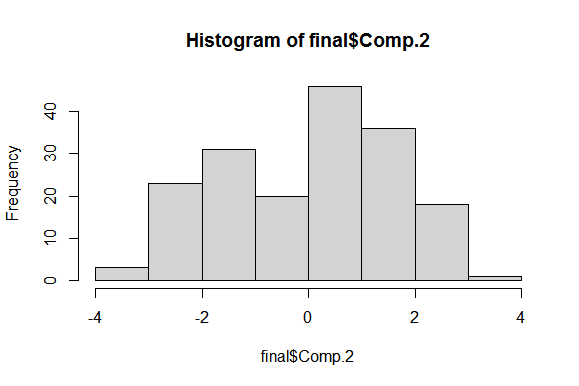
skewness= -0.0997769071949706"

excess kurtosis= -1.2734661291021"

outliers= no outliers"

number of outliers= 0"

hist(final$Comp.2)



**univariate\_analysis(final$Comp.2)**

variance= 2.51108092964512"

std. deviation= 1.58463905342672"

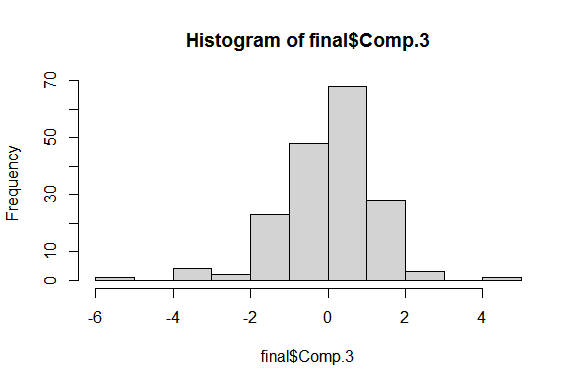
skewness= -0.182649400571563"

excess kurtosis= -0.969430697874297"

outliers= no outliers"

number of outliers= 0"

hist(final$Comp.3)



**univariate\_analysis(final$Comp.3)**

variance= 1.45424186784646"

std. deviation= 1.20591951134662"

skewness= -0.628788548953184"

excess kurtosis= 2.78742749246315"

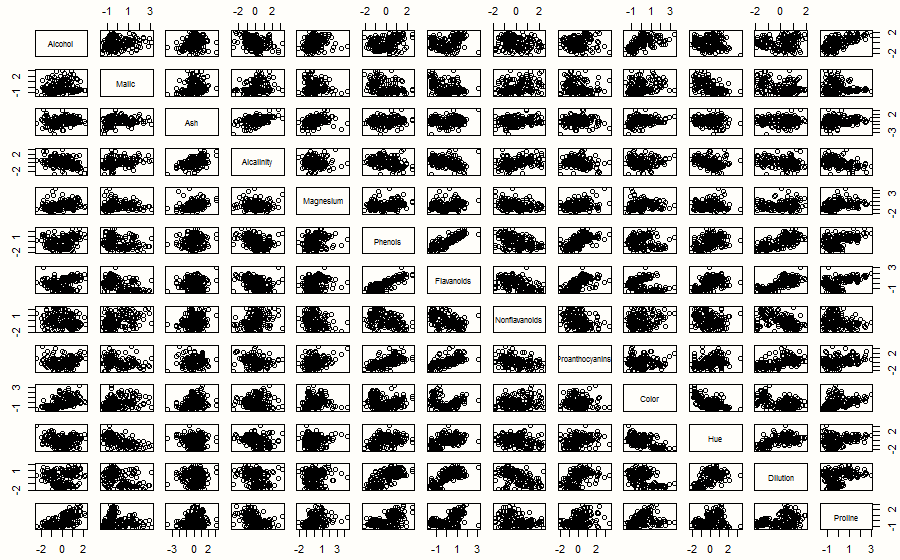
outliers= -3.82090800819484, 4.5850640071482, -3.37439396223976, -5.34538817919574, - 3.36117555538643"

number of outliers= 5"

* 1. **Bivariate analysis**

### pair plots of data ###

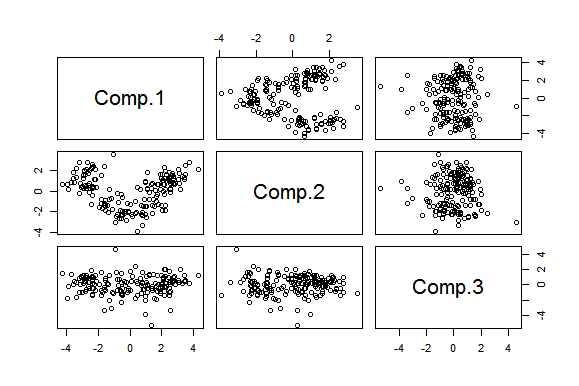
pairs(standardized\_data[,-c(1)]) ## pair plots of data



###### univariate and Bivariate analysis on PCA output data ####

### pair plots of data ###

pairs(final[,2:4]) ## pair plots of data



1. **Model Building**
   1. **Build the model on the scaled data (try multiple options)**

