Multivariate Analysis Assignment 1

Harshad Kumar Elangovan - 19200349

29/02/2020

Loading the Glass dataset and excluding a random row using the below code block.

```
glassData<-read.csv("Glass.csv",header = T)
set.seed(19200349)

num=floor(runif(1,min=1,max =nrow(glassData)))

glassData<-glassData[-num,]
```

Question1

1 (a) (i) Compute the correlation matrix R of the 13 chemical elements using the fact that R = D-1/2SD-1/2 where S is the sample covariance matrix and D-1/2 is a diagonal matrix with the inverse of each variable’s standard deviation on the diagonal. Ensure your calculation is correct by showing that your result and that computed by the cor function in R are equal.

```
data1 <- glassData[,-1]
S <-c()
D<-c()
for (i in 1:ncol(data1)) {
  colmeans1<-mean(data1[,i])
  for (j in 1:ncol(data1)) {
    colmeans2<-mean(data1[,j])
    sum1 <- 0
    for (a in 1:nrow(data1)) {
      sum1<- sum1 + ((data1[a,i]-colmeans1)*(data1[a,j]-colmeans2))
    }
    S<-c(S,sum1/(nrow(data1)-1))
    if(i==j){
      D<-c(D,1/sqrt(sum1/(nrow(data1)-1)))
    }
    else{
      D<-c(D,0)
    }

  }

}
Covariance <- matrix(S,nrow = 13,ncol = 13,byrow = T)
Diagonal <- matrix(D,nrow = 13,ncol = 13,byrow = T)
Correlation<-Diagonal%*%Covariance%*%Diagonal
cat("The computed correlation is ")
```

```
## The computed correlation is
```

```
print(Correlation)
```

```
##             [,1]         [,2]        [,3]        [,4]        [,5]         [,6]
##  [1,]  1.0000000 -0.298446708 -0.66321261  0.45079249 -0.62770292  0.323034508
##  [2,] -0.2984467  1.000000000  0.43127798 -0.82251663  0.56965876 -0.008008344
##  [3,] -0.6632126  0.431277981  1.00000000 -0.59240794  0.83670451 -0.146552517
##  [4,]  0.4507925 -0.822516629 -0.59240794  1.00000000 -0.68500822  0.219644224
##  [5,] -0.6277029  0.569658764  0.83670451 -0.68500822  1.00000000 -0.195056285
##  [6,]  0.3230345 -0.008008344 -0.14655252  0.21964422 -0.19505628  1.000000000
##  [7,]  0.7288715 -0.227007213 -0.53033417  0.30562921 -0.31990608 -0.109956855
##  [8,] -0.6195952 -0.141797946  0.06235079 -0.08514607 -0.07547531 -0.254552904
##  [9,] -0.5784374  0.803944711  0.72890902 -0.88512970  0.84473530 -0.218354005
## [10,] -0.4823594  0.296146318  0.76338540 -0.48284109  0.81207592 -0.229465519
## [11,] -0.5248654  0.615676878  0.73078595 -0.69186543  0.68119125 -0.189636156
## [12,] -0.5741345  0.275189445  0.86847368 -0.40355179  0.73038608 -0.079187476
## [13,] -0.3693876 -0.196324352 -0.10435520  0.00131025 -0.11347285 -0.211578849
##             [,7]          [,8]       [,9]      [,10]      [,11]         [,12]
##  [1,]  0.7288715 -0.6195952052 -0.5784374 -0.4823594 -0.5248654 -0.5741345248
##  [2,] -0.2270072 -0.1417979459  0.8039447  0.2961463  0.6156769  0.2751894447
##  [3,] -0.5303342  0.0623507896  0.7289090  0.7633854  0.7307860  0.8684736832
##  [4,]  0.3056292 -0.0851460673 -0.8851297 -0.4828411 -0.6918654 -0.4035517890
##  [5,] -0.3199061 -0.0754753070  0.8447353  0.8120759  0.6811912  0.7303860828
##  [6,] -0.1099569 -0.2545529041 -0.2183540 -0.2294655 -0.1896362 -0.0791874763
##  [7,]  1.0000000 -0.6593611695 -0.2883574 -0.2855134 -0.3986842 -0.4413047212
##  [8,] -0.6593612  1.0000000000 -0.0857766 -0.0692623  0.1000219 -0.0004795419
##  [9,] -0.2883574 -0.0857765997  1.0000000  0.5984421  0.6964855  0.5804186655
## [10,] -0.2855134 -0.0692622996  0.5984421  1.0000000  0.6057412  0.7215256619
## [11,] -0.3986842  0.1000219156  0.6964855  0.6057412  1.0000000  0.5884884689
## [12,] -0.4413047 -0.0004795419  0.5804187  0.7215257  0.5884885  1.0000000000
## [13,] -0.2390017  0.5024734548 -0.1140585 -0.1476340 -0.1383293 -0.0991270358
##             [,13]
##  [1,] -0.36938756
##  [2,] -0.19632435
##  [3,] -0.10435520
##  [4,]  0.00131025
##  [5,] -0.11347285
##  [6,] -0.21157885
##  [7,] -0.23900165
##  [8,]  0.50247345
##  [9,] -0.11405849
## [10,] -0.14763403
## [11,] -0.13832929
## [12,] -0.09912704
## [13,]  1.00000000
```

```
cat("Correlation computed using R code is ")
```

```
## Correlation computed using R code is
```

```
RCorrelation<-cor(data1)
print(RCorrelation)
```

```
##             Na2O          MgO       Al2O3        SiO2        P2O5          SO3
## Na2O   1.0000000 -0.298446708 -0.66321261  0.45079249 -0.62770292  0.323034508
## MgO   -0.2984467  1.000000000  0.43127798 -0.82251663  0.56965876 -0.008008344
## Al2O3 -0.6632126  0.431277981  1.00000000 -0.59240794  0.83670451 -0.146552517
## SiO2   0.4507925 -0.822516629 -0.59240794  1.00000000 -0.68500822  0.219644224
## P2O5  -0.6277029  0.569658764  0.83670451 -0.68500822  1.00000000 -0.195056285
## SO3    0.3230345 -0.008008344 -0.14655252  0.21964422 -0.19505628  1.000000000
## Cl     0.7288715 -0.227007213 -0.53033417  0.30562921 -0.31990608 -0.109956855
## K2O   -0.6195952 -0.141797946  0.06235079 -0.08514607 -0.07547531 -0.254552904
## CaO   -0.5784374  0.803944711  0.72890902 -0.88512970  0.84473530 -0.218354005
## MnO   -0.4823594  0.296146318  0.76338540 -0.48284109  0.81207592 -0.229465519
## Fe2O3 -0.5248654  0.615676878  0.73078595 -0.69186543  0.68119125 -0.189636156
## BaO   -0.5741345  0.275189445  0.86847368 -0.40355179  0.73038608 -0.079187476
## PbO   -0.3693876 -0.196324352 -0.10435520  0.00131025 -0.11347285 -0.211578849
##               Cl           K2O        CaO        MnO      Fe2O3           BaO
## Na2O   0.7288715 -0.6195952052 -0.5784374 -0.4823594 -0.5248654 -0.5741345248
## MgO   -0.2270072 -0.1417979459  0.8039447  0.2961463  0.6156769  0.2751894447
## Al2O3 -0.5303342  0.0623507896  0.7289090  0.7633854  0.7307860  0.8684736832
## SiO2   0.3056292 -0.0851460673 -0.8851297 -0.4828411 -0.6918654 -0.4035517890
## P2O5  -0.3199061 -0.0754753070  0.8447353  0.8120759  0.6811912  0.7303860828
## SO3   -0.1099569 -0.2545529041 -0.2183540 -0.2294655 -0.1896362 -0.0791874763
## Cl     1.0000000 -0.6593611695 -0.2883574 -0.2855134 -0.3986842 -0.4413047212
## K2O   -0.6593612  1.0000000000 -0.0857766 -0.0692623  0.1000219 -0.0004795419
## CaO   -0.2883574 -0.0857765997  1.0000000  0.5984421  0.6964855  0.5804186655
## MnO   -0.2855134 -0.0692622996  0.5984421  1.0000000  0.6057412  0.7215256619
## Fe2O3 -0.3986842  0.1000219156  0.6964855  0.6057412  1.0000000  0.5884884689
## BaO   -0.4413047 -0.0004795419  0.5804187  0.7215257  0.5884885  1.0000000000
## PbO   -0.2390017  0.5024734548 -0.1140585 -0.1476340 -0.1383293 -0.0991270358
##               PbO
## Na2O  -0.36938756
## MgO   -0.19632435
## Al2O3 -0.10435520
## SiO2   0.00131025
## P2O5  -0.11347285
## SO3   -0.21157885
## Cl    -0.23900165
## K2O    0.50247345
## CaO   -0.11405849
## MnO   -0.14763403
## Fe2O3 -0.13832929
## BaO   -0.09912704
## PbO    1.00000000
```

