Homework 4

Manikanta Kalyan Gokavarapu

2023-05-19

Loading Packages:

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
library(reshape2)
library(kohonen)
```

```
## Warning: package 'kohonen' was built under R version 4.2.3
```

```
library(ggbiplot)
```

```
## Loading required package: plyr
```

```
## Loading required package: scales
```

```
## Loading required package: grid
```

```
library(cluster)
```

```
## Warning: package 'cluster' was built under R version 4.2.3
```

```
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.2.3
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library('glasso')
library('graph')
```

```
## Loading required package: BiocGenerics
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, setdiff, sort,
##     table, tapply, union, unique, unsplit, which.max, which.min
```

```
##
## Attaching package: 'graph'

## The following object is masked from 'package:plyr':
##
##     join
```

```
library('recommenderlab')
```

```
## Loading required package: Matrix

## Loading required package: arules

## Warning: package 'arules' was built under R version 4.2.3

##
## Attaching package: 'arules'

## The following objects are masked from 'package:base':
##
##     abbreviate, write

## Loading required package: proxy

##
## Attaching package: 'proxy'

## The following object is masked from 'package:Matrix':
##
##     as.matrix

## The following objects are masked from 'package:stats':
##
##     as.dist, dist

## The following object is masked from 'package:base':
##
##     as.matrix

## Registered S3 methods overwritten by 'registry':
##   method               from
##   print.registry_field proxy
##   print.registry_entry proxy

##
## Attaching package: 'recommenderlab'

## The following object is masked from 'package:BiocGenerics':
##
##     normalize
```

```
library("cluster")
library('kohonen')
library('ggplot2')
library('corrplot')
```

```
## corrplot 0.92 loaded
```

```
library('igraph')
```

```
## Warning: package 'igraph' was built under R version 4.2.3
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:recommenderlab':
##
##     normalize, similarity

## The following object is masked from 'package:arules':
##
##     union

## The following objects are masked from 'package:graph':
##
##     degree, edges, intersection, union

## The following objects are masked from 'package:BiocGenerics':
##
##     normalize, path, union

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

```
library('corrplot')
library("multtest")
```

```
## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
library("fpc")
```

```
## Warning: package 'fpc' was built under R version 4.2.3
```

```
library("bootcluster")
```

```
## Warning: package 'bootcluster' was built under R version 4.2.3

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library("fossil")
```

```
## Warning: package 'fossil' was built under R version 4.2.3

## Loading required package: sp

## Warning: package 'sp' was built under R version 4.2.3

## Loading required package: maps

## Warning: package 'maps' was built under R version 4.2.3
```

```
##
## Attaching package: 'maps'

## The following object is masked from 'package:cluster':
##
##     votes.repub

## The following object is masked from 'package:plyr':
##
##     ozone

## The following object is masked from 'package:kohonen':
##
##     map

## Loading required package: shapefiles

## Loading required package: foreign

##
## Attaching package: 'shapefiles'

## The following objects are masked from 'package:foreign':
##
##     read.dbf, write.dbf
```

```
library('Rgraphviz')
library('ggm')
```

```
## Warning: package 'ggm' was built under R version 4.2.3

##
## Attaching package: 'ggm'

## The following object is masked from 'package:igraph':
##
##     pa
```

```
library('bnlearn')
```

```
## Warning: package 'bnlearn' was built under R version 4.2.3

##
## Attaching package: 'bnlearn'

## The following objects are masked from 'package:igraph':
##
##     as.igraph, compare, subgraph

## The following object is masked from 'package:arules':
##
##     discretize
```

```
library('glasso')
library('graph')
library('igraph')
library(multtest)
library('fpc')
library('bootcluster')
library('fossil')
library ('ggm')
```

```r
library('bnlearn')
library('gRain')
```

```
## Warning: package 'gRain' was built under R version 4.2.3
```

```
## Loading required package: gRbase
```

```
## Warning: package 'gRbase' was built under R version 4.2.3
```

```
##
## Attaching package: 'gRbase'
```

```
## The following objects are masked from 'package:bnlearn':
##
##     ancestors, children, parents
```

```
## The following object is masked from 'package:Biobase':
##
##     description<-
```

```
## The following objects are masked from 'package:igraph':
##
##     edges, is_dag, topo_sort
```

```
## The following object is masked from 'package:scales':
##
##     ordinal
```

```r
#library('gRim')
library('gRbase')
#library(devtools)
library('huge')
```

```
## Warning: package 'huge' was built under R version 4.2.3
```

```r
#library('glmpath')
```

Question 1: Consider the "cad1" data set in the package gRbase. There are 236 observations on fourteen variables from the Danish Heart Clinic.

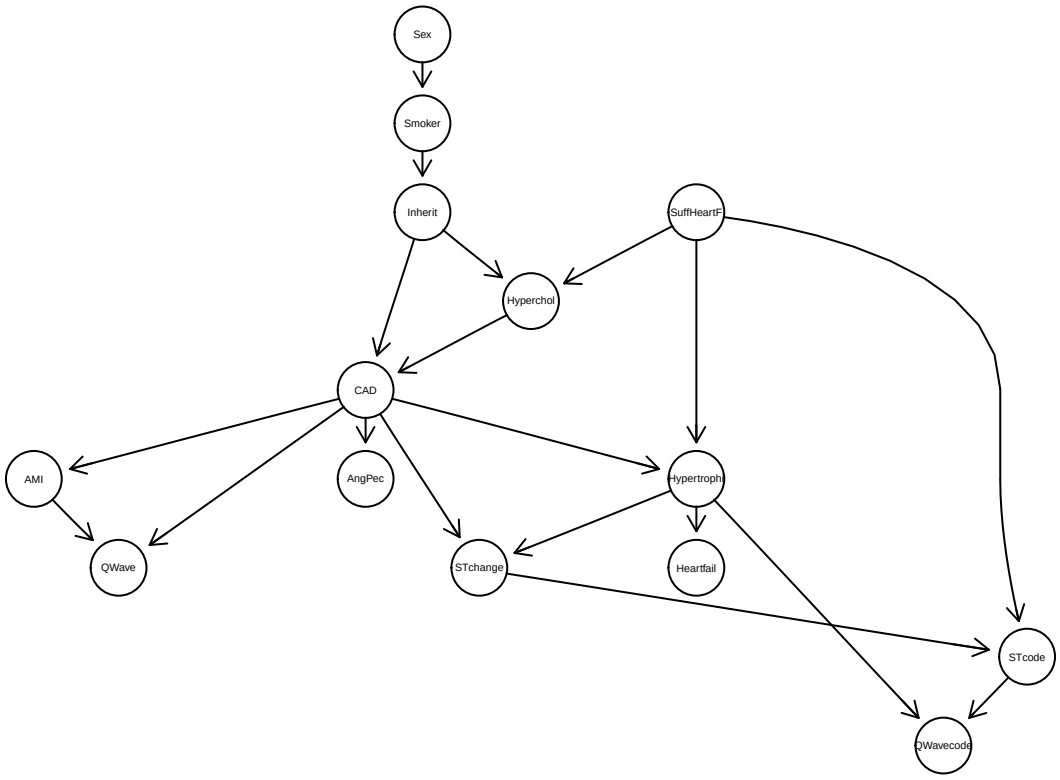
a) Use a structural learning algorithm to infer a Bayesian Network for the cad1 data. Be sure to consider the nature of the variables in your dataset, and your knowledge about variable ordering.

```
data(cad1)
cad_copy = cad1
head(cad_copy)
```

Answer:

```
##      Sex   AngPec        AMI QWave QWavecode    STcode STchange SuffHeartF
## 1   Male     None NotCertain    No    Usable    Usable       No          No
## 2   Male Atypical NotCertain    No    Usable    Usable       No          No
## 3 Female     None   Definite    No    Usable    Usable       No          No
## 4   Male     None NotCertain    No    Usable Nonusable       No          No
## 5   Male     None NotCertain    No    Usable Nonusable       No          No
## 6   Male     None NotCertain    No    Usable Nonusable       No          No
##   Hypertrophi Hyperchol Smoker Inherit Heartfail CAD
## 1          No        No     No      No        No  No
## 2          No        No     No      No        No  No
## 3          No        No     No      No        No  No
## 4          No        No     No      No        No  No
## 5          No        No     No      No        No  No
## 6          No        No     No      No        No  No
```

```
 block <- c(1, 3, 3, 4, 4, 4, 4, 1, 2, 1, 1, 1, 3, 2) #assign variables a block
blM <- matrix(0, nrow = 14, ncol = 14)
rownames(blM) <- names(cad_copy)
colnames(blM) <- names(cad_copy)
# fill in the illegal edges
for (b in 2:4){
blM[block == b, block < b] <- 1
}
blackL <- data.frame(get.edgelist(as(blM, "igraph")))
names(blackL) <- c("from", "to")
#Refit the network under the new constraints
cad.bn2 <- hc(cad1, blacklist = blackL)
net.constr <- as(amat(cad.bn2), "graphNEL")
plot(net.constr )
```



b) Use a structural learning algorithm to infer a Bayesian Network for the cad1 data. Be sure to consider the nature of the variables in your dataset, and your knowledge about variable ordering.

```
### Fit the data to Bayesian network
bnfit = bn.fit(cad.bn2 , cad1)
```

```
### Here are d-seperations  for D-Seperated variables
dsep(cad.bn2 , 'Sex' , 'SuffHeartF' , 'Smoker')
```

Answer

```
## [1] TRUE
```

```
dsep(cad.bn2 , 'Sex' , 'SuffHeartF' , 'Smoker')
```

```
## [1] TRUE
```

c) Use a structural learning algorithm to infer a Bayesian Network for the cad1 data. Be sure to consider the nature of the variables in your dataset, and your knowledge about variable ordering.

```
dsep(cad.bn2 , 'Sex' , 'SuffHeartF' , 'Smoker')
```

Answer:

```
## [1] TRUE
```

d) Use a structural learning algorithm to infer a Bayesian Network for the cad1 data. Be sure to consider the nature of the variables in your dataset, and your knowledge about variable ordering.

```
cpquery(bnfit, event = (CAD == "Yes"), evidence = (QWave == "Yes"&AMI =='Definite'))
```

Answer

```
## [1] 0.8462396
```

Question 2: Consider the wine quality data.

Suppose that for a particular data set, we perform hierarchical clustering using single linkage and using complete linkage. We obtain two dendrograms.

(a) Perform exploratory data analysis on the data. Summarize the data quality and characteristics. Discuss any outliers and associations.

```
# Load the red wine and white wine data sets
red_wine <- read.csv("D:/University at Buffalo CSE/Spring Semester 2023/STA 546 DataM II/R/Assignment 4
white_wine <- read.csv("D:/University at Buffalo CSE/Spring Semester 2023/STA 546 DataM II/R/Assignment
dim(red_wine)
```