**Portion not yet covered**

* 1. **Perform PCA analysis and get the maximum variance between components**

**R code:-**

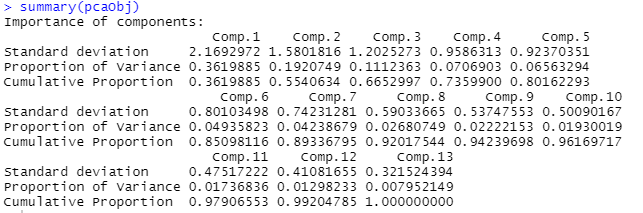
## initiating pca function

?princomp

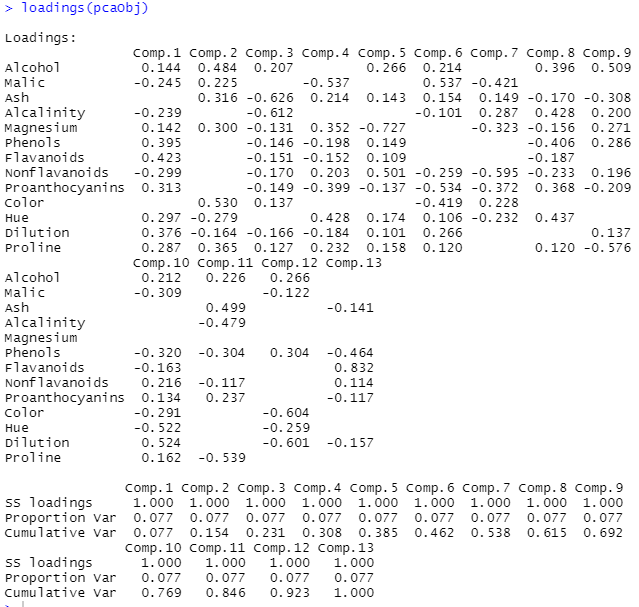
pcaObj <- princomp(data, cor = TRUE, scores = TRUE, covmat = NULL)

str(pcaObj)

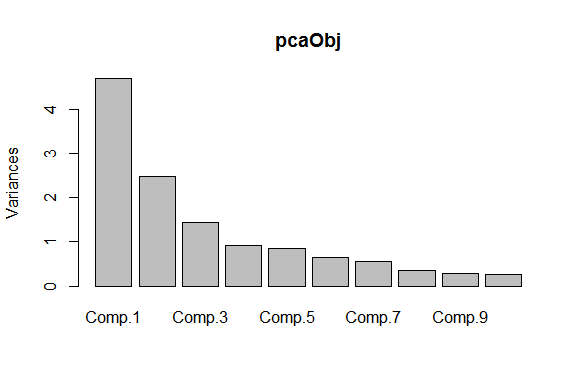
summary(pcaObj)



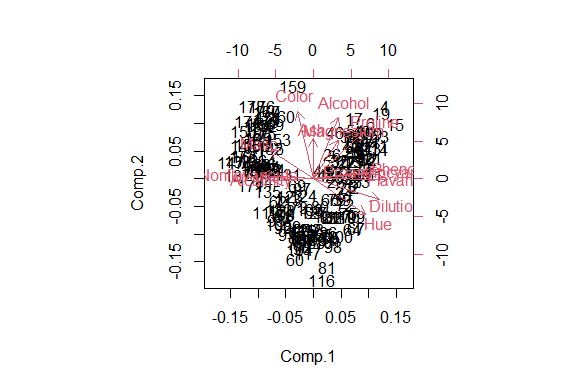
loadings(pcaObj)

****

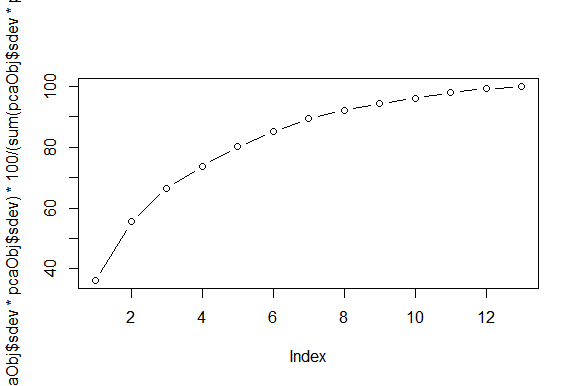
plot(pcaObj) # graph showing importance of principal components



biplot(pcaObj)



plot(cumsum(pcaObj$sdev \* pcaObj$sdev) \* 100 / (sum(pcaObj$sdev \* pcaObj$sdev)), type = "b")



# selecting first 3 pca

pcaObj$scores

pcaObj$scores[, 1:3]

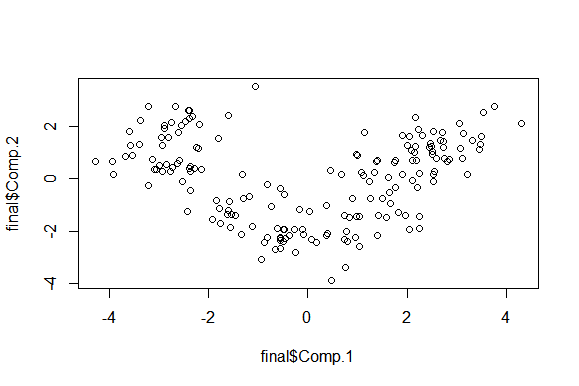
# Top 3 pca scores

final <- cbind(input[, 1], pcaObj$scores[, 1:3])

View(final)

# Scatter diagram

plot(final$Comp.1, final$Comp.2)



**### scatter between pca components showing that they are completely uncorelated with each other##**

* 1. **Perform clustering before and after applying PCA to cross the number of clusters formed.**