So, the output of both the variables are same and shows that the computation is correct.

a.

   ii. Using R, determine the first two eigenvalues and eigenvectors of the covariance matrix S. Using R, show that they are indeed eigenvalues and eigenvectors of S.

```
eigendata <- eigen(Covariance)
#Determining the first two eigen values and eigen vactors
eigendata$values[1:2]
```

```
## [1] 43.42167 18.36155
```

```
eigendata$vectors[,c(1,2)]
```

```
##               [,1]          [,2]
##  [1,]  0.523503264  5.633375e-01
##  [2,] -0.089927759  8.422879e-02
##  [3,] -0.072929073  4.370222e-03
##  [4,]  0.537937789 -2.884352e-01
##  [5,] -0.083531208  2.997705e-02
##  [6,]  0.003599120  3.346908e-03
##  [7,]  0.017145643  2.829115e-02
##  [8,] -0.133147751 -6.735031e-01
##  [9,] -0.629625094  3.508758e-01
## [10,] -0.033880023  1.073583e-02
## [11,] -0.016252608  3.673247e-03
## [12,] -0.008292093  9.010358e-05
## [13,] -0.015504256 -1.174022e-01
```

```
#Verification
Covariance%*%eigendata$vectors[,1]
```

```
##              [,1]
##  [1,]  22.7313846
##  [2,]  -3.9048132
##  [3,]  -3.1667020
##  [4,]  23.3581557
##  [5,]  -3.6270643
##  [6,]   0.1562798
##  [7,]   0.7444924
##  [8,]  -5.7814974
##  [9,] -27.3393714
## [10,]  -1.4711271
## [11,]  -0.7057153
## [12,]  -0.3600565
## [13,]  -0.6732206
```

```
matrix(eigendata$values[1]%*%eigendata$vectors[,1])
```

```
##              [,1]
##  [1,]  22.7313846
##  [2,]  -3.9048132
##  [3,]  -3.1667020
##  [4,]  23.3581557
##  [5,]  -3.6270643
##  [6,]   0.1562798
##  [7,]   0.7444924
##  [8,]  -5.7814974
##  [9,] -27.3393714
## [10,]  -1.4711271
## [11,]  -0.7057153
## [12,]  -0.3600565
## [13,]  -0.6732206
```

```
#Both the output are same. So, it proves that they are eigen values and eigen vecto
rs of S.

Covariance%*%eigendata$vectors[,2]
```

```
##                [,1]
##  [1,]  10.343746453
##  [2,]   1.546570726
##  [3,]   0.080244025
##  [4,]  -5.296115428
##  [5,]   0.550424961
##  [6,]   0.061454410
##  [7,]   0.519469223
##  [8,] -12.366557425
##  [9,]   6.442621023
## [10,]   0.197126466
## [11,]   0.067446491
## [12,]   0.001654441
## [13,]  -2.155685528
```

```
matrix(eigendata$values[2]%*%eigendata$vectors[,2])
```

```
##                [,1]
##  [1,]  10.343746453
##  [2,]   1.546570726
##  [3,]   0.080244025
##  [4,]  -5.296115428
##  [5,]   0.550424961
##  [6,]   0.061454410
##  [7,]   0.519469223
##  [8,] -12.366557425
##  [9,]   6.442621023
## [10,]   0.197126466
## [11,]   0.067446491
## [12,]   0.001654441
## [13,]  -2.155685528
```

a.

iii. Using R, verify that the first two eigenvectors are orthonormal.

A set of vectors is defined orthogonal when u.uj = 0 for i <> j

and it is defined orthonormal when also ui.uj= 1 for i=j.

```
eigendata$vectors[,1]
```

```
##  [1]  0.523503264 -0.089927759 -0.072929073  0.537937789 -0.083531208
##  [6]  0.003599120  0.017145643 -0.133147751 -0.629625094 -0.033880023
## [11] -0.016252608 -0.008292093 -0.015504256
```

```
t(eigendata$vectors[,1])%*%eigendata$vectors[,1]
```

```
##      [,1]
## [1,]    1
```

```
t(eigendata$vectors[,2])%*%eigendata$vectors[,2]
```

```
##      [,1]
## [1,]    1
```

```
t(eigendata$vectors[,1])%*%(eigendata$vectors[,2])
```

```
##             [,1]
## [1,] 2.33754e-16
```

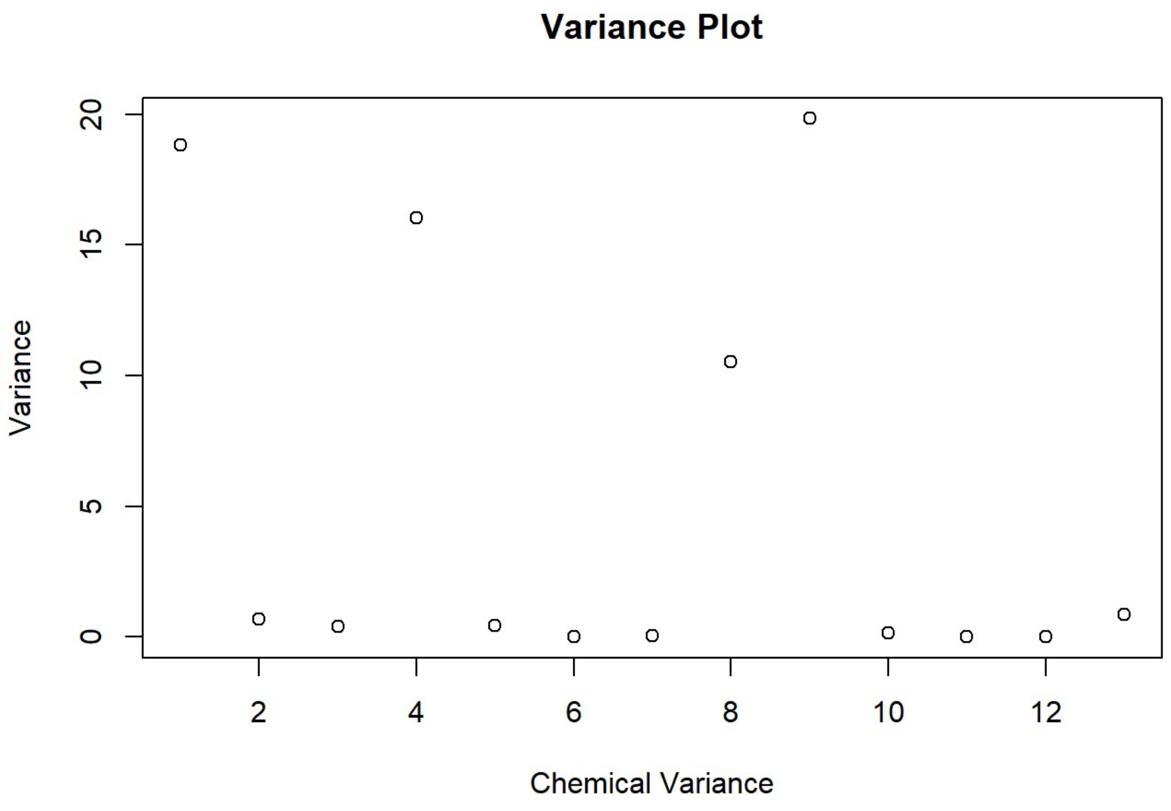
```
t(eigendata$vectors[,2])%*%eigendata$vectors[,1]
```

```
##              [,1]
## [1,] 2.33754e-16
```