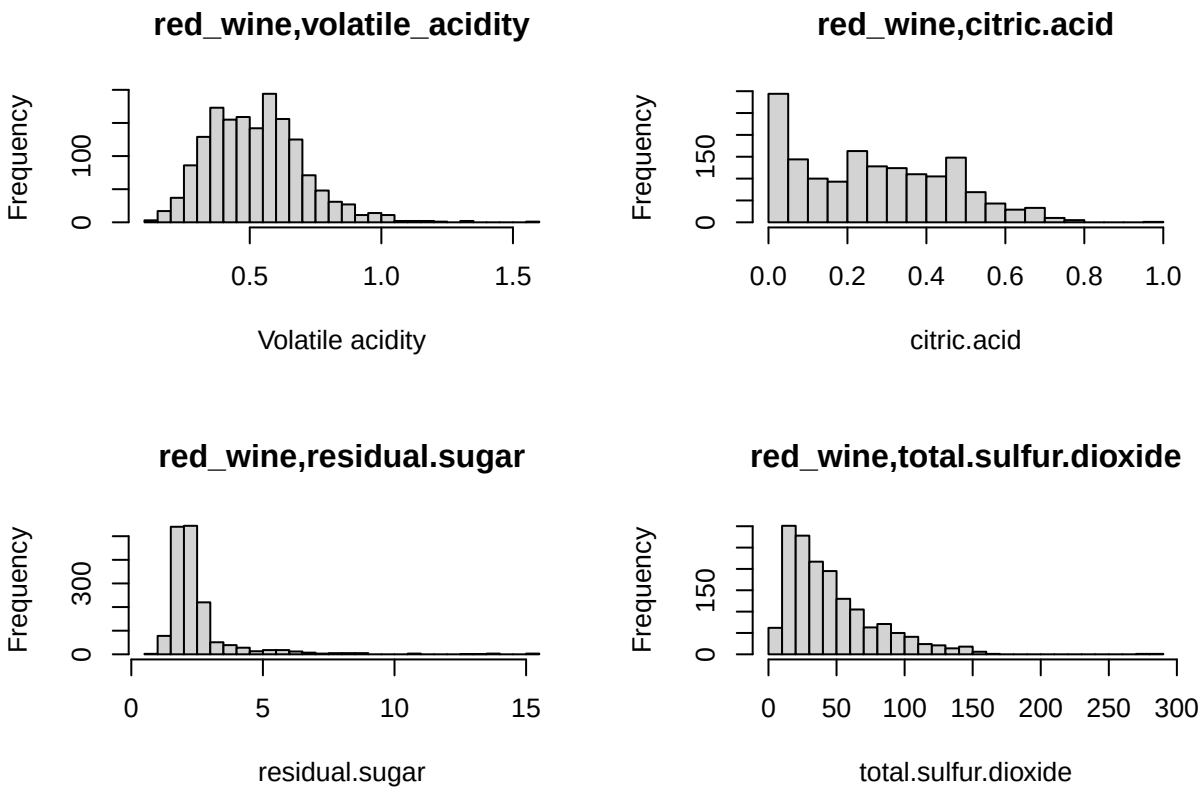
Answer:

```
## [1] 1599   12
```

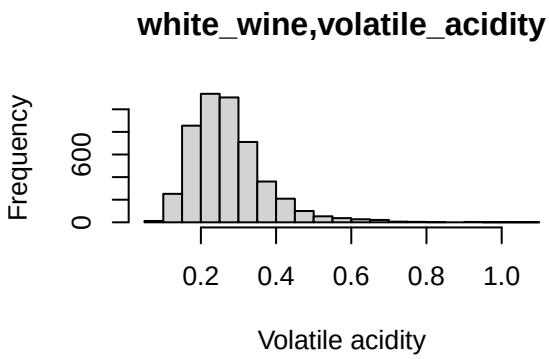
```
dim(white_wine)
```

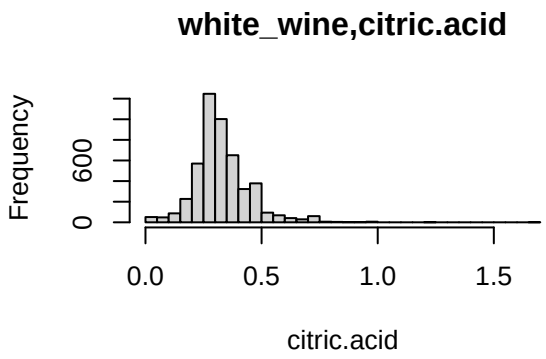
```
## [1] 4898   12
```

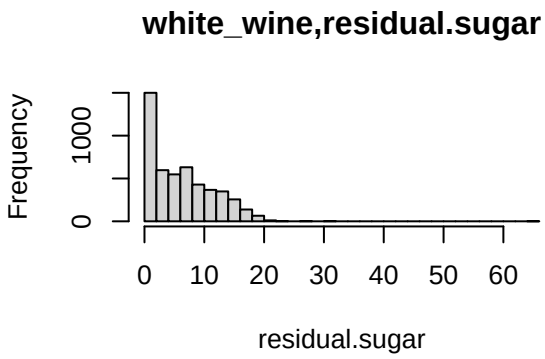
```
##EDA
# Create a histogram of red wine w.r.t different attributes.
par(mfrow=c(2,2))
hist(red_wine[,2], breaks = 25, main = "red_wine,volatile_acidity", xlab = "Volatile acidity")
hist(red_wine[,3], breaks = 25, main = "red_wine,citric.acid", xlab = "citric.acid")
hist(red_wine[,4], breaks = 25, main = "red_wine,residual.sugar", xlab = "residual.sugar")
hist(red_wine[,7], breaks = 25, main = "red_wine,total.sulfur.dioxide", xlab = "total.sulfur.dioxide")
```

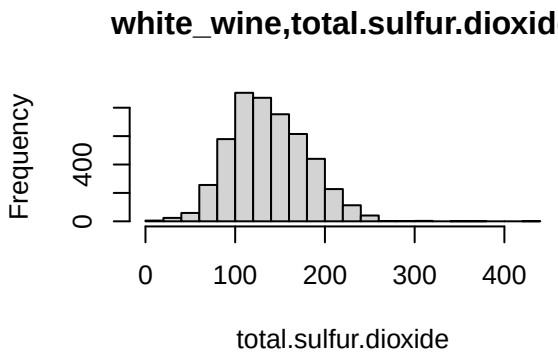



```
#Create histogram of white whine w.r.t different attributes
par(mfrow=c(2,2))
hist(white_wine[,2], breaks = 25, main = "white_wine,volatile_acidity", xlab = "Volatile acidity")
hist(white_wine[,3], breaks = 25, main = "white_wine,citric.acid", xlab = "citric.acid")
hist(white_wine[,4], breaks = 25, main = "white_wine,residual.sugar", xlab = "residual.sugar")
hist(white_wine[,7], breaks = 25, main = "white_wine,total.sulfur.dioxide", xlab = "total.sulfur.dioxid
```

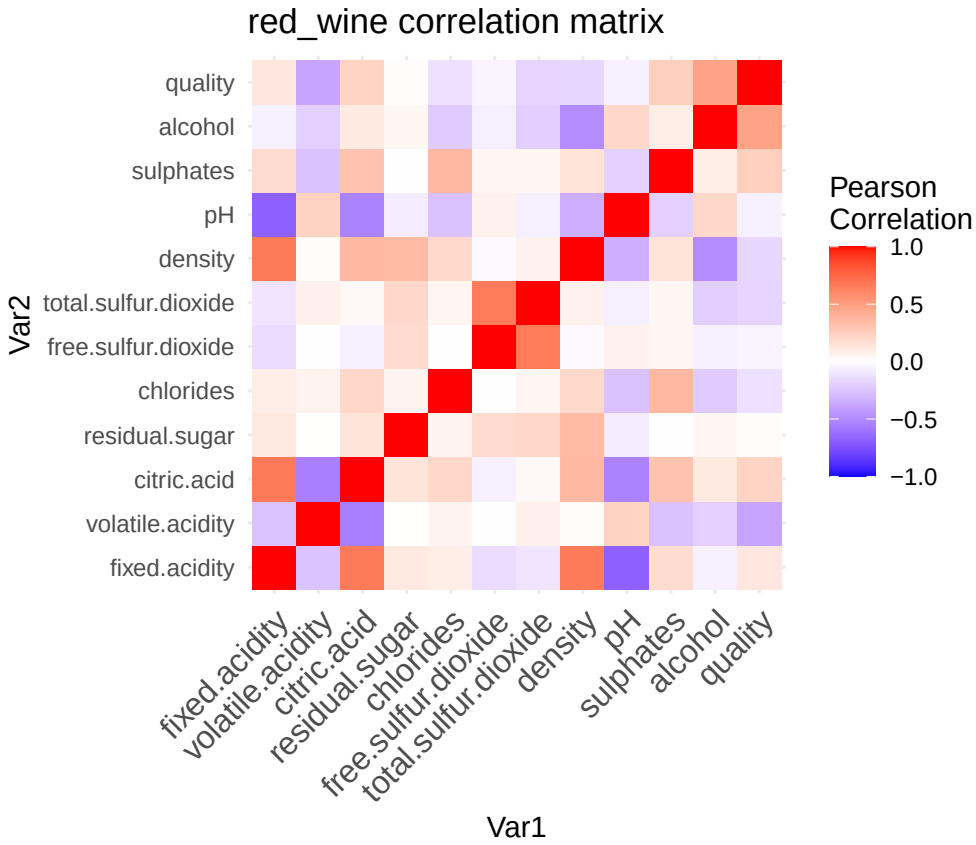




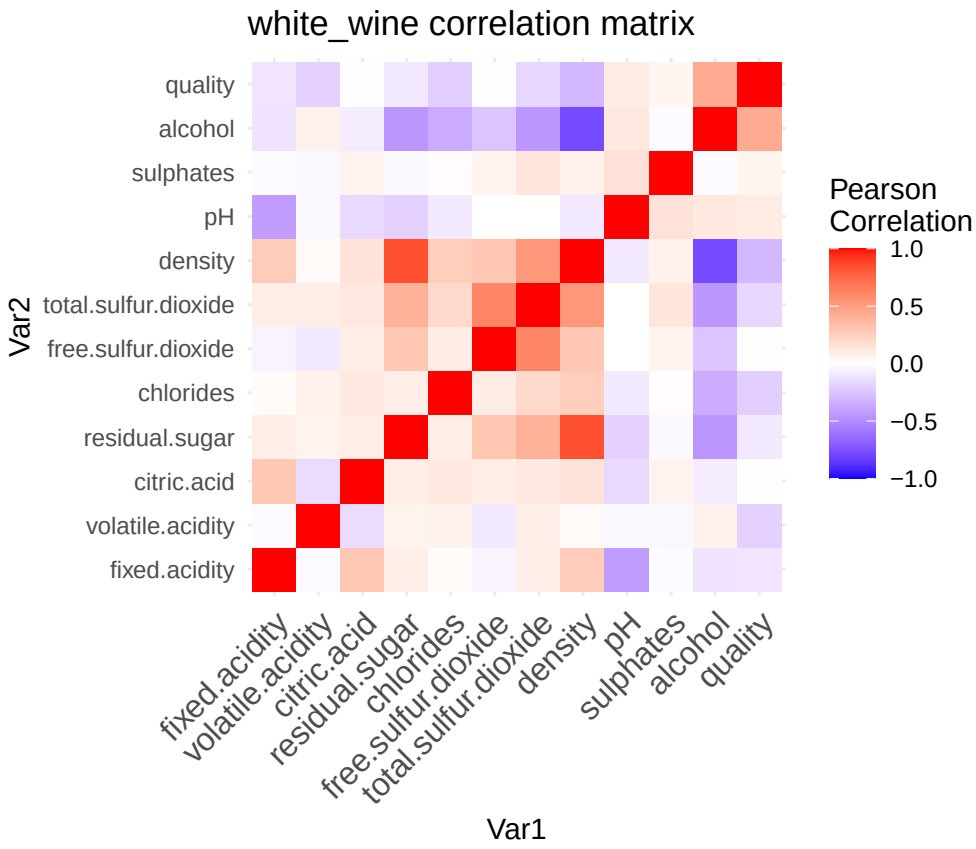




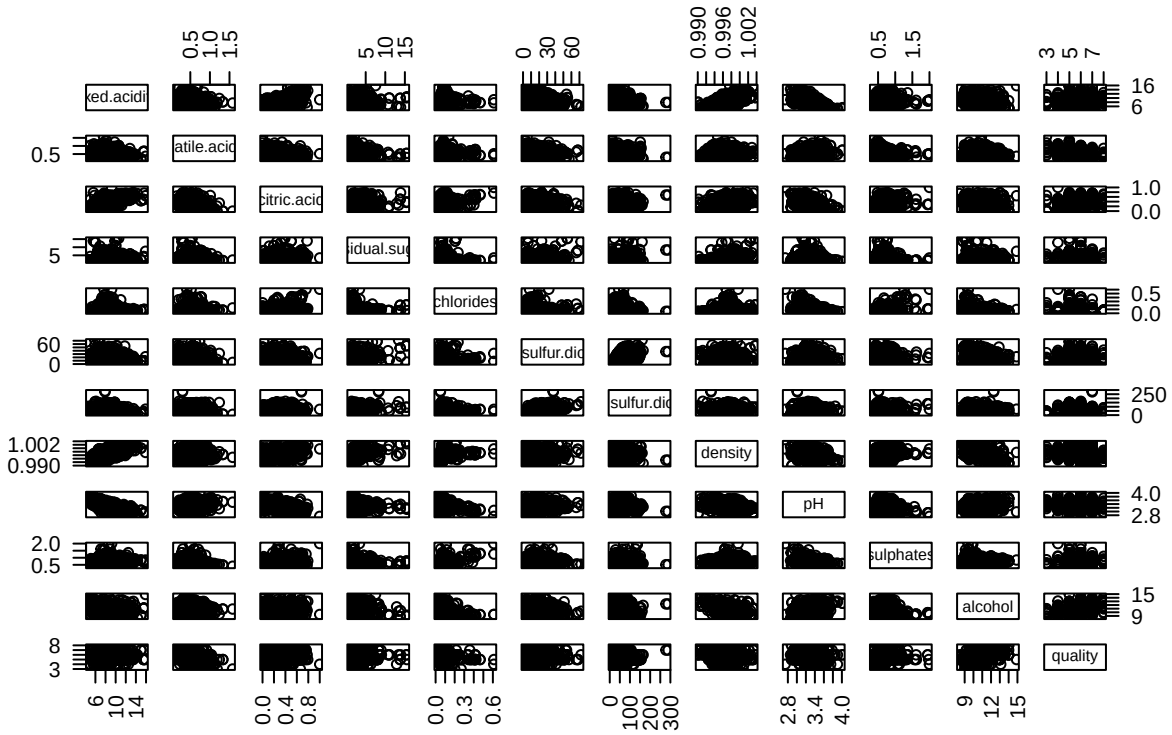

```r
# Calculate the correlation  matrix of red wine
corr_matrix <- cor(red_wine)
melted_corr_matrix <- melt(corr_matrix)
ggplot(data = melted_corr_matrix, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", mid = "white", high = "red",
                       midpoint = 0, limit = c(-1,1), space = "Lab",
                       name="Pearson\nCorrelation") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1,
                                   size = 12, hjust = 1)) +
  coord_fixed() +
  labs(title = "red_wine correlation matrix")
```