**R code:-**

# hierarchical clustering before applying PCA

# Distance matrix of standardized data

d <- dist(standardized\_data, method = "euclidean")

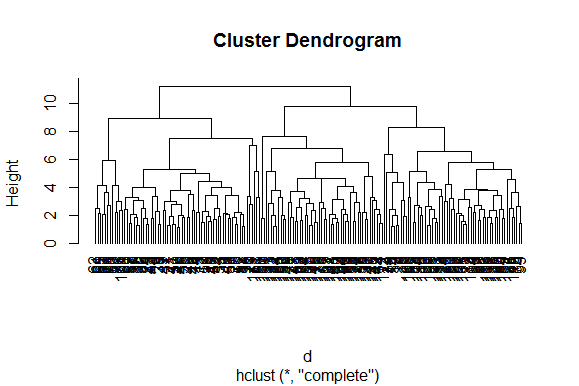
# initiate clustering on distance matrix

fit <- hclust(d, method = "complete")

# Display dendrogram

plot(fit)

plot(fit, hang = -1)

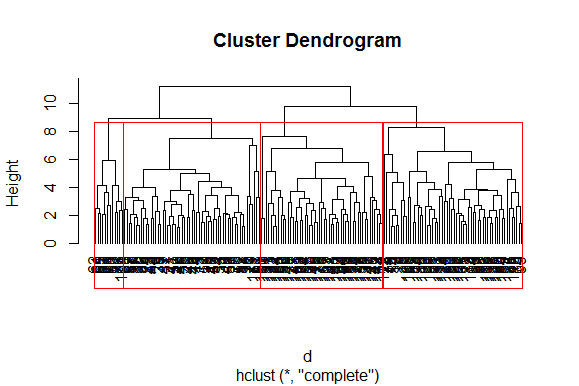


# cut the total records into 4 clusters according to distance

groups <- cutree(fit, k = 4) # Cut tree into 4 clusters

# draw red outline for each cluster

rect.hclust(fit, k = 4, border = "red")



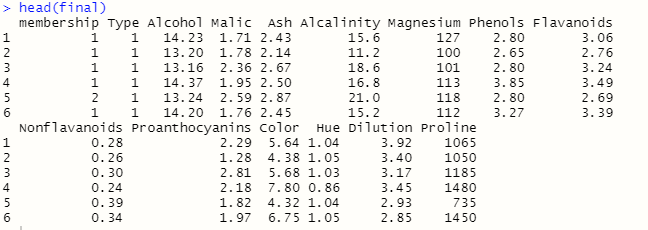
# convert the group data into matrix format

membership <- as.matrix(groups)

# cluster details merge with the original data

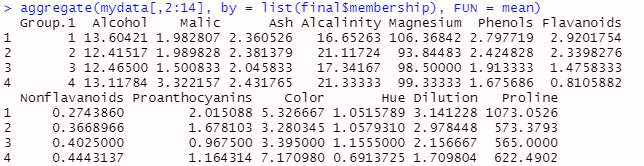
final <- data.frame(membership, mydata)

head(final)

****

# aggregate mean value of each cluster varable

aggregate(mydata[, 2:14], by = list(final$membership), FUN = mean)

****

**########## Non-hierarchical K-Means clustering ################**

install.packages("plyr")

library(plyr)

### Standardize the data ###

standardized\_data

# Elbow curve to decide the k value

twss <- NULL

for (i in 2:8) {

twss <- c(twss, kmeans(standardized\_data, centers = i)$tot.withinss) ## where tot.withinss is a column of kmean function containing TWSS value

}

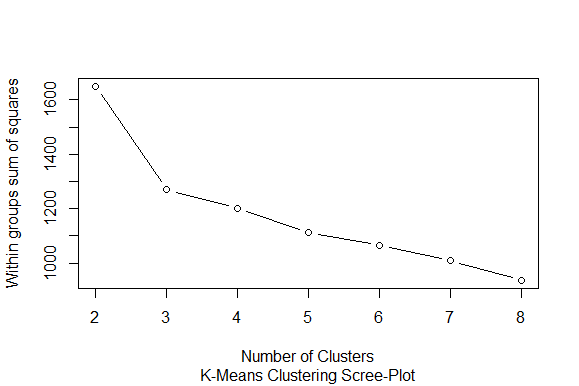
twss

**[1] 1649.4400 1270.7491 1194.1411 1098.7390 1046.2468 995.7665 964.7370**

# Look for an "elbow" in the scree plot

plot(2:8, twss, type = "b", xlab = "Number of Clusters", ylab = "Within groups sum of squares")

title(sub = "K-Means Clustering Scree-Plot")



**# 3 Cluster Solution, because there have a steep bend at k=3**

fit <- kmeans(standardized\_data, 3)

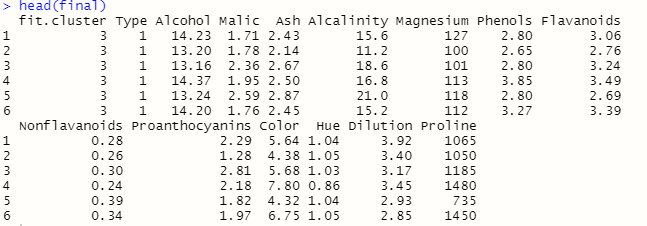
str(fit)

fit$cluster

final <- data.frame(fit$cluster, mydata) # Append cluster membership

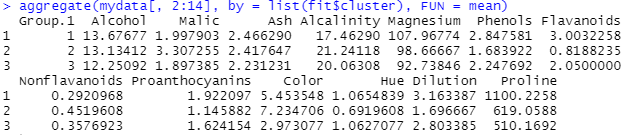
# final cluster data set

head(final)