a.
   iv. Compute the variance of each element and produce a suitable summarising plot. Would you advise standardizing the glass data prior to analysis? Explain your reasoning.

The variance can be computed from the co-variance matrix.

```
#The diagonal elements of S are the variance and that can be extracted to a variabl
e for plotting.
variance<-diag(Covariance)
plot(variance,xlab = "Chemical Variance",ylab = "Variance",main = "Variance Plot")
```



From the plot, we can see that there are few abnormal points in the graph. From the data aswell, we can see that there are few columns which has high values comparatively to other columns. So we will have to standardize the dataset before creating clusters for an unbiased result.

1 (b) Suppose E[X1] = 5 and V ar[X1] = 6. Suppose E[X2] = 8 and V ar[X2] = 7. Suppose also that the covariance between X1 and X2 is 2.5.

i. In R, calculate the expected value and variance of X2???X1.

```
#Given Data Initialization
EX1<-5
EX2<-8
VarX1<-6
VarX2<-7
CovX1X2<-2.5

#Expected value of X2-X1 is EX2-EX1
EX2X1<-EX2-EX1

#Variance of X2-X1 is VX2+VX1-2CovX1X2
VarX2X1<- VarX1 + VarX2 -(2*CovX1X2)

cat("Expected Value of X2-X1 is ",EX2X1)
```

```
## Expected Value of X2-X1 is  3
```

```
cat("Variance of X2-X1 is ",VarX2X1)
```

```
## Variance of X2-X1 is  8
```

b.

ii. In R, calculate the correlation of U = X2-X1 and V = X2-2X1.

```
#The Expected value and variance of U=X2-X1 is
cat("Expected Value and Variance of X2-X1 is ",EX2X1,"and",VarX2X1)
```

```
## Expected Value and Variance of X2-X1 is  3 and 8
```

```
#Expected value and Variance of V=X2-2X1
EX2_2X1 <-EX2-(2*EX1)
VarX2_2X1<- VarX2 + (4*VarX1) - (4*CovX1X2)
cat("Expected Value and Variance of X2-2X1 is ",EX2_2X1,"and",VarX2_2X1)
```

```
## Expected Value and Variance of X2-2X1 is  -2 and 21
```

```
#We have to find the covariance(X2-X1,X2-2X1) to calculate correlation by multiplyi
ng the matrix (1,-1)*(7,2.5,2.5,6)*t(1,-2)

totalcov<-matrix(c(1,-1),nrow=1,ncol = 2,byrow = T)%*%matrix(c(7,2.5,2.5,6),nrow =
2,ncol = 2,byrow = T)%*%matrix(c(1,-2),nrow = 2,ncol = 1,byrow = T)

CorrectionUV<-totalcov/sqrt(VarX2_2X1*VarX2X1)
cat("The correlation of U & V is ",CorrectionUV)
```

```
## The correlation of U & V is  0.8872443
```