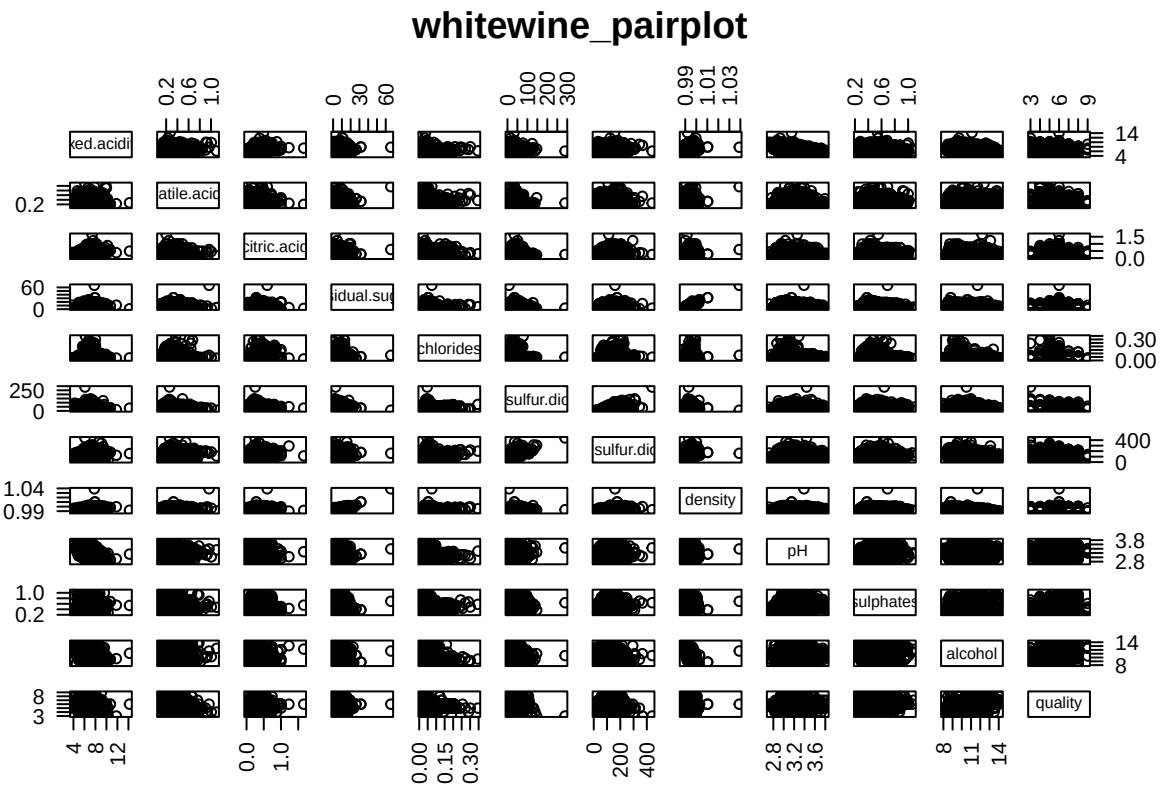

```
#calculate the correlation matrix of white wine.
corr_matrix <- cor(white_wine)
melted_corr_matrix <- melt(corr_matrix)
ggplot(data = melted_corr_matrix, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", mid = "white", high = "red",
                       midpoint = 0, limit = c(-1,1), space = "Lab",
                       name="Pearson\nCorrelation") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1,
                                   size = 12, hjust = 1)) +
  coord_fixed() +
  labs(title = "white_wine correlation matrix")
```




```
# Create a pairplot matrix for red wine
pairs(red_wine, las=2, main= "redwine_pairplot")
```




```
# Create a pairplot matrix for white wine
pairs(white_wine, las=2, main = "whitewine_pairplot")
```




```
# Summarize the dataset
summary(red_wine)
```

Data Quality and characterstics

```
##  fixed.acidity   volatile.acidity  citric.acid    residual.sugar
##  Min.   : 4.60   Min.   :0.1200   Min.   :0.000   Min.   : 0.900
##  1st Qu.: 7.10   1st Qu.:0.3900   1st Qu.:0.090   1st Qu.: 1.900
##  Median : 7.90   Median :0.5200   Median :0.260   Median : 2.200
##  Mean   : 8.32   Mean   :0.5278   Mean   :0.271   Mean   : 2.539
##  3rd Qu.: 9.20   3rd Qu.:0.6400   3rd Qu.:0.420   3rd Qu.: 2.600
##  Max.   :15.90   Max.   :1.5800   Max.   :1.000   Max.   :15.500
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.01200   Min.   : 1.00       Min.   :  6.00       Min.   :0.9901
##  1st Qu.:0.07000   1st Qu.: 7.00       1st Qu.: 22.00       1st Qu.:0.9956
##  Median :0.07900   Median :14.00       Median : 38.00       Median :0.9968
##  Mean   :0.08747   Mean   :15.87       Mean   : 46.47       Mean   :0.9967
##  3rd Qu.:0.09000   3rd Qu.:21.00       3rd Qu.: 62.00       3rd Qu.:0.9978
##  Max.   :0.61100   Max.   :72.00       Max.   :289.00       Max.   :1.0037
##        pH          sulphates         alcohol         quality
##  Min.   :2.740   Min.   :0.3300   Min.   : 8.40   Min.   :3.000
##  1st Qu.:3.210   1st Qu.:0.5500   1st Qu.: 9.50   1st Qu.:5.000
##  Median :3.310   Median :0.6200   Median :10.20   Median :6.000
##  Mean   :3.311   Mean   :0.6581   Mean   :10.42   Mean   :5.636
##  3rd Qu.:3.400   3rd Qu.:0.7300   3rd Qu.:11.10   3rd Qu.:6.000
```

```
##  Max.   :4.010   Max.   :2.0000   Max.   :14.90   Max.   :8.000
```

```
summary(white_wine)
```

```
##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
##  Min.   : 3.800   Min.   :0.0800   Min.   :0.0000   Min.   : 0.600
##  1st Qu.: 6.300   1st Qu.:0.2100   1st Qu.:0.2700   1st Qu.: 1.700
##  Median : 6.800   Median :0.2600   Median :0.3200   Median : 5.200
##  Mean   : 6.855   Mean   :0.2782   Mean   :0.3342   Mean   : 6.391
##  3rd Qu.: 7.300   3rd Qu.:0.3200   3rd Qu.:0.3900   3rd Qu.: 9.900
##  Max.   :14.200   Max.   :1.1000   Max.   :1.6600   Max.   :65.800
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.00900   Min.   :  2.00      Min.   :  9.0        Min.   :0.9871
##  1st Qu.:0.03600   1st Qu.: 23.00      1st Qu.:108.0        1st Qu.:0.9917
##  Median :0.04300   Median : 34.00      Median :134.0        Median :0.9937
##  Mean   :0.04577   Mean   : 35.31      Mean   :138.4        Mean   :0.9940
##  3rd Qu.:0.05000   3rd Qu.: 46.00      3rd Qu.:167.0        3rd Qu.:0.9961
##  Max.   :0.34600   Max.   :289.00      Max.   :440.0        Max.   :1.0390
##        pH          sulphates         alcohol         quality
##  Min.   :2.720   Min.   :0.2200   Min.   : 8.00   Min.   :3.000
##  1st Qu.:3.090   1st Qu.:0.4100   1st Qu.: 9.50   1st Qu.:5.000
##  Median :3.180   Median :0.4700   Median :10.40   Median :6.000
##  Mean   :3.188   Mean   :0.4898   Mean   :10.51   Mean   :5.878
##  3rd Qu.:3.280   3rd Qu.:0.5500   3rd Qu.:11.40   3rd Qu.:6.000
##  Max.   :3.820   Max.   :1.0800   Max.   :14.20   Max.   :9.000
```

```
# Examine the structure of red and white whine datasets
str(red_wine)
```

```
## 'data.frame':    1599 obs. of  12 variables:
##  $ fixed.acidity       : num  7.4 7.8 7.8 11.2 7.4 7.4 7.9 7.3 7.8 7.5 ...
##  $ volatile.acidity    : num  0.7 0.88 0.76 0.28 0.7 0.66 0.6 0.65 0.58 0.5 ...
##  $ citric.acid         : num  0 0 0.04 0.56 0 0 0.06 0 0.02 0.36 ...
##  $ residual.sugar      : num  1.9 2.6 2.3 1.9 1.9 1.8 1.6 1.2 2 6.1 ...
##  $ chlorides           : num  0.076 0.098 0.092 0.075 0.076 0.075 0.069 0.065 0.073 0.071 ...
##  $ free.sulfur.dioxide : num  11 25 15 17 11 13 15 15 9 17 ...
##  $ total.sulfur.dioxide: num  34 67 54 60 34 40 59 21 18 102 ...
##  $ density             : num  0.998 0.997 0.997 0.998 0.998 ...
##  $ pH                  : num  3.51 3.2 3.26 3.16 3.51 3.51 3.3 3.39 3.36 3.35 ...
##  $ sulphates           : num  0.56 0.68 0.65 0.58 0.56 0.56 0.46 0.47 0.57 0.8 ...
##  $ alcohol             : num  9.4 9.8 9.8 9.8 9.4 9.4 9.4 10 9.5 10.5 ...
##  $ quality             : int  5 5 5 6 5 5 5 7 7 5 ...
```

```
str(white_wine)
```

```
## 'data.frame':    4898 obs. of  12 variables:
##  $ fixed.acidity       : num  7 6.3 8.1 7.2 7.2 8.1 6.2 7 6.3 8.1 ...
##  $ volatile.acidity    : num  0.27 0.3 0.28 0.23 0.23 0.28 0.32 0.27 0.3 0.22 ...
##  $ citric.acid         : num  0.36 0.34 0.4 0.32 0.32 0.4 0.16 0.36 0.34 0.43 ...
##  $ residual.sugar      : num  20.7 1.6 6.9 8.5 8.5 6.9 7 20.7 1.6 1.5 ...
##  $ chlorides           : num  0.045 0.049 0.05 0.058 0.058 0.05 0.045 0.045 0.049 0.044 ...
##  $ free.sulfur.dioxide : num  45 14 30 47 47 30 30 45 14 28 ...
##  $ total.sulfur.dioxide: num  170 132 97 186 186 97 136 170 132 129 ...
##  $ density             : num  1.001 0.994 0.995 0.996 0.996 ...
##  $ pH                  : num  3 3.3 3.26 3.19 3.19 3.26 3.18 3 3.3 3.22 ...
##  $ sulphates           : num  0.45 0.49 0.44 0.4 0.4 0.44 0.47 0.45 0.49 0.45 ...
```

```
##  $ alcohol             : num  8.8 9.5 10.1 9.9 9.9 10.1 9.6 8.8 9.5 11 ...
##  $ quality             : int  6 6 6 6 6 6 6 6 6 6 ...
```

```
# Count missing values per variable in red and white wines
colSums(is.na(red_wine))
```

```
##        fixed.acidity     volatile.acidity          citric.acid
##                    0                    0                    0
##       residual.sugar            chlorides  free.sulfur.dioxide
##                    0                    0                    0
## total.sulfur.dioxide              density                   pH
##                    0                    0                    0
##            sulphates              alcohol              quality
##                    0                    0                    0
```

```
colSums(is.na(white_wine))
```

```
##        fixed.acidity     volatile.acidity          citric.acid
##                    0                    0                    0
##       residual.sugar            chlorides  free.sulfur.dioxide
##                    0                    0                    0
## total.sulfur.dioxide              density                   pH
##                    0                    0                    0
##            sulphates              alcohol              quality
##                    0                    0                    0
```

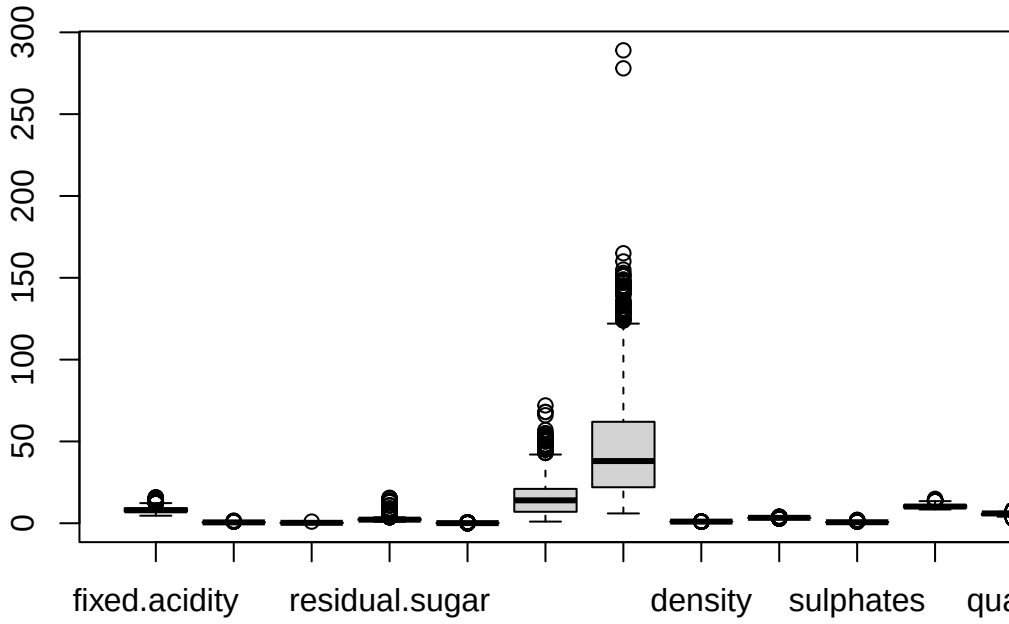
```
# Count the number of duplicate rows in red and white wines
num_duplicates <- sum(duplicated(red_wine))
print(num_duplicates)
```

```
## [1] 240
```

```
num_duplicates2 <- sum(duplicated(white_wine))
print(num_duplicates2)
```