# aggregate mean value of each cluster variable

aggregate(mydata[, 2:14], by = list(fit$cluster), FUN = mean)



########### Applying K-Means clustering after applying PCA #######

install.packages("plyr")

library(plyr)

### Standardized data set after applying PCA ###

final\_data <- final[,-1]

# Elbow curve to decide the k value

twss <- NULL

for (i in 2:8) {

twss <- c(twss, kmeans(final\_data, centers = i)$tot.withinss) ## where tot.withinss is a column of kmean function containing TWSS value

}

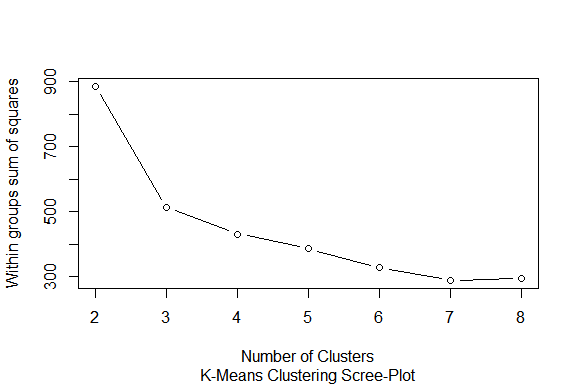
twss

**[1] 886.1611 512.9995 431.6038 386.1619 328.3210 288.9114 294.1892**

# Look for an "elbow" in the scree plot

plot(2:8, twss, type = "b", xlab = "Number of Clusters", ylab = "Within groups sum of squares")

title(sub = "K-Means Clustering Scree-Plot")



**### showing a steep bend for k=3**

# 3 Cluster Solution

fit <- kmeans(final\_data, 3)

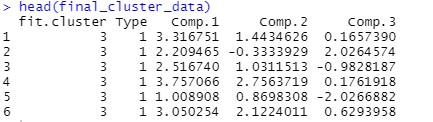
str(fit)

fit$cluster

final\_cluster\_data <- data.frame(fit$cluster, final) # Append cluster membership

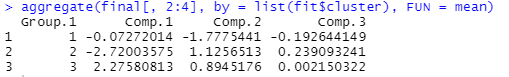
# final cluster data

head(final\_cluster\_data)

****

# aggregate mean value of variables of each cluster group

aggregate(final[, 2:4], by = list(fit$cluster), FUN = mean)

****

**Python code:-**

## R codes are given all the plots and summery, so simply pasting python code

######## hierarchical && non\_hierarchical clustering(k-mean) ######

import pandas as pd

import matplotlib.pylab as plt

wine2 = pd.read\_csv("G:\\pca\\Datasets\_PCA\\wine.csv")

wine1 = wine2

wine1.describe()

wine1.info()

### Here the data set is not a mixed data set. Since all the informative datas are numeric applying standardisation scaling here

### standardization scaling ###

#Importing the Libraries

from sklearn.preprocessing import StandardScaler, MinMaxScaler,RobustScaler

# define standard scaler

scaler = StandardScaler() # Standard Scaler or Standardization

# Standardized data frame (considering only the input variables columns of data)

df\_norm = scaler.fit\_transform(wine1.iloc[:, 1:]) #Fit to data, then transform it.

print("Standardized Scaler :\n",df\_norm)

# for creating dendrogram

from scipy.cluster.hierarchy import linkage

import scipy.cluster.hierarchy as sch

z = linkage(df\_norm, method = "complete", metric = "euclidean")

# Dendrogram

plt.figure(figsize=(15, 8));plt.title('Hierarchical Clustering Dendrogram');plt.xlabel('Index');plt.ylabel('Distance')

sch.dendrogram(z,

leaf\_rotation = 0, # rotates the x axis labels

leaf\_font\_size = 10 # font size for the x axis labels

)

plt.show()

# Now applying AgglomerativeClustering choosing 3 as clusters from the above dendrogram

from sklearn.cluster import AgglomerativeClustering

h\_complete = AgglomerativeClustering(n\_clusters = 3, linkage = 'complete', affinity = "euclidean").fit(df\_norm)

h\_complete.labels\_

# convert the from array format to pandas series object

cluster\_labels = pd.Series(h\_complete.labels\_)

wine1['clust'] = cluster\_labels # creating a new column and assigning it to new column

# convert into a dataframe format where first column indicate the cluster details

wine = wine1.iloc[:, [14,0,1,2,3,4,5,6,7,8,9,10,11,12,13]]

wine.head()

# Aggregate mean of each cluster

wine1.iloc[:, 2:].groupby(wine1.clust).mean()

# creating a csv file

wine1.to\_csv("wine.csv", encoding = "utf-8")

# save the .csv file in location

import os

os.getcwd()

##################### non\_hierarchical K- means clustering ###################

import pandas as pd

import numpy as np

import matplotlib.pylab as plt

wine1= wine2

from sklearn.cluster import KMeans

# from scipy.spatial.distance import cdist

# standardized data frame (considering the input variable column part of data)

df\_norm

###### scree plot or elbow curve ############

TWSS = [] # initiate TWSS list

k = list(range(2, 9))

for i in k:

kmeans = KMeans(n\_clusters = i)

kmeans.fit(df\_norm)