Question 2

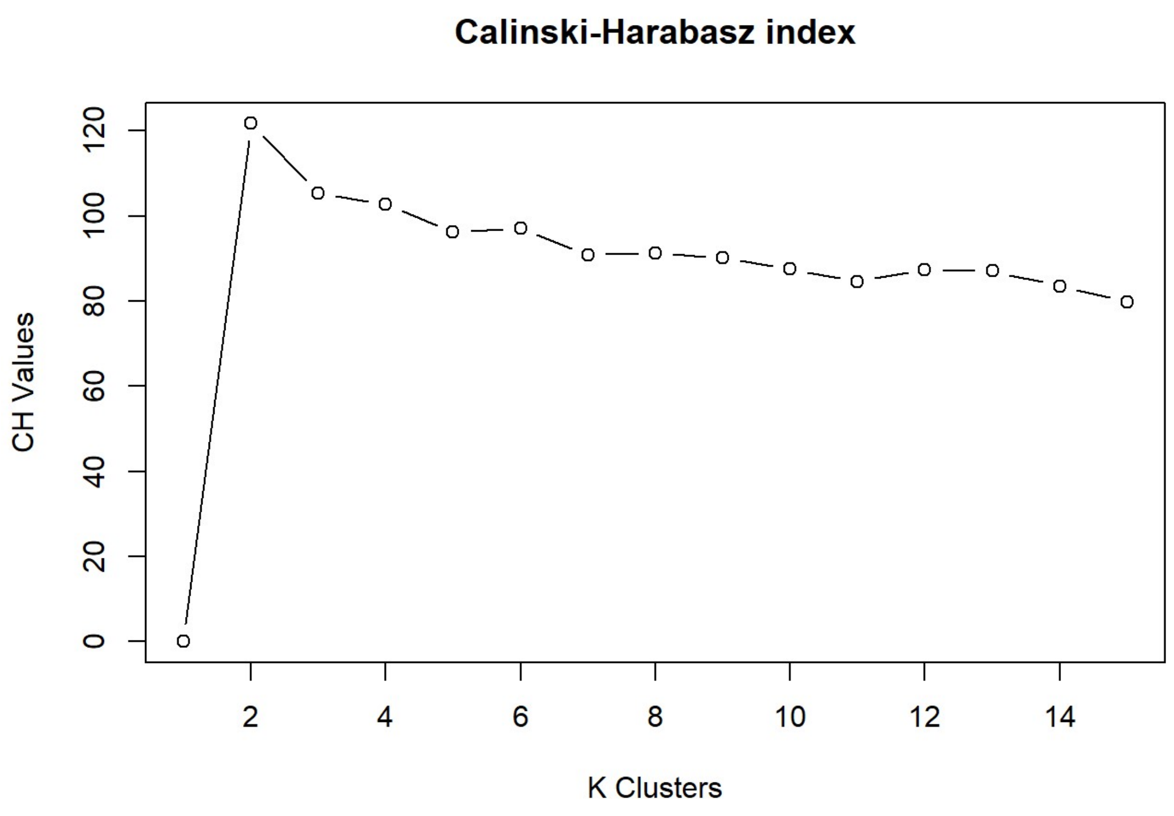
a. In R, cluster the glass vessels using k-means clustering. How many clusters would you suggest are present in this data set? Detail any decisions you make when running this procedure. Provide details of your reasoning and fully commented R code

As we pointed out in 1(a)(iv), we will have to standardize the dataset before working on clusters.

```
sdval<-apply(data1,2,sd)
meanval<-apply(data1,2,mean)
data2<-scale(data1,meanval,sdval)
```

Now we can use it for exploring clusters.

```
k<-15
wss<-bss<-rep(NA,k)
for (i in 1:k) {
  fit1 <- kmeans(data2,centers = i,nstart = 50)
  wss[i]<-fit1$tot.withinss
  bss[i]<-fit1$betweenss
}
#computing CH Index
N<-nrow(data1)
ch<-(bss/(1:k-1))/(wss/(N-1:k))
ch[1]<-0
plot(1:k,ch,type = "b",ylab = "CH Values",xlab = "K Clusters", main="Calinski-Harab
asz index")
```



From the plot, we can see that there is a shart point for CH values at K=2. After that, the line starts to decrease gradually with minimum upward. So there is an high possibility that there can be only two cluster for the given data. We can confirm this by comparing our dataset for clusters more than two.

```
fit2<- kmeans(data2,centers = 2,nstart = 50)
fit3<- kmeans(data2,centers = 3,nstart = 50)
fit4<- kmeans(data2,centers = 4,nstart = 50)

table(fit2$cluster,glassData$Group)
```

```
##
##       1   2   3   4
##   1 144  15   0   0
##   2   0   0  10  10
```

```
table(fit3$cluster,glassData$Group)
```

```
##
##       1   2   3   4
##   1   0   0  10  10
##   2   0  15   0   0
##   3 144   0   0   0
```

```
table(fit4$cluster,glassData$Group)
```