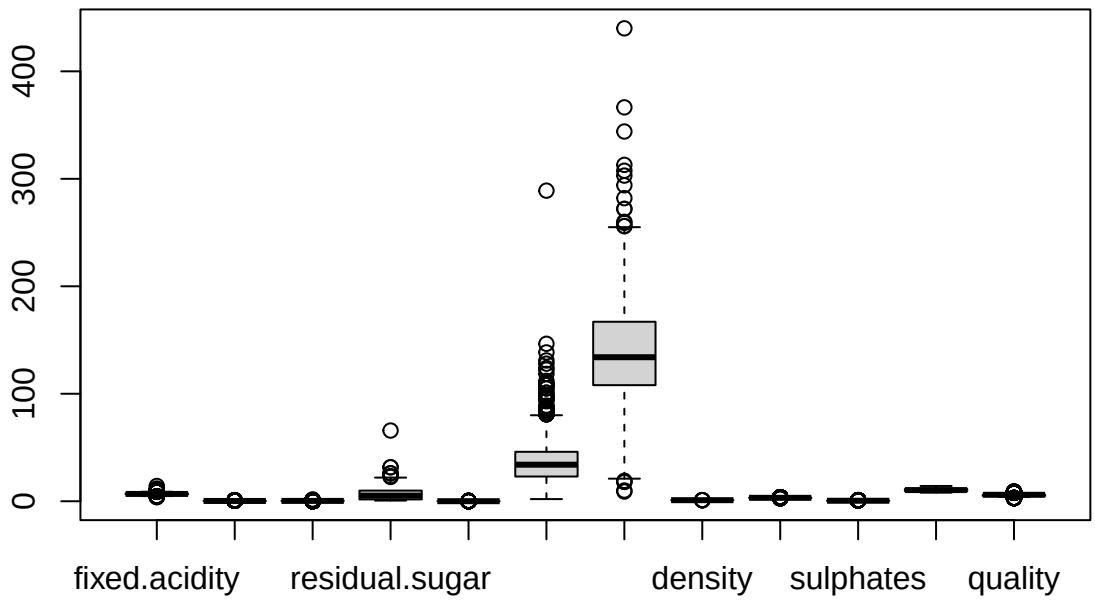
```
## [1] 937
```

```
#Create a box plot of red wine.
boxplot(red_wine)
```



Outliers and associations

```
#Create a box plot of white wine.
boxplot(white_wine)
```



EDA summary of red wines

The red wine data set contains 1599 observations (rows) and 12 variables (columns).

The variables in the red wine data set include various chemical properties such as fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, and quality.

The "quality" variable represents the sensory quality of the wines which is ranging from 3 to 8.

From the summary and structure of red wines we could see the mean, median, minimum, maximum, and quartiles for each variable in the dataset. and we could observe there is a central tendency on the spread of the data

From The structure output we could see that all the variables are of numeric type.

There are no missing values in any variable of red_wine.

There are 240 duplicates in red_wine data set.

From the correlation matrix and pair plots we could obviously see total and free sulfur dioxide are highly related and also denisty and citric.acid are related with fixed acidity.

From the box plot we could see the outliers exist for variables fixed.acidity, residual.sugar, free.sulfur dioxide, total.sulfur.dioxide

EDA summary of white wines

The white wine data set contains 4898 observations (rows) and 12 variables (columns).

The variables in the white wine data set are the same as those in the red wine data set, representing chemical properties and quality.

From the summary and structure of white wines we could see the mean, median, minimum, maximum, and quartiles for each variable in the dataset. and we could observe there is a central tendency on the spread of the data

From The structure output we could see that all the variables are of numeric type.

There are no missing values in any variable of white_wine.

There are 937 duplicates in white_wine data set.

From correlation matrix and pair plots we could see total and free sulfur dioxide are correlated and also density and residual sugar are linearly correlated.

There are more number of outliers in total and free sulfur dioxide attributes and compartively less number of outliers in residual sugar.

(b) Perform PCA on the red wines. What summarizations can you extract from the biplot and scree plots.

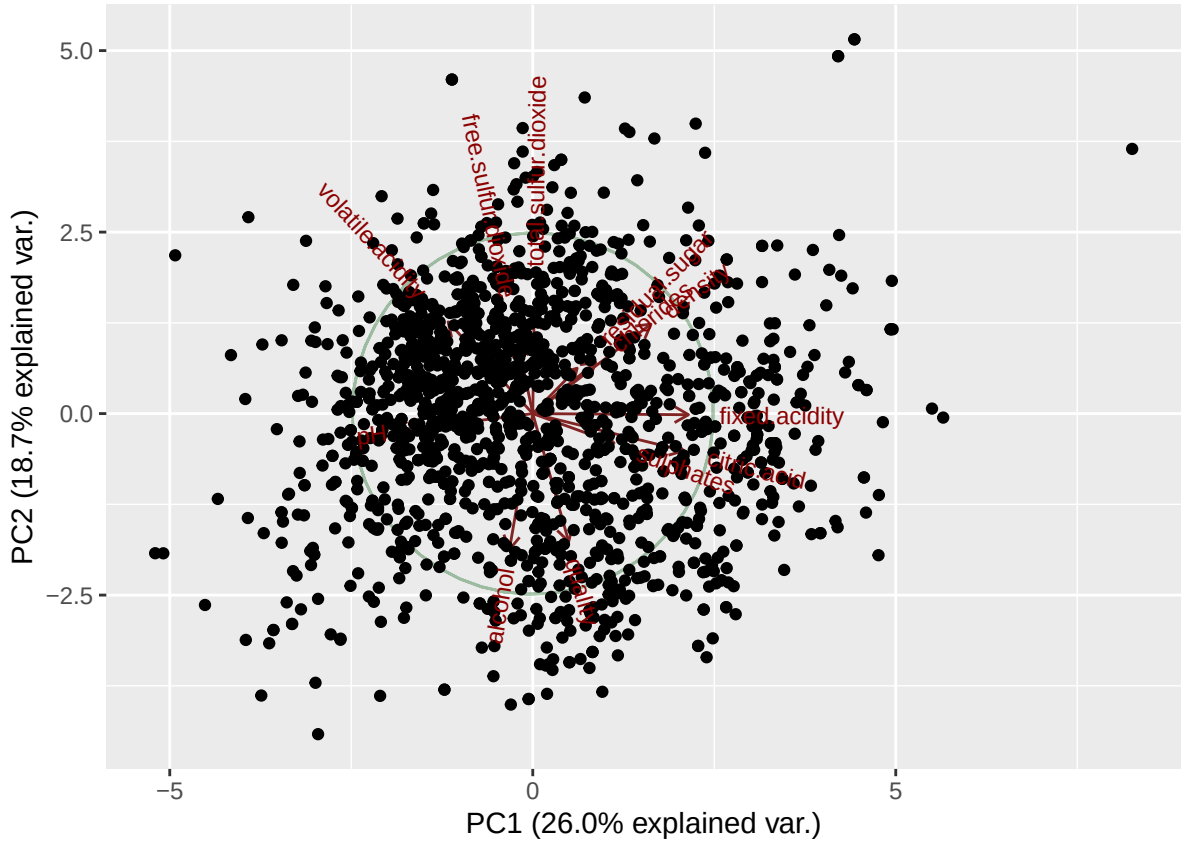
```
# Exclude non-numeric columns
numeric_cols <- sapply(red_wine, is.numeric)
red_wine_numeric <- red_wine[, numeric_cols]
```

```
# Perform PCA
pca <- prcomp(red_wine_numeric, scale. = TRUE)
```

```
# Create a scree plot
plot(pca, type = "lines", main = "Scree Plot of Red wines")
```

Answer:

```
#create a biplot.
ggbiplot(pca, obs.scale = 1, var.scale = 1, ellipse = TRUE, circle = TRUE) +
  scale_color_discrete(name = '') +
  theme(legend.direction = 'horizontal', legend.position = 'top')
```



Observations for red wines:

A scree plot captures how much variation each principal component captures in the data

From red wines scree plot we could say that first four principal components captures most of the variance in the data, so they are enough to describe the data.

From bi-plot between PC1 and PC2 of red wines we could observe that fixed acidity, citric acid, sulphates are in same direction of PC1 so they are positively correlated with PC1

From bi-plot, pH has a negative association with PC1.

(c) Perform PCA on the white wines. What summarizations can you extract from the biplot and scree plots.

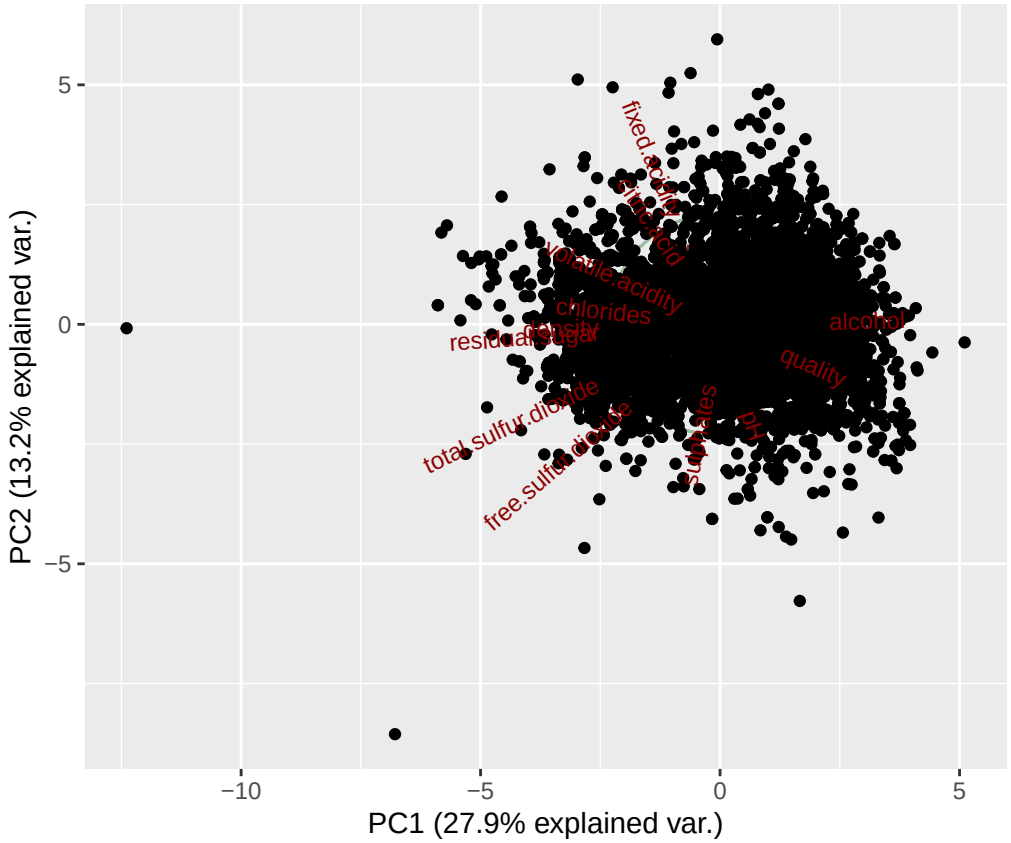
```
# Exclude non-numeric columns
numeric_cols2 <- sapply(white_wine, is.numeric)
white_wine_numeric <- white_wine[, numeric_cols2]
```

```
# Perform PCA
pca2 <- prcomp(white_wine_numeric, scale. = TRUE)
```

```r
# Create a scree plot
plot(pca2, type = "lines", main = "Scree Plot of white wines")
```

Answer:

```r
#create a biplot.
ggbiplot(pca2, obs.scale = 1, var.scale = 1, ellipse = TRUE, circle = TRUE) +
  scale_color_discrete(name = '') +
  theme(legend.direction = 'horizontal', legend.position = 'top')
```



Observations for white wines: #### A scree plot captures how much variation each principal component captures in the data #### From white wines scree plot we could say that first three principal components captures most of the variance in the data, so they are enough to describe the data. #### From bi-plot between PC1 and PC2 of white wines we could observe that alcohol and quality are in same direction of PC1 so they are positively correlated with PC1 #### From bi-plot, residual sugar, total and free sulfur dioxide have negative association with PC1 values.