TWSS.append(kmeans.inertia\_) ## appending the value of TWSS for each k\_value

TWSS

# Scree plot

plt.plot(k, TWSS, 'ro-');plt.xlabel("No\_of\_Clusters");plt.ylabel("total\_within\_SS")

# Selecting 3 clusters from the above scree plot which is the optimum number of clusters

model = KMeans(n\_clusters = 3)

model.fit(df\_norm)

model.labels\_ # getting the labels of clusters assigned to each row

mb = pd.Series(model.labels\_) # converting numpy array into pandas series object

wine1['clust'] = mb # creating a new column and assigning it to new column

wine1.head()

wine = wine1.iloc[:,[14,0,1,2,3,4,5,6,7,8,9,10,11,12,13]]

wine.head()

# Aggregate mean of each cluster

wine1.iloc[:, 2:8].groupby(wine1.clust).mean()

Univ.to\_csv("Kmeans\_university.csv", encoding = "utf-8")

import os

os.getcwd()

##################### K-mean clustering after applying PCA #####################

import pandas as pd

import numpy as np

from sklearn.decomposition import PCA

import matplotlib.pyplot as plt

from sklearn.preprocessing import scale

wine1= wine2

# standardized data set

df\_norm

pca = PCA(n\_components = 3)

pca\_values = pca.fit\_transform(df\_norm)

# The amount of variance that each PCA explains is

var = pca.explained\_variance\_ratio\_

var

pca.components\_

pca.components\_[0]

# Cumulative variance

var1 = np.cumsum(np.round(var, decimals = 4) \* 100)

var1

# Variance plot for PCA components obtained

plt.plot(var1, color = "red")

# PCA scores

pca\_values

pca\_data = pd.DataFrame(pca\_values)

pca\_data.columns = "comp0", "comp1", "comp2"

final = pd.concat([wine1.Type, pca\_data.iloc[:, 0:3]], axis = 1)

# Scatter diagram

import matplotlib.pylab as plt

plt.scatter(x = final.comp0, y = final.comp1)

## K-means clustering ##

import pandas as pd

import numpy as np

import matplotlib.pylab as plt

wine1= wine2

from sklearn.cluster import KMeans

# from scipy.spatial.distance import cdist

# standardized PCA data frame (considering the input variable column part of data)

final

###### scree plot or elbow curve ############

TWSS = [] # initiate TWSS list

k = list(range(2, 9))

for i in k:

kmeans = KMeans(n\_clusters = i)

kmeans.fit(final)

TWSS.append(kmeans.inertia\_) ## appending the value of TWSS for each k\_value

TWSS

# Scree plot

plt.plot(k, TWSS, 'ro-');plt.xlabel("No\_of\_Clusters");plt.ylabel("total\_within\_SS")

# Selecting 3 clusters from the above scree plot which is the optimum number of clusters

model = KMeans(n\_clusters = 3)

model.fit(final)

model.labels\_ # getting the labels of clusters assigned to each row

mb = pd.Series(model.labels\_) # converting numpy array into pandas series object

final['clust'] = mb # creating a new column and assigning it to new column

final.head()

final1 = final.iloc[:,[4,0,1,2,3]]

final1.head()

# Aggregate mean of each cluster

final.iloc[:, 1:4].groupby(final.clust).mean()

final1.to\_csv("Kmeans\_wine.csv", encoding = "utf-8")

import os

os.getcwd()

**Summary:-**

Hierarchical & Non\_hierarchical Clustering is done on both PCA applied and not applied datas.

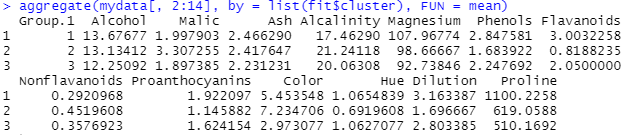
In both the cases the scree plot(elbow) curve shown the first steep bend for K=3. So in both the cases

optimum number of cluster groups is 3.

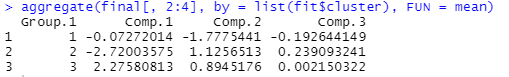
1. **Share the benefits/impact of the solution - how or in what way the business (client) gets benefit from the solution provided.**

**Ans:-**

**Aggregate mean value of clusters feature before applying PCA**



**Aggregate mean value of clusters feature after applying PCA**

****

**SUMMARY:-**

1. **The records, here liquids comes under “Group 1” has –ve values for all the 3 PCA components and it is lowest all groups for Comp.2 & Comp.3 PCA component**
2. **Except “Comp.1” “Group 2” having the highest values for all other PCA components. So that type of liquids will fall under those groups.**
3. **Group 3 is those group where all PCA components values comes in between that of other Groups except Comp.1 component.**

**In both the cases optimum number of cluster group is 3. In first case where data set clustered before applying PCA having more number of feature datas to consider. So we have to consider more input or independant variables while creating regression equation therby getting an accurate ouput data. Considering more variables will make the system complicated and time lagging. In the above itself while considering all the features and its up down values in different records make difficult to implement a accurate prediction.**

**After establishing PCA analysis the total independent input variable we have to consider for making an accurate prediction is only 3. Its will helps to enhance the over all performance of the system theby giving an fast output.**

**Problem Statement: -**

A Pharmaceutical drug manufacturing company is studying on a new medicine to treat Heart diseases, it has gathered data from its secondary sources, and it would like you to provide high level analytical insights on the data, its aim is to segregate patients depending on their age group and other factors as given in the data, perform PCA and Clustering Machine learning Algorithm on the dataset given, and check if the clusters formed before and after PCA are same and provide a brief report on your model. You can also explore more on ways to improve your model.