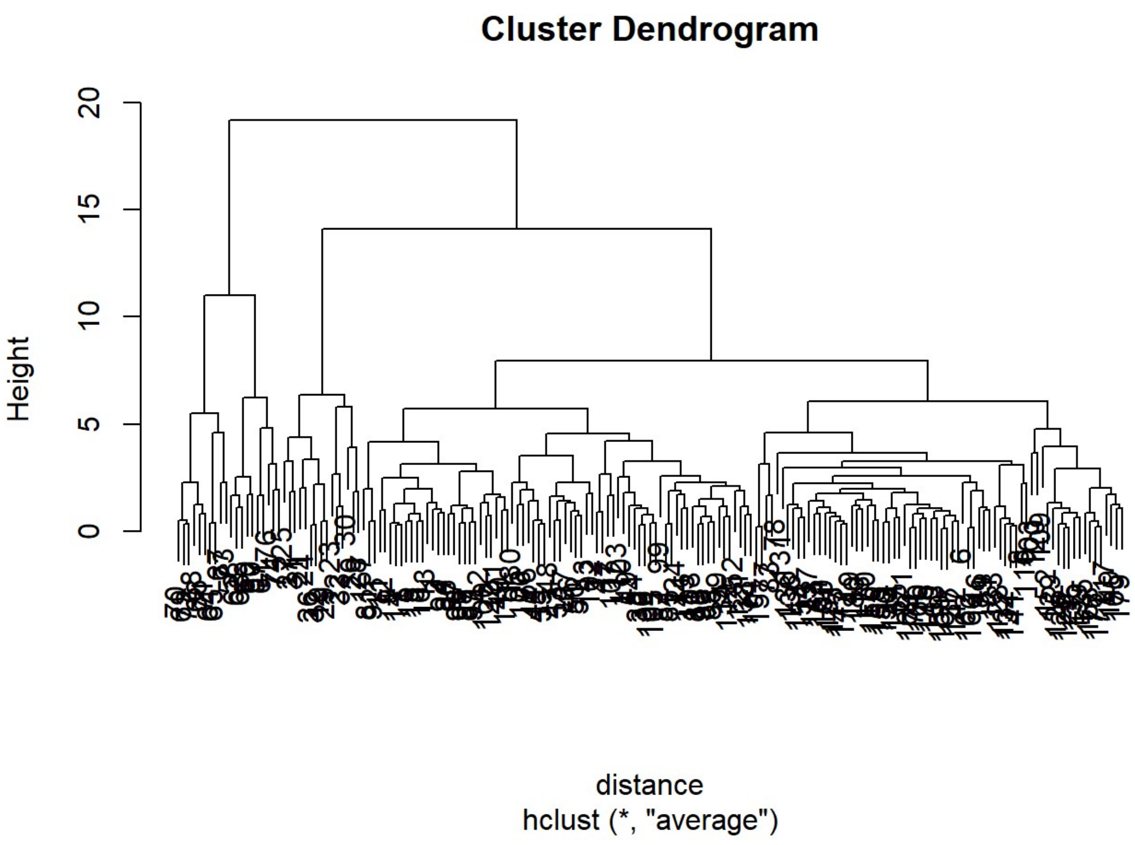
```
##
##       1  2  3  4
##    1 63  0  0  0
##    2  0 15  0  0
##    3  0  0 10 10
##    4 81  0  0  0
```

The table of cluster 2 also looks fine compared with tables ofother cluster(k=3,k=4).

b. Cluster the glass vessels using hierarchical clustering. Detail any decisions you make when conducting the hierarchical clustering. From the dendrogram, how many clusters would you suggest are present in this data set? Cut the dendrogram at the desired number of clusters.

First the distance between the data points is calculated using euclidean method and then the cluster are created using hiercharchical clustering.

```
distance<-dist(glassData,method = "euclidean")
c1.glassData<-hclust(distance,method = "average")
plot(c1.glassData)
```



```
#plot(c1.single)
#plot(c1.complete)
#c1.single<-hclust(distance,method = "single")
#c1.complete<-hclust(distance,method = "complete")
```

From the dendrogram taking the method for hierarchical clustering to “Average”, we can see that the dendrogram can be divided to four clusters.

```
hierarchical2<-cutree(c1.glassData,k=2)
table(hierarchical2,glassData$Group)
```

```
##
## hierarchical2   1   2   3   4
##             1 144  15   0   0
##             2   0   0  10  10
```

```
hierarchical4<-cutree(c1.glassData,k=4)
table(hierarchical4,glassData$Group)
```

```
##
## hierarchical4   1   2   3   4
##             1 144   0   0   0
##             2   0  15   0   0
##             3   0   0   0  10
##             4   0   0  10   0
```

c. Examine the cross tabulation of the cluster solutions obtained in (a) and (b) and then quantitatively compare them using an appropriate measure. Comment on the agreement between the two solutions. Provide details of your reasoning and fully commented R code.

```
#install.packages("e1071")
library("e1071")
```

```
## Warning: package 'e1071' was built under R version 3.6.2
```

```
#cross tabulation with 2 and four cluster
table2<-table(hierarchical2,fit2$cluster)
table4<-table(hierarchical4,fit4$cluster)
classAgreement(table2)
```

```
## $diag
## [1] 1
##
## $kappa
## [1] 1
##
## $rand
## [1] 1
##
## $crand
## [1] 1
```