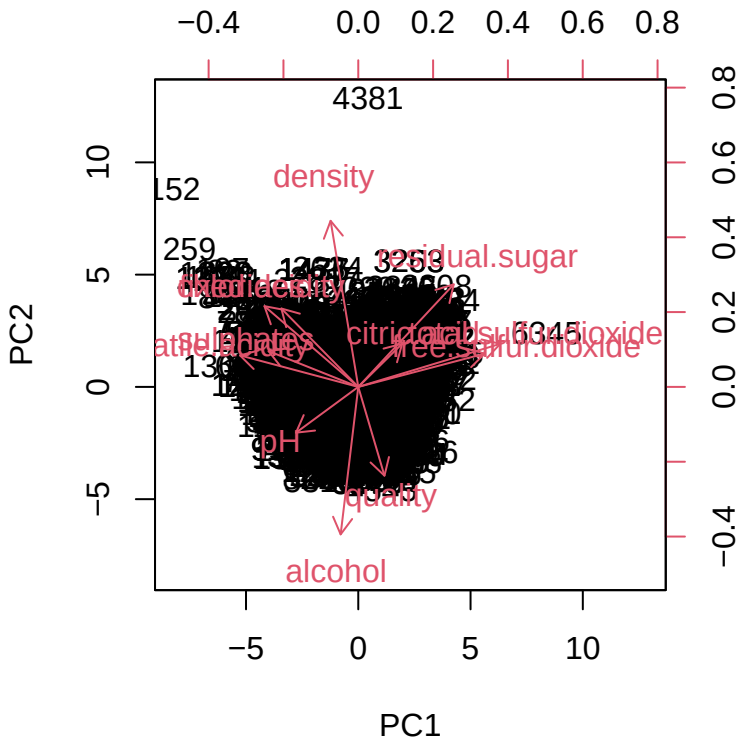
(d) Combine the data and perform PCA. Color the biplot according to wine type.

```
# Add a column to indicate the wine type
red_wine$wine_type <- "red"
white_wine$wine_type <- "white"
```

```
# Combine the red wine and white wine datasets
wine <- rbind(red_wine, white_wine)
```

```
# Perform PCA on the combined data
set.seed(123)
pca3 <- prcomp(wine[,-ncol(wine)], scale. = TRUE)
```

```
# Create the biplot
biplot(pca3, choices = c(1, 2), scale = 0)
```

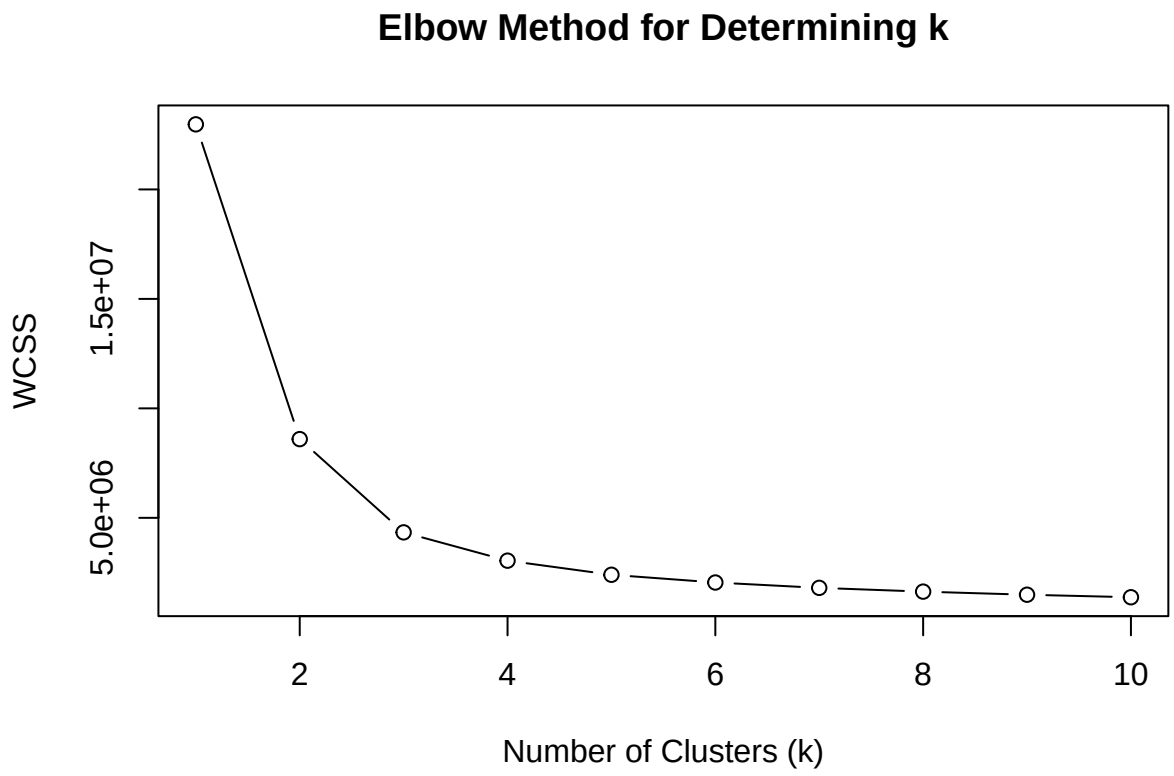


Answer:

```
# Add color to the bi plot according to wine type
set.seed(123)
ggbiplot::ggbiplot(pca3, obs.scale = 1, var.scale = 1,
                   groups = wine$wine_type,
                   ellipse = TRUE, circle = TRUE) +
  theme_minimal()
```

(e) Perform k-means using the wine data. Justify your choice in "k" and report your findings.

```
#Elbow method.
# Determine the optimal number of clusters using the elbow method
wcss <- numeric(length = 10)  # Initialize vector to store WCSS values
# Iterate over different values of k
for (k in 1:10) {
  # Perform k-means clustering
  set.seed(123)
  kmeans_model <- kmeans(wine[,-ncol(wine)], centers = k, nstart = 10)
  # Store the within-cluster sum of squares (WCSS)
  wcss[k] <- kmeans_model$tot.withinss
}
# Plot the WCSS values
plot(1:10, wcss, type = "b", xlab = "Number of Clusters (k)", ylab = "WCSS",
     main = "Elbow Method for Determining k")
```



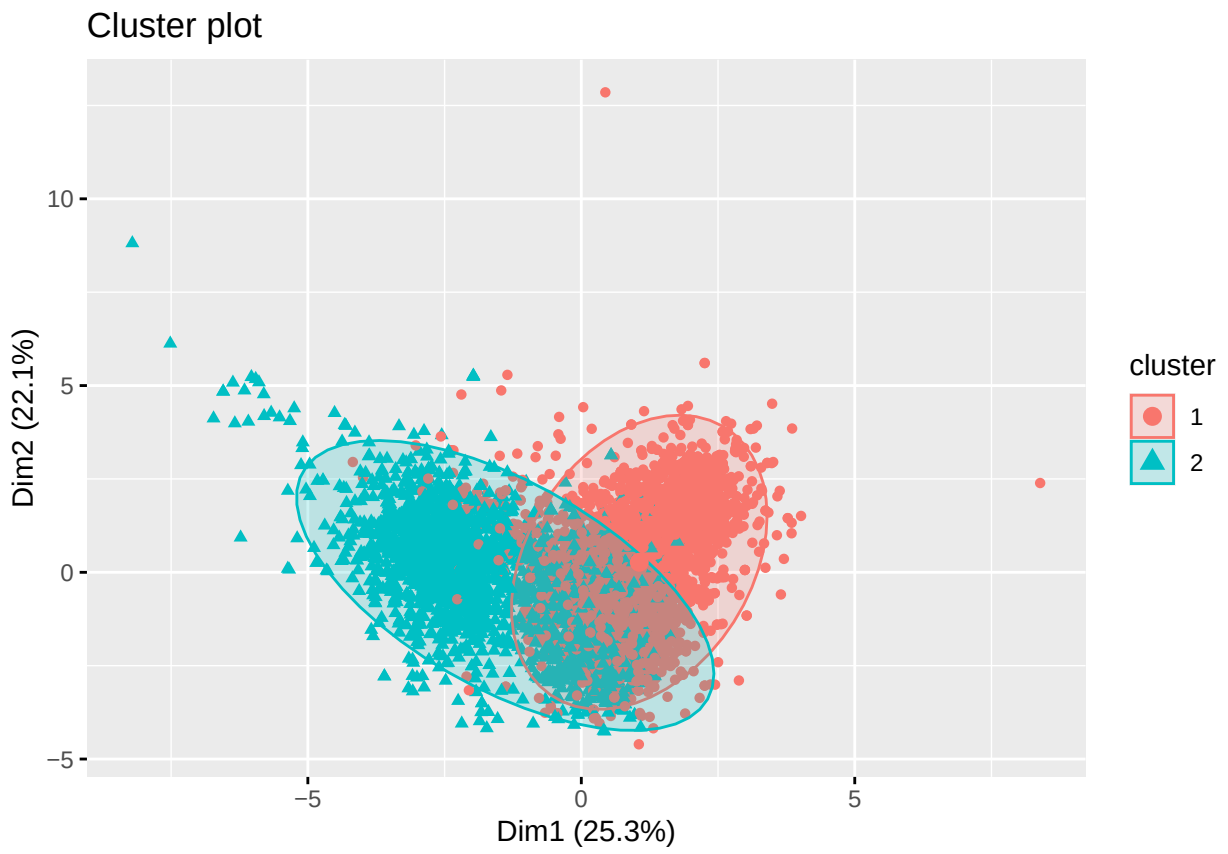
Answer:

```
# Perform k-means clustering with the chosen value of k
set.seed(123)
kmeans_model <- kmeans(wine[,-ncol(wine)], centers = 2, nstart = 10)
# Get cluster assignments for each data point
cluster_assignments <- as.factor(kmeans_model$cluster)
table(cluster_assignments)  # Frequency table of cluster assignments
```

Basis on the elbow plot above the optimal k value is 2.

```
## cluster_assignments
##    1    2
## 3689 2808
```

```
fviz_cluster(kmeans_model, data = wine[, -ncol(wine)], geom = "point", ellipse.type = "norm")
```



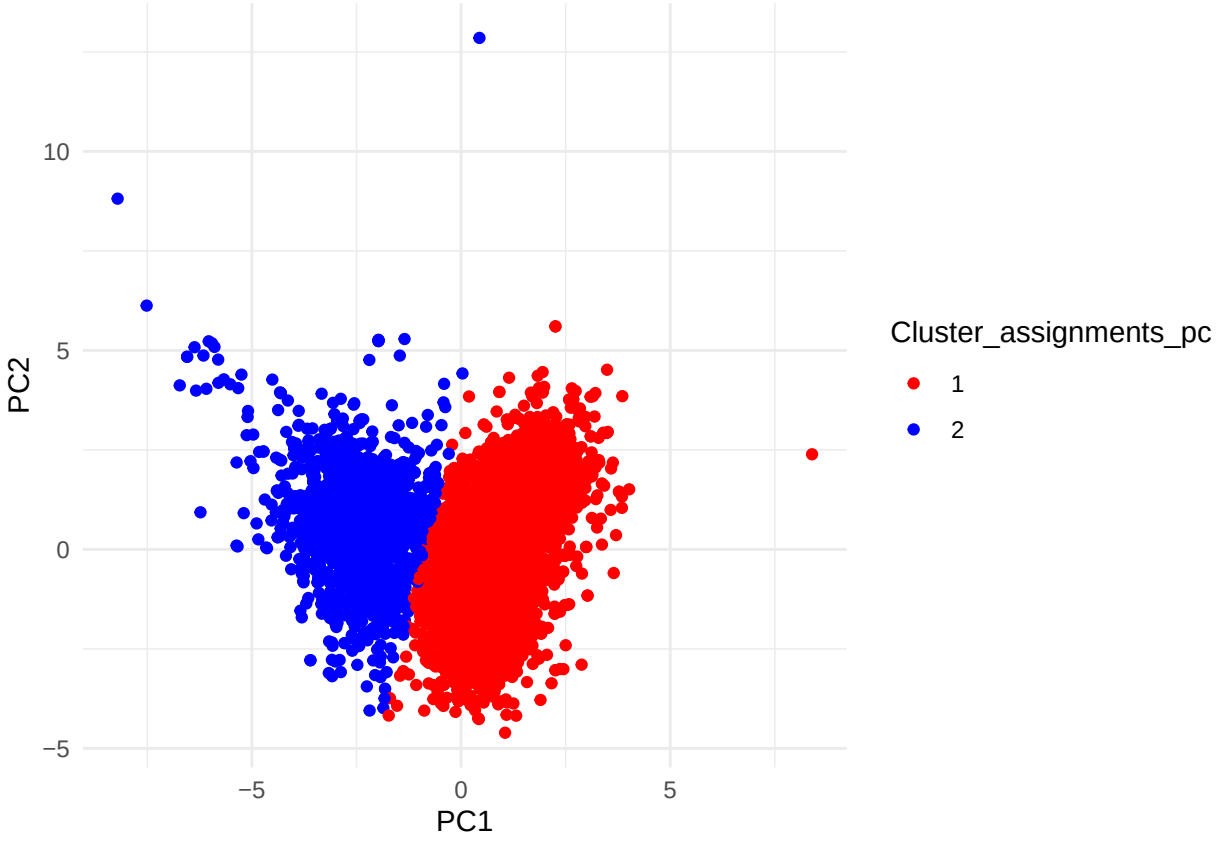
The optimal k value is 2 for the wine data as shown in elbow plot. This might be due to two different wines red and white wine data.

From the table output we can see after clustering there is good amount of separation between data and has nearly equal number of points distributed between two clusters. But there is some overlap which can be solved by preprocessing the data further.

(f) Perform k-means using Principal Components from the wine data. Justify your choice in "k" and report your findings.

```
# Perform PCA
pca_k <- prcomp(wine[,-ncol(wine)], scale. = TRUE)
# Extract PC scores
PC1_k <- pca_k$x[, 1]
PC2_k <- pca_k$x[, 2]
PC_dats <- cbind(PC1_k, PC2_k)
# Perform k-means clustering with the chosen value of k
set.seed(123)
kmeans_model_pc <- kmeans(PC_dats, centers = 2, nstart = 10)
# Get cluster assignments for each data point
cluster_assignments_pc <- as.factor(kmeans_model_pc$cluster)
# Plot the k-means clusters using PC1 and PC2
# Define a vector of colors for each cluster
cluster_colors <- c("red", "blue")
```

```
ggplot(wine[,-ncol(wine)], aes(x = PC1_k, y = PC2_k, color = cluster_assignments_pc)) +
  geom_point() +
  labs(x = "PC1", y = "PC2", color = "Cluster_assignments_pc") +
  scale_color_manual(values = cluster_colors) +
  theme_minimal()
```



Answer:

Observations:

The optimal value of k is 2 for principal compenent data of wine.