**Objective:**

Aim is to segregate patients depending on their age group and other factors as given in the data

**Constraints (if any):**

Maximize: Enhance the performance by minimizinnng the process time

Maximize: maximize the accuracy of the analysis results

**Work on each feature of the dataset to create a data dictionary as displayed in the below image:**



**R code:-**

# Loading heart disease data

library(readxl)

library(readr)

input <- read\_csv(file.choose())

mydata <- input

## the 14th column in mydata eliminating as it has target information treating as output

data <- mydata[, -14]

attach(data)

## DATA CLEANING AND EDA BEGINS

## missing data checking

sum(is.na(data)) ## no null values

head(data)

str(data)

summary(data)

### Standardize the data ###

standardized\_data <- scale(data) # Excluding the university name

head(standardized\_data)

summary(standardized\_data)

### pair plots of data ###

pairs(standardized\_data) ## pair plots of data

####### univariate\_analysis ######

univariate\_analysis <- function(variable)

{

library(readr)

p <- variable

b <- list("variance=", "std. deviation=")

c <- list("skewness=", "excess kurtosis=", "outliers=", "number of outliers=")

#second moment business decision

b[[1]][2] <- var(p)

b[[2]][2] <- sd(p)

#install e1071 package

library(e1071)

#third & fourth moment business decision

bx <- boxplot(p)

##skewness###

c1 <- skewness(p)

s <- c(1:3)

s[1]= mean(p)>median(p)

s[2]= mean(p)<median(p)

s[3]= mean(p)==median(p)

sk <- c("+ve skewness: because mean > median", "-ve skewness: because mean<median ", "normal or zero: mean = median ")

for(i in 1:3)

{

if(s[i]==TRUE)

{

actual\_skew <- sk[i]

}

}

##kurtosis

c2 <- kurtosis(p)

k <- c(1:3)

k[1]= c2>0

k[2]= c2<0

k[3]= c2==0

ku <- c("leptokurtic", "platykurtic", "mesokurtic")

for(i in 1:3)

{

if(k[i]==TRUE)

{

kurt <- ku[i]

}

}

outliers <- bx$out

##outliers

if(length(bx$out)==0)

{

c[[3]][2] <- "no outliers"

c[[4]][2] <- 0

}else

{

c3 <- bx$out

c4 <- length(bx$out)+1

for(i in 2:c4)

{

c[[3]][i] <- bx$out[i-1]

}

c[[4]][2]=c4-1

}

c[[1]][2]=c1

c[[2]][2]=c2

### conclusion ###

##second moment business decision

print("second moment business decisions are")

print(b)

##third & fourth moment business decisions & conclusions

print("third & fourth moment business decisions are")

print(c)

print("actual skewness is")

print(actual\_skew)

print("type of kurtosis is")

print(kurt)

}

########## Non-hierarchical K-Means clustering ################

install.packages("plyr")

library(plyr)

### Standardized data set ###

standardized\_data

# Elbow curve to decide the k value

twss <- NULL

for (i in 2:8) {

twss <- c(twss, kmeans(standardized\_data, centers = i)$tot.withinss) ## where tot.withinss is a column of kmean function containing TWSS value

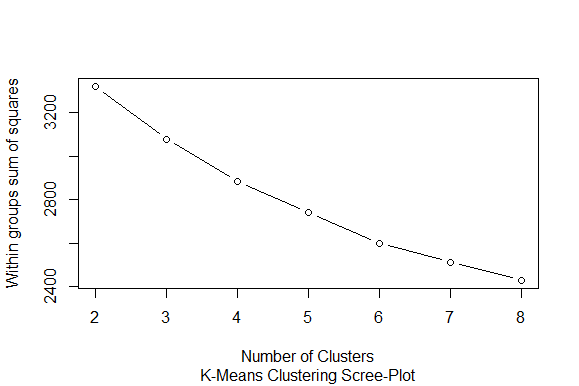
}

twss

**[1] 3319.745 3078.195 2881.322 2741.139 2599.554 2512.317 2428.532**

# Look for an "elbow" in the scree plot

plot(2:8, twss, type = "b", xlab = "Number of Clusters", ylab = "Within groups sum of squares")

title(sub = "K-Means Clustering Scree-Plot")

## having a steep bend for k=3

# 3 Cluster Solution

fit <- kmeans(standardized\_data, 3)

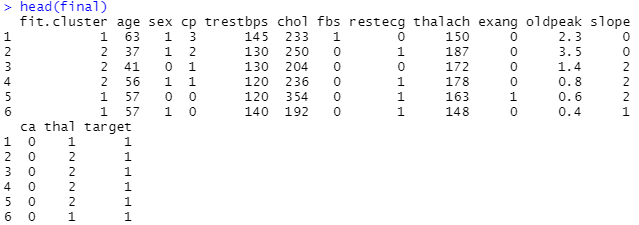
str(fit)

fit$cluster

final <- data.frame(fit$cluster, mydata) # Append cluster membership

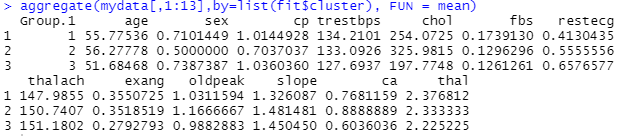
# final cluster data

head(final)