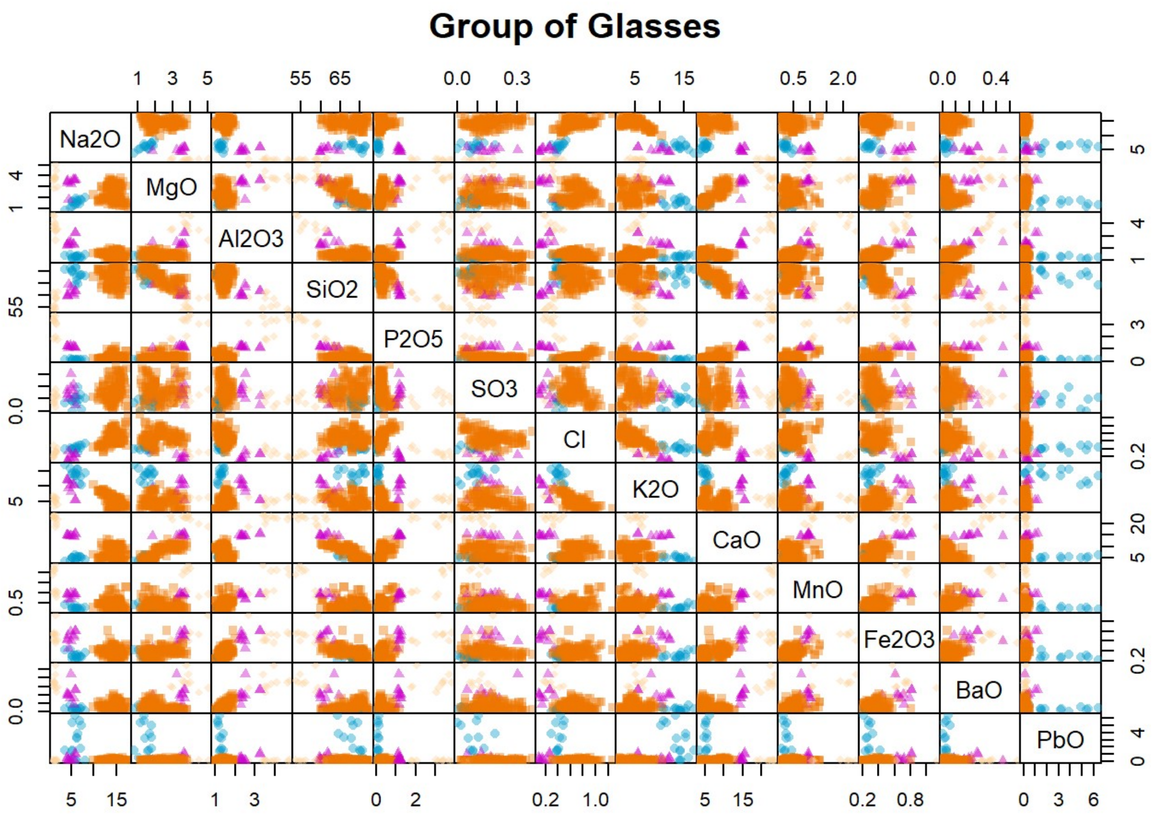
```
classAgreement(table4)
```

```
## $diag
## [1] 0.4916201
##
## $kappa
## [1] 0.2505291
##
## $rand
## [1] 0.6734041
##
## $crand
## [1] 0.4054396
```

From the agreement, cluster 2 has the highest agreement compared with the model with four clusters. The Rand Index and Adjusted Rand Index of model with cluster two is 1 and the the Rand Index and Adjusted Rand Index of model with cluster four has values 0.67 and 0.40 respectively. From this, we can take clusters k=2 on highest priority than k=4.

d. Create a pairs plot of the data, highlighting vessels from di???erent glass types using colour and/or plotting characters. Comment on the relative size of the ???rst glass type group and on the distribution of the PbO variable. Explore the impact of removing these data from your analysis. Why would detecing such issues be challenging in a truly unsupervised and high-dimensional setting?

```
symb<-c(15,16,17,18)
col<-c("darkorange2","deepskyblue3","magenta3","burlywood1")
plot(data1,gap=0,pch=symb[glassData$Group],col=adjustcolor(col[glassData$Group],0.
4),main="Group of Glasses")
```



From the plot, we can clearly see that one of the group is highly dominating the data since it is huge in numbers compared with other groups.

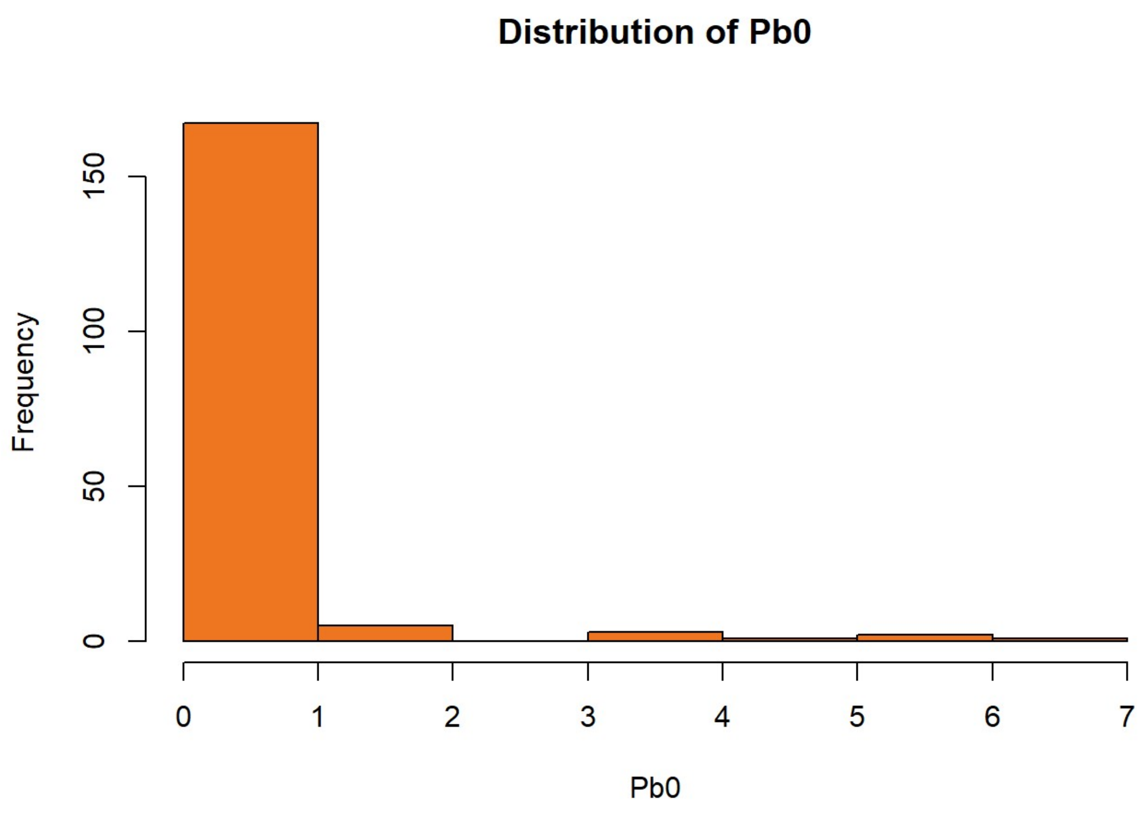
Relative size of the first group:

```
nrow(glassData[glassData$Group==1,])
```

```
## [1] 144
```