The data is clustered nicely into two groups may be indicating red and white wines.

The no.of data points between the two group are also nearly equal.

(g) How do your answers between E and F compare.

Answer:

Overall, I think E and F are nearly similar plots and also the separation nearly seems same.

But overall the F cluster separation is better than E as there is no overlap between clusters when plotted w.r.t principal component data.

"E" clustering has better split of number of data points in each cluster.

Question 3: Consider the wine quality data.

```
 # Load the red wine and white wine data sets
red_wine <- read.csv("D:/University at Buffalo CSE/Spring Semester 2023/STA 546 DataM II/R/Assignment 4
white_wine <- read.csv("D:/University at Buffalo CSE/Spring Semester 2023/STA 546 DataM II/R/Assignment
dim(red_wine)
```

(a) Construct a Self Organizing Map that clusters the samples, use a sensible grid choice. Color the samples on th6e map by wine color.

```
## [1] 1599   12
```

```
dim(white_wine)
```

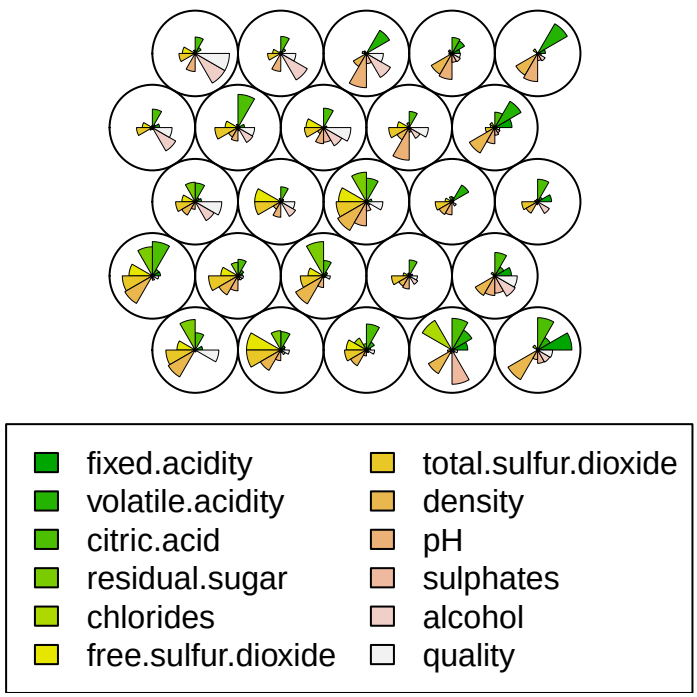
```
## [1] 4898   12
```

```
# Add a column to indicate the wine type
red_wine$wine_type <- "red"
white_wine$wine_type <- "white"

# Combine the red wine and white wine datasets
wines <- rbind(red_wine, white_wine)

#scale the wines data
wines.scaled <- scale(wines[,-ncol(wines)])
```

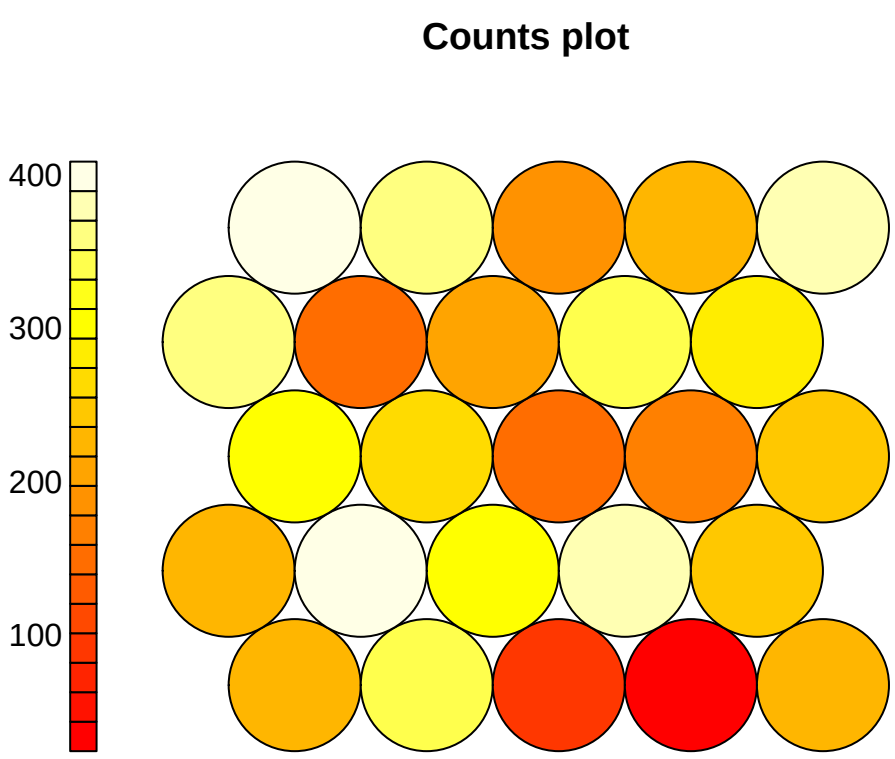
```
 # fit an SOM
set.seed(123)
som_grid <- somgrid(xdim = 5, ydim = 5, topo = "hexagonal")
wine.som <- som(wines.scaled, grid = som_grid, rlen = 3000)
```

```
#SOM codes
codes <- wine.som$codes[[1]]
plot(wine.som, main = "Wines Data")
```

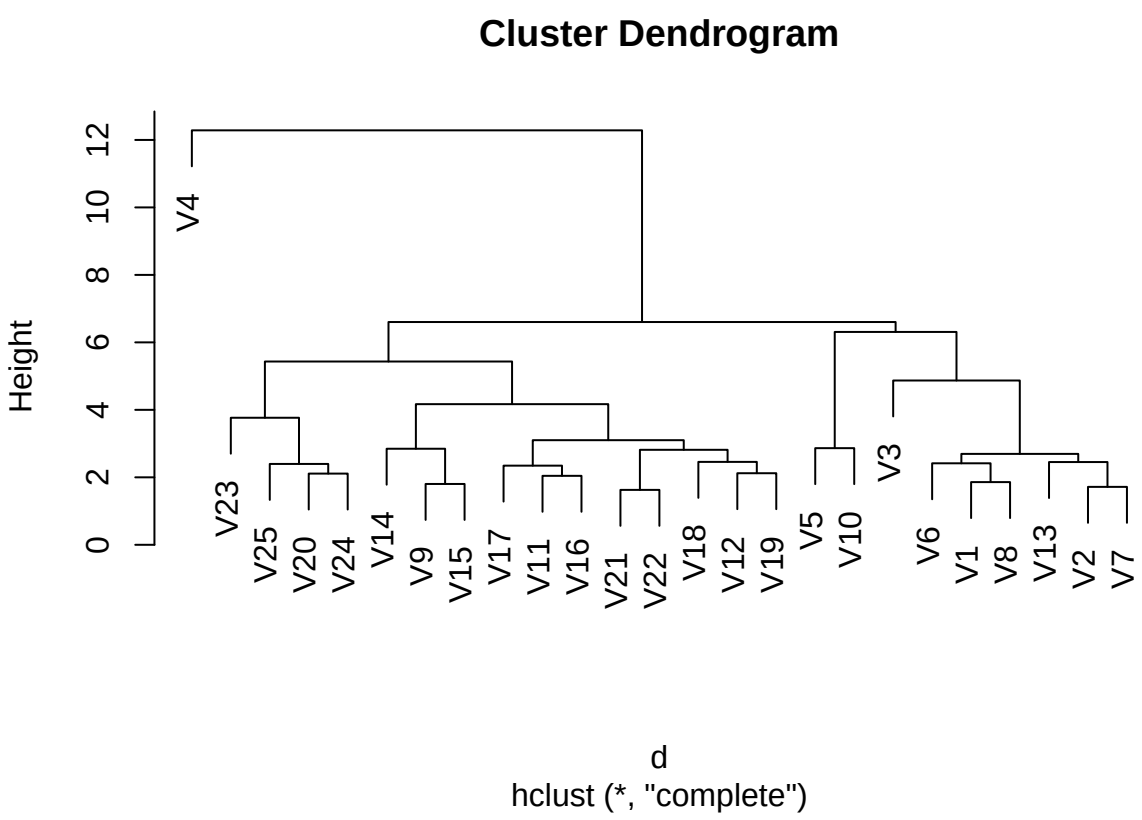
```
## Warning in par(opar): argument 1 does not name a graphical parameter
```




```
plot(wine.som, type = "count")
```



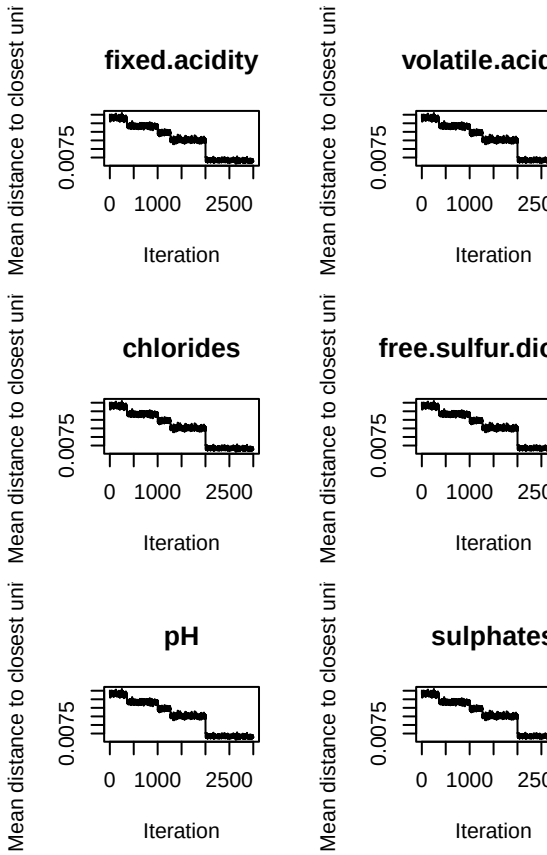

```
d <- dist(codes)
hc <- hclust(d)
plot(hc)
```




```
# plot the SOM with the found clusters
som_cluster <- cutree(hc, h = 6)
my_pal <- c("red", "white")
my_bhcol <- my_pal[som_cluster]
graphics.off()
plot(wine.som, type = "mapping", col = "black", bgcol = my_bhcol)
add.cluster.boundaries(wine.som, som_cluster)
```

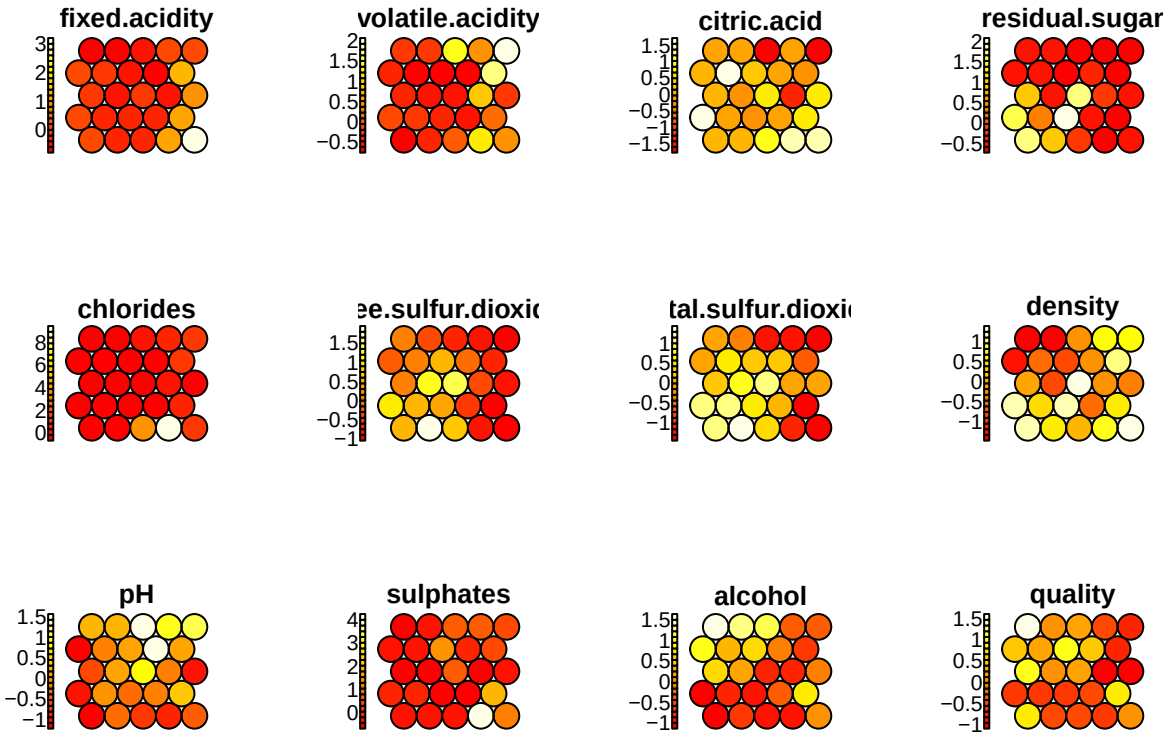
Plot the som with the found clusters here.

```
#Phase plots for each variable in dataset
par(mfrow = c(3, 4))
for (i in 1:12){
plot(wine.som, type = "changes", property=codes[,i], main = ifelse(is.na(colnames(codes)[i]), "", colnar
}
```