****

# aggregate mean value of variables of each cluster group

aggregate(mydata[, 2:14], by = list(fit$cluster), FUN = mean)

****

######################### Applying pca ###############################

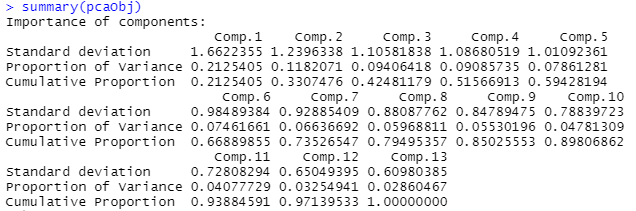
## initiating pca function

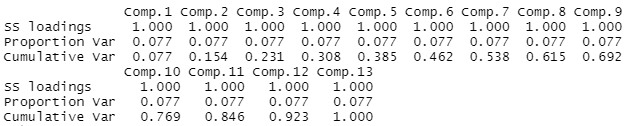
?princomp

pcaObj <- princomp(data, cor = TRUE, scores = TRUE, covmat = NULL)

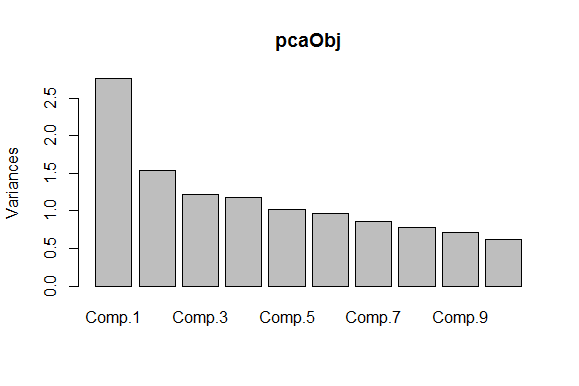
str(pcaObj)

summary(pcaObj)

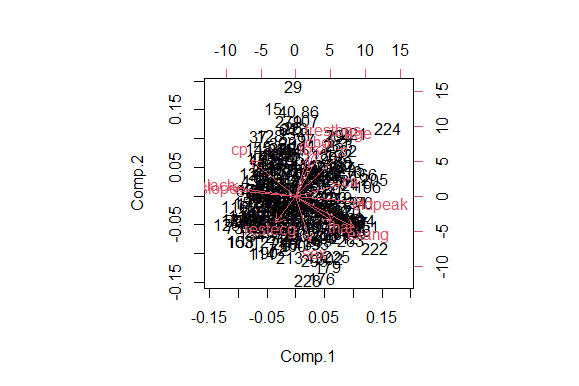


loadings(pcaObj)

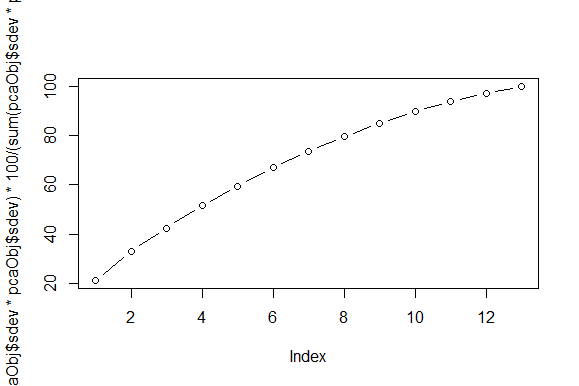
plot(pcaObj) # graph showing importance of principal components



biplot(pcaObj)



plot(cumsum(pcaObj$sdev \* pcaObj$sdev) \* 100 / (sum(pcaObj$sdev \* pcaObj$sdev)), type = "b")



# selecting first 3 pca

pcaObj$scores

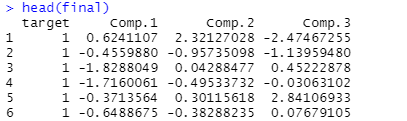
pcaObj$scores[, 1:3]

# Top 3 pca scores

final <- cbind(input[, 1], pcaObj$scores[, 1:3])

view(final)

head(final)



########### Applying K-Means clustering after applying PCA #######

install.packages("plyr")

library(plyr)

### Standardized data set after applying PCA ###

final\_data <- final[,-1]

# Elbow curve to decide the k value

twss <- NULL

for (i in 2:8) {

twss <- c(twss, kmeans(final\_data, centers = i)$tot.withinss) ## where tot.withinss is a column of kmean function containing TWSS value

}

Twss

**[1] 1071.2245 850.7015 667.8584 569.2491 505.6671 440.7999 407.4822**

# Look for an "elbow" in the scree plot

plot(2:8, twss, type = "b", xlab = "Number of Clusters", ylab = "Within groups sum of squares")

title(sub = "K-Means Clustering Scree-Plot")

# 3 Cluster Solution

fit <- kmeans(final\_data, 3)

str(fit)

fit$cluster

final\_cluster\_data <- data.frame(fit$cluster, final) # Append cluster membership