(b) Construct phase-plots for the different variables in the dataset.

```
# phase/Component plane plots for each variable in dataset.
par(mfrow = c(3, 4))
for (i in 1:12){
plot(wine.som, type = "property", property=codes[,i], main = ifelse(is.na(colnames(codes)[i]), "", colna
}
```

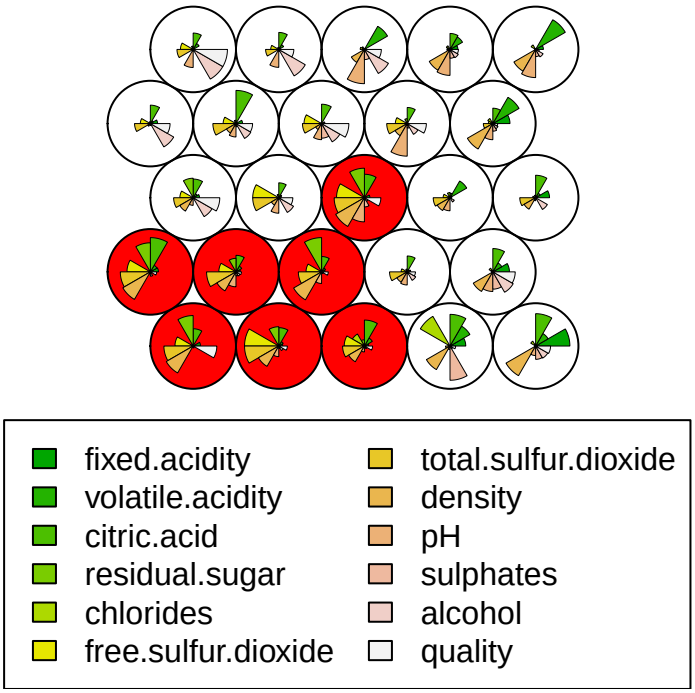



```
# Plot the prototypes on the SOM
plot(wine.som, type = "codes", bgcol = my_bhcol, main = "Prototypes on SOM")
```

(c) Plot the prototypes on the SOM

```
## Warning in par(opar): argument 1 does not name a graphical parameter
```

Prototypes on SOM



Question 4: Consider the following webgraphs.

Read data from CSV file

```
dataset <- read.csv("D:/University at Buffalo CSE/Spring Semester 2023/STA 546 DataM II/R/Assignment 4 v
```

```
dim(dataset)
```

```
## [1] 887   8
```

```
# Check the structure and summary statistics of the data set
str(dataset)
```

```
## 'data.frame':    887 obs. of  8 variables:
##  $ Survived               : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass                 : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name                   : chr  "Mr. Owen Harris Braund" "Mrs. John Bradley (Florence Briggs Thayer)
##  $ Sex                    : chr  "male" "female" "female" "female" ...
##  $ Age                    : num  22 38 26 35 35 27 54 2 27 14 ...
##  $ Siblings.Spouses.Aboard: int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parents.Children.Aboard: int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Fare                   : num  7.25 71.28 7.92 53.1 8.05 ...
```

```
summary(dataset)
```

```
##     Survived          Pclass          Name               Sex
##  Min.   :0.0000   Min.   :1.000   Length:887         Length:887
##  1st Qu.:0.0000   1st Qu.:2.000   Class :character   Class :character
##  Median :0.0000   Median :3.000   Mode  :character   Mode  :character
##  Mean   :0.3856   Mean   :2.306
##  3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :1.0000   Max.   :3.000
##       Age        Siblings.Spouses.Aboard Parents.Children.Aboard
##  Min.   : 0.42   Min.   :0.0000          Min.   :0.0000
##  1st Qu.:20.25   1st Qu.:0.0000          1st Qu.:0.0000
##  Median :28.00   Median :0.0000          Median :0.0000
##  Mean   :29.47   Mean   :0.5254          Mean   :0.3833
##  3rd Qu.:38.00   3rd Qu.:1.0000          3rd Qu.:0.0000
##  Max.   :80.00   Max.   :8.0000          Max.   :6.0000
##       Fare
##  Min.   :  0.000
##  1st Qu.:  7.925
##  Median : 14.454
##  Mean   : 32.305
##  3rd Qu.: 31.137
##  Max.   :512.329
```

```
# Calculate the survival rate overall and by different categories
survival_rate <- mean(dataset$Survived)
cat("Survival Rate: ", survival_rate, "\n")
```

```
## Survival Rate:  0.3855693
```

```
# Is there evidence that women and children were evacuated first?
# Calculate the survival rate for women and children separately
women_survival_rate <- mean(dataset$Survived[dataset$Sex == "female"])
children_survival_rate <- mean(dataset$Survived[dataset$Age < 18])
```

```r
cat("Survival Rate for Women: ", women_survival_rate, "\n")
```

```
## Survival Rate for Women:  0.7420382
```

```r
cat("Survival Rate for Children: ", children_survival_rate, "\n")
```

```
## Survival Rate for Children:  0.5
```

```r
# Characteristics/demographics more likely in surviving passengers
# Calculate survival rates by Pclass, Sex, and Age categories
survival_rate_pclass <- aggregate(Survived ~ Pclass, data = dataset, FUN = mean)
survival_rate_sex <- aggregate(Survived ~ Sex, data = dataset, FUN = mean)
survival_rate_age <- aggregate(Survived ~ (Age < 18), data = dataset, FUN = mean)
```

```r
cat("Survival Rate by Pclass:\n")
```

```
## Survival Rate by Pclass:
```

```r
print(survival_rate_pclass)
```

```
##   Pclass  Survived
## 1      1 0.6296296
## 2      2 0.4728261
## 3      3 0.2443532
```

```r
cat("Survival Rate by Sex:\n")
```

```
## Survival Rate by Sex:
```

```r
print(survival_rate_sex)
```

```
##      Sex  Survived
## 1 female 0.7420382
## 2   male 0.1902269
```

```r
cat("Survival Rate by Age (<18):\n")
```

```
## Survival Rate by Age (<18):
```

```r
print(survival_rate_age)
```

```
##   Age < 18  Survived
## 1    FALSE 0.3659181
## 2     TRUE 0.5000000
```

```r
# Characteristics/demographics more likely in passengers that perished
# Calculate the opposite of survival rates for Pclass, Sex, and Age categories
perished_rate_pclass <- aggregate(Survived ~ Pclass, data = dataset, FUN = function(x) 1 - mean(x))
perished_rate_sex <- aggregate(Survived ~ Sex, data = dataset, FUN = function(x) 1 - mean(x))
perished_rate_age <- aggregate(Survived ~ (Age < 18), data = dataset, FUN = function(x) 1 - mean(x))
```

```r
cat("Perished Rate by Pclass:\n")
```

```
## Perished Rate by Pclass:
```

```r
print(perished_rate_pclass)
```

```
##   Pclass  Survived
## 1      1 0.3703704
## 2      2 0.5271739
## 3      3 0.7556468
```

```r
cat("Perished Rate by Sex:\n")
```

```
## Perished Rate by Sex:
```

```r
print(perished_rate_sex)
```

```
##      Sex  Survived
## 1 female 0.2579618
## 2   male 0.8097731
```

```r
cat("Perished Rate by Age (<18):\n")
```

```
## Perished Rate by Age (<18):
```

```r
print(perished_rate_age)
```

```
##   Age < 18  Survived
## 1    FALSE 0.6340819
## 2     TRUE 0.5000000
```

```r
# Probability of survival for specific passengers
# Define the characteristics of the passengers
passenger1 <- list(Pclass = 1, Sex = "female", Age = 22)
passenger2 <- list(Pclass = 3, Sex = "male", Age = 24)

# Calculate the probability of survival for the defined passengers
prob_survival_passenger1 <- mean(dataset$Survived[dataset$Pclass == passenger1$Pclass &
                                                  dataset$Sex == passenger1$Sex &
                                                  dataset$Age == passenger1$Age])
prob_survival_passenger2 <- mean(dataset$Survived[dataset$Pclass == passenger2$Pclass &
                                                  dataset$Sex == passenger2$Sex &
                                                  dataset$Age == passenger2$Age])
cat("Probability of survival for Passenger 1: ", prob_survival_passenger1, "\n")
```

```
## Probability of survival for Passenger 1:  1
```

```r
cat("Probability of survival for Passenger 2: ", prob_survival_passenger2, "\n")
```

```
## Probability of survival for Passenger 2:  0.1818182
```

Yes my analysis supports the movie here passenger 1 is rose travelling in first class age 24 probability of survival is 1 so she survives. and passenger 2 jack male with age 24 chance is surviving is 0.18 which is pretty low.

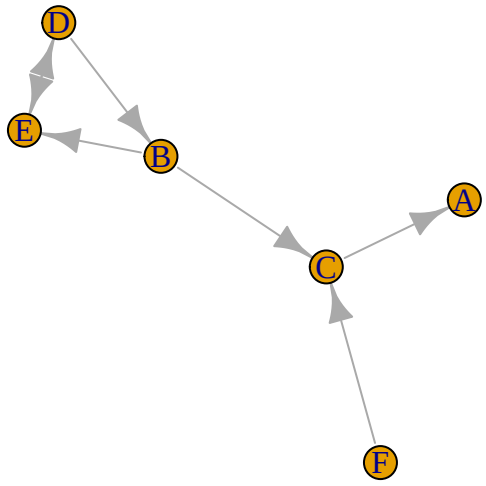
So my results support the popular movie "Titanic"

Question 5: Consider the following webgraphs.

(a) Compute the PageRank vector of Webgraph A for damping constants p = 0.05, 0.25, 0.50, 0.75, and 0.95. How sensitive is the PageRank vector, and overall ranking of importance, to the damping constant? Does the relative ranking of importance according to PageRank support your intuition?

```
nodes <- data.frame(names = c("A", "B", "C", "D" , "E" , "F"))
```

```
relations <- data.frame(
from = c("C", "B", "D", "D", "B", "E", "F"),
to = c("A", "C", "B", "E", "E", "D", "C"))
g <- graph.data.frame(relations, directed = TRUE, vertices = nodes)
plot(g)
```



Answer:

```
#page vector at damping constant 0.05
pg <- page.rank(g, damping = .05)
pg$vector
```

```
##         A         B         C         D         E         F
## 0.1683271 0.1639395 0.1718214 0.1681380 0.1680380 0.1597361
```

```
#page rank vector at damping constant 0.25
pg <- page.rank(g, damping = .25)
pg$vector
```

```
##         A         B         C         D         E         F
## 0.1786588 0.1544288 0.1848587 0.1758772 0.1737324 0.1324441
```

```
#page rank vector at damping at 0.50
pg <- page.rank(g, damping = .5)
pg$vector
```

```
##          A          B          C          D          E          F
## 0.19227231 0.14719411 0.18583257 0.19135235 0.18399264 0.09935603
```

```
#page rank vector at damping at 0.75
 pg <- page.rank(g, damping = .75)
pg$vector
```

```
##          A          B          C          D          E          F
## 0.19399617 0.14778661 0.17077331 0.21832113 0.20320659 0.06591619
```

```
#page rank vector at damping constant 0.25
pg <- page.rank(g, damping = .95)
pg$vector
```

```
##          A          B          C          D          E          F
## 0.17305017 0.15761096 0.14454445 0.25658531 0.23247617 0.03573294
```

observations:

In general The PageRank algorithm is an algorithm used by search engines to rank web pages based on their importance and relevance

PageRank works by assigning a numerical value, called a PageRank score, to each web page in a search engine's index. The score represents the importance of the page in the overall link structure of the web. The underlying idea behind PageRank is that a web page is considered more important if it receives many links from other important pages.

As we change the damping factors there are different pages that are highly ranked each time. for example at damping factor 0.05 Webpage 'C' is highly ranked. similarly at damping factor 0.25 also C is highly ranked.

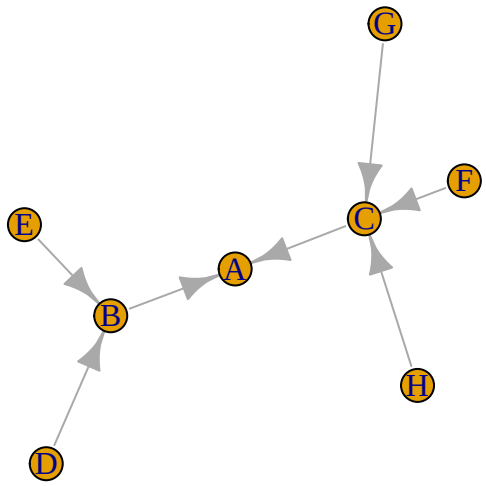
At damping factor 0.5 'A'. at damping factor 0.75 D is highly ranked webpage and damping factor 0.95 'D' is highly ranked webpage.

I think as we are keep on increasing the damping factor in the PageRank algorithm it is reinforcing the importance of link structure and making the algorithm more focused on the interconnectedness of web pages and it is improving the convergence and reducing the impact of random jumps, and enhancing the differentiation of PageRank scores among pages based on their link-based authority. So, yes the relative ranking of importance according to PageRank generally aligns with our intuition

(b) Compute the PageRank vector of Webgraph A for damping constants p = 0.05, 0.25, 0.50, 0.75, and 0.95. How sensitive is the PageRank vector, and overall ranking of importance, to the damping constant? Does the relative ranking of importance according to PageRank support your intuition?

```
nodes <- data.frame(names = c("A", "B", "C", "D" , "E" , "F" , "G" , "H"))
```

```
relations <- data.frame(
from = c("B", "C", "D", "E", "F", "G", "H"),
to = c("A", "A", "B", "B", "C", "C", "C"))
g <- graph.data.frame(relations, directed = TRUE, vertices = nodes)
plot(g)
```



Answer:

```
pg <- page.rank(g, damping = .25)
pg$vector
```

```
##          A          B          C          D          E          F          G
## 0.18012422 0.14906832 0.17391304 0.09937888 0.09937888 0.09937888 0.09937888
##          H
## 0.09937888
```

Observations:

As the damping factor is set to 0.25, it means there is a 25% chance of randomly jumping to any page on the web instead of following a link. This introduces a degree of randomness and models user behavior in navigating the web.

The relationship between the number of incoming links and relative importance, the PageRank algorithm tends to assign higher scores to pages that receive more incoming links from

important and well-connected pages. This reflects the notion that pages with many incoming links are considered more reputable or authoritative.

And also our pagerank algorithm not only considers just the incoming links it also considers the quality of the links (from what webpages the links are coming from). for example if just consider consider no.of incoming links webpage C should have high rank score but in our output we have A with high rank score.

So, In my opinion, The relative importance ranking determined by PageRank,generally confirms our intuition. However, it's important to take into account other variables that affect PageRank scores, such as link quality, relevance, and the unique properties of the web network under analysis.

Question 6: Data released from the US department of Commerce, Bureau of the Census is available in R

data(state)

?state

```
data(state)
#scale the state.x77 data.
state.x77.scaled <- scale(state.x77)
dim(state.x77.scaled)
```

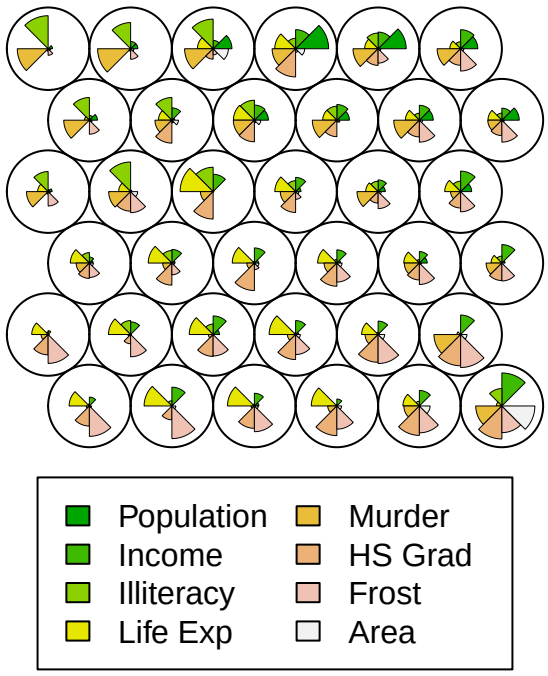
(a) Focus on the data {Population, Income, Illiteracy, Life Exp, Murder, HS Grad, Frost, Area}. Cluster this data using SOM. Keep the class labels (region, or state name) in mind, but do not use them in the modeling. Report your detailed findings. ** You may have done this step in an earlier assignment.

```
## [1] 50  8
```

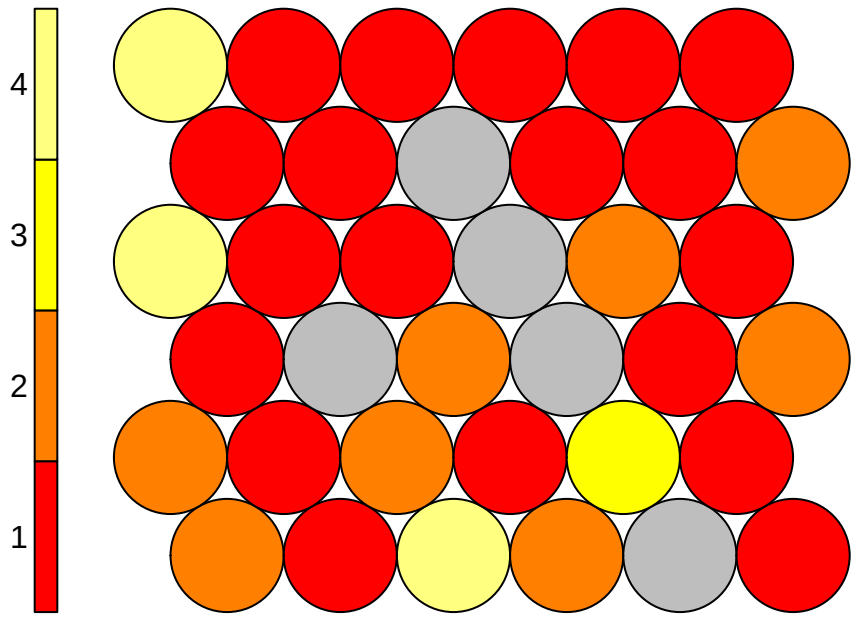
```
# fit an SOM
set.seed(123)
state.x77.grid <- somgrid(xdim = 6, ydim = 6, topo = "hexagonal")
state.x77.som <- som(state.x77.scaled, grid = state.x77.grid, rlen = 3000)

#state.x77 codes
codes_x <- state.x77.som$codes[[1]]
```

```
plot(state.x77.som, main = "state.x77 Data")
```

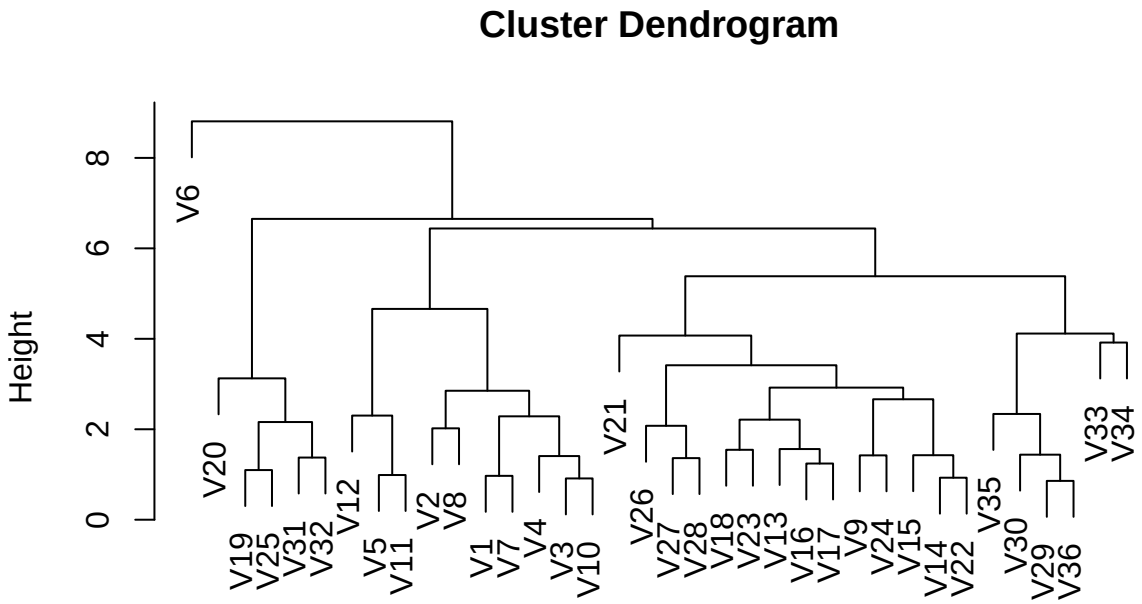
```
## Warning in par(opar): argument 1 does not name a graphical parameter
```




```
plot(state.x77.som, type = "count")
```

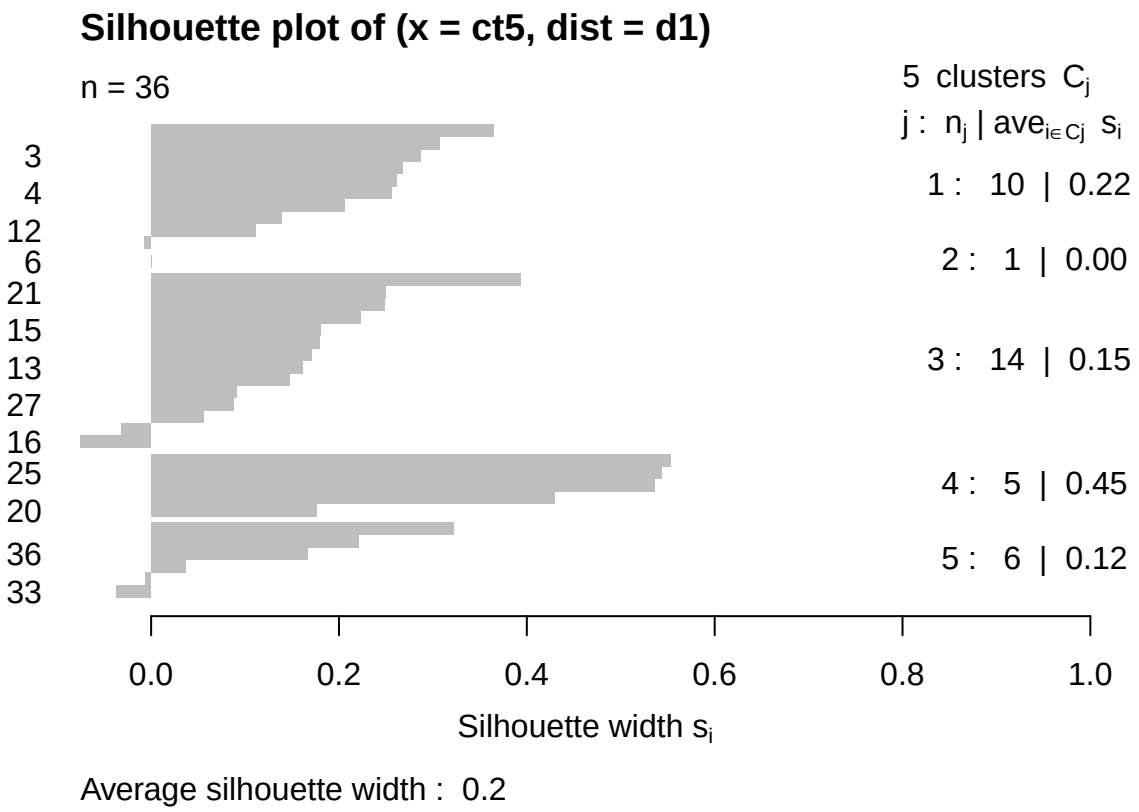



```
#do hierarchical clustering
d1 <- dist(codes_x)
hc1 <- hclust(d1, method = "complete")
plot(hc1)
```



d1
hclust (*, "complete")

```
ct5 <- cutree(hc1, k = 5)
si5 <- silhouette(ct5, dist = d1)
plot(si5)
```




```
#Plot SOM Cluster
graphics.off()
som_cluster1 <- cutree(hc1, h = 5)
# plot the SOM with the found clusters
my_pal <- c("red", "blue","orange","green","yellow")
my_bhcol <- my_pal[som_cluster1]

plot(state.x77.som, type = "mapping", col = "black", bgcol = my_bhcol)
add.cluster.boundaries(state.x77.som, som_cluster1)
```

show the som clustering here..

Observations:

From the silhouette plots we could observe that K = 5 has better clustering Because it has uniform distribution of datapoints between the clusters and has uniform pattern of clusters and also has less negative values

I think SOM also did a good job in clustering and if we see the optimal clusters given by SOM is 4 and the one cluster is just because of an outlier which is accurately reported in silhoutte, SOM and hirerachical clustering.

The number of datapoints in each cluster in SOM are pretty uniform. And also as SOM is converting the high dimensional data low dimensional data it is faster here.

(b) Build a Gaussian Graphical Model using the Graphical Lasso for the 8 predictors mentioned in Part A. What do you find for different penalties, and how does it compliment (and/or contradict) your results in part A

```
state.x77cov <- cov.wt(state.x77.scaled, method = "ML")
```

```
S <- state.x77cov$cov
m0.lasso <- glasso(S, rho = 0.6) ## Regularization parameter
my.edges <- m0.lasso$wi != 0
diag(my.edges) <- 0
g.lasso <- as(my.edges, "graphNEL")
nodes(g.lasso) <- colnames(state.x77.scaled)
```

```
# Estimate the precision matrix using graphical lasso
ggm_model <- huge(state.x77.scaled, method = "glasso", lambda = 1)
```

Answer:

```
##
## Conducting the graphical lasso (glasso)....done.
```

```
# Access the estimated precision matrix
precision_matrix <- ggm_model$omega
```

observations:

By performing both clustering using SOM and building a Gaussian Graphical Model, I gained insights into the relationships between the variables and identified patterns in the data.

Alabama:

Population: 3615

Income: 3624

Illiteracy: 2.1

Life Exp: 69.05

Murder: 15.1

HS Grad: 41.3

Frost: 20

Area: 50708

California:

Population: 21198

Income: 5114

Illiteracy: 1.1

Life Exp: 71.71

Murder: 10.3

HS Grad: 62.6

Frost: 20

Area: 156361

New York:

Population: 18076

Income: 2448

Illiteracy: 0.7

Life Exp: 70.55

Murder: 10.9

HS Grad: 52.7

Frost: 166

Area: 47831

These observations represent three different states and their corresponding values for the given dataset variable. By analyzing the relationships between these variables and comparing them across different states, By using two different clustering we got insights into the socio-economic characteristics of each state and identified any patterns or trends present in the dataset.