# # final cluster data

# head(final\_cluster\_data)

# G:\pca\clus.png

# # aggregate mean value of variables of each cluster group

# aggregate(final[, 2:4], by = list(fit$cluster), FUN = mean)

# G:\pca\clus.png

# ###### univariate and Bivariate analysis on PCA output data ####

# ### pair plots of data ###

# pairs(final[,2:4]) ## pair plots of data

# 

# ### PLOTS ####

# ## HISTOGRAM

# hist(final$Comp.1)

# 

# hist(final$Comp.2)

# 

# hist(final$Comp.3)

# 

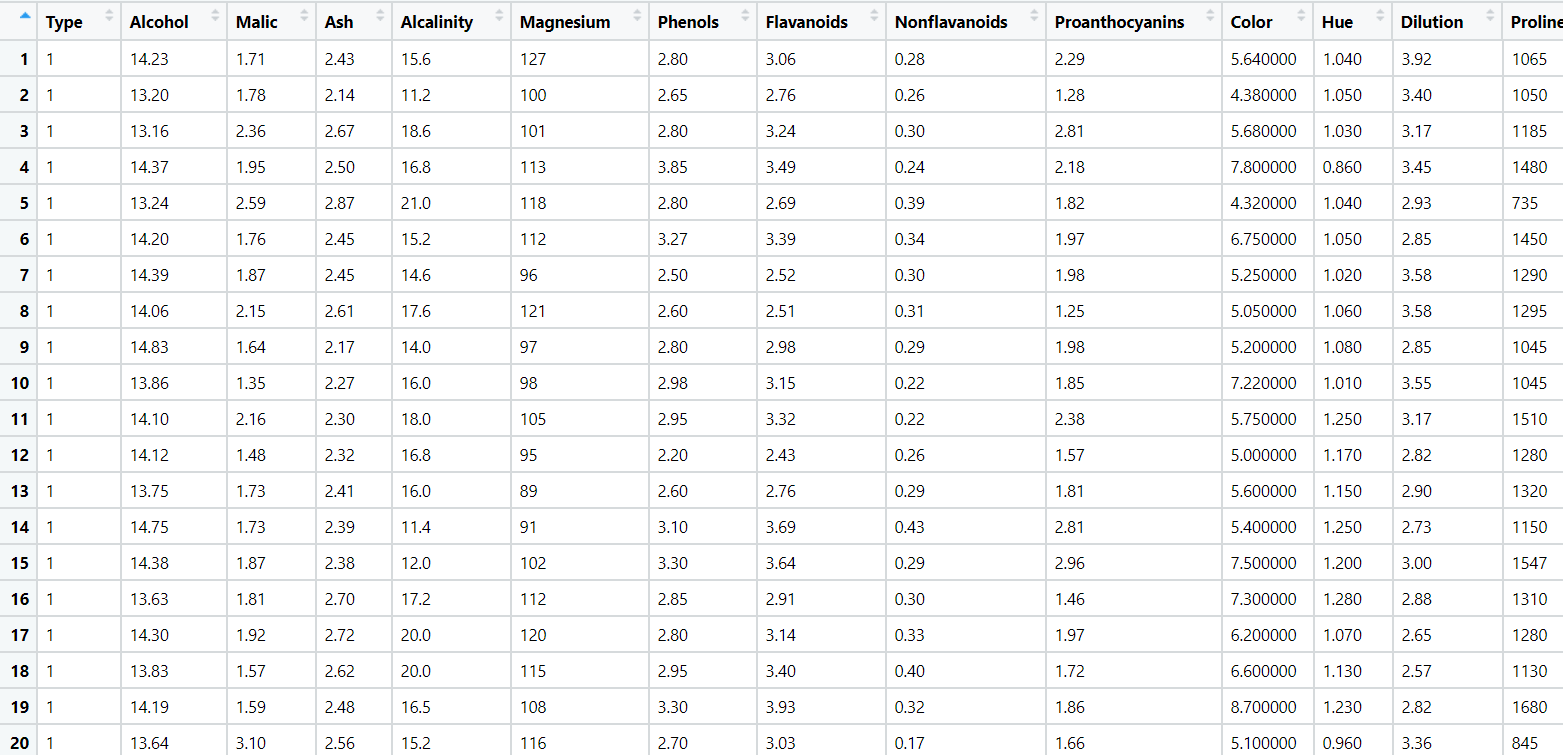
# Note:

The assignment should be submitted in the following format:

* R code
* Python code
* Code Modularization should be maintained
* Documentation of the model building (elaborating on steps mentioned above)

**Problem Statement: -**

Perform Principal component analysis and perform clustering using first 3 principal component scores (both Hierarchical & K-Mean clustering). Use Scree plot or elbow curve and obtain optimum number of clusters and check whether we have obtained same number of clusters with the original data



**Problem Statement: -**

A Pharmaceutical drug manufacturing company is studying on a new medicine to treat Heart diseases, it has gathered data from its secondary sources, and it would like you to provide high level analytical insights on the data, its aim is to segregate patients depending on their age group and other factors as given in the data, perform PCA and Clustering Machine learning Algorithm on the dataset given, and check if the clusters formed before and after PCA are same and provide a brief report on your model. You can also explore more on ways to improve your model.

Note: - This is just a snap shot of the data, the datasets can be downloaded from Aispry LMS in the Hands on Material section.

A screenshot of a cell phone

Description automatically generated