So, this group 1 has the highest data in the dataset and it was clearly visible in the above pair plot.

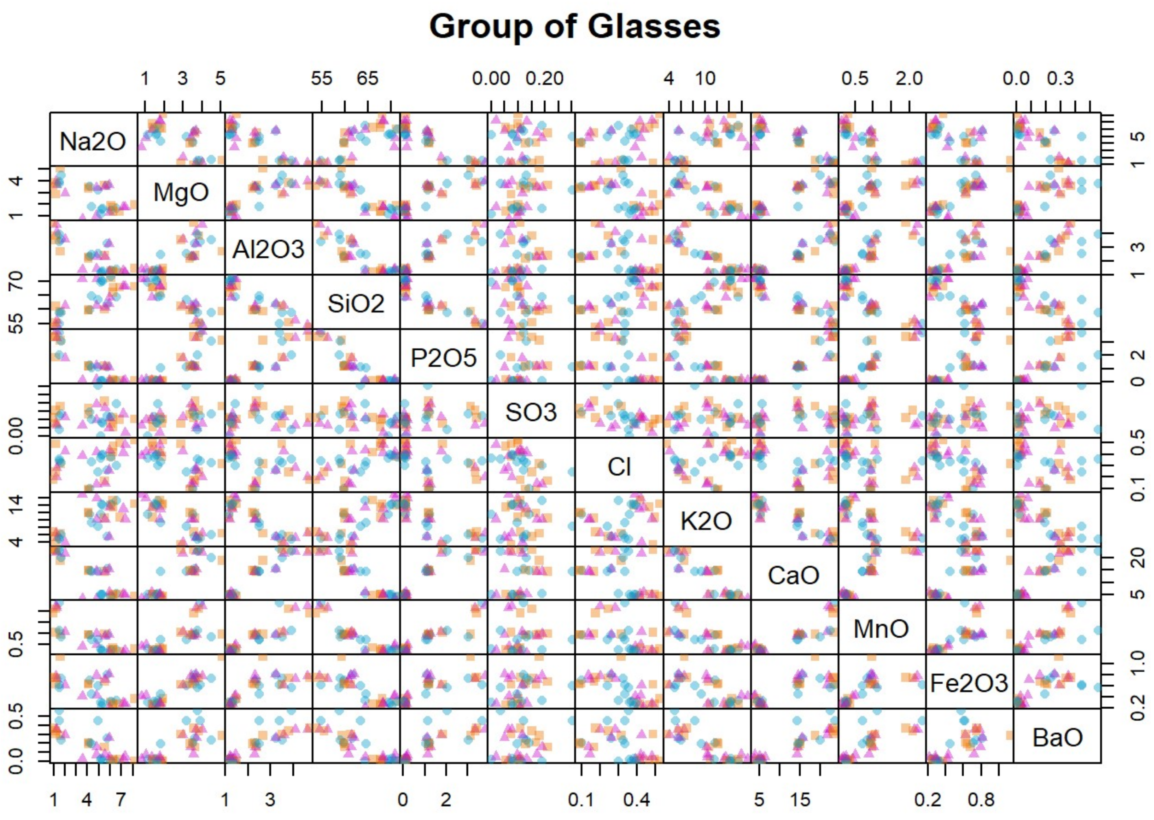
```
hist(glassData$PbO,xlab = "Pb0",ylab = "Frequency",col="chocolate2",main = "Distrib
ution of Pb0")
```



The histgram shows that majority of the values of Pb0 was in between 0 and 0.5 and the graph is positively skewed. After that there are small data values of Pb0.

Impact of removing group 1 and Pb0 data and analyzing the data again.

```
glassdata2<-data1[glassData$Group!="1",-13]
fitnew2<-kmeans(glassdata2,centers = 2,nstart = 50)
fitnew3<-kmeans(glassdata2,centers = 3,nstart = 50)
plot(glassdata2,gap=0,pch=symb[1:3],col=adjustcolor(col[1:3],0.4),main="Group of Gl
asses")
```



```
table(fitnew2$cluster,glassData[glassData$Group!="1","Group"])
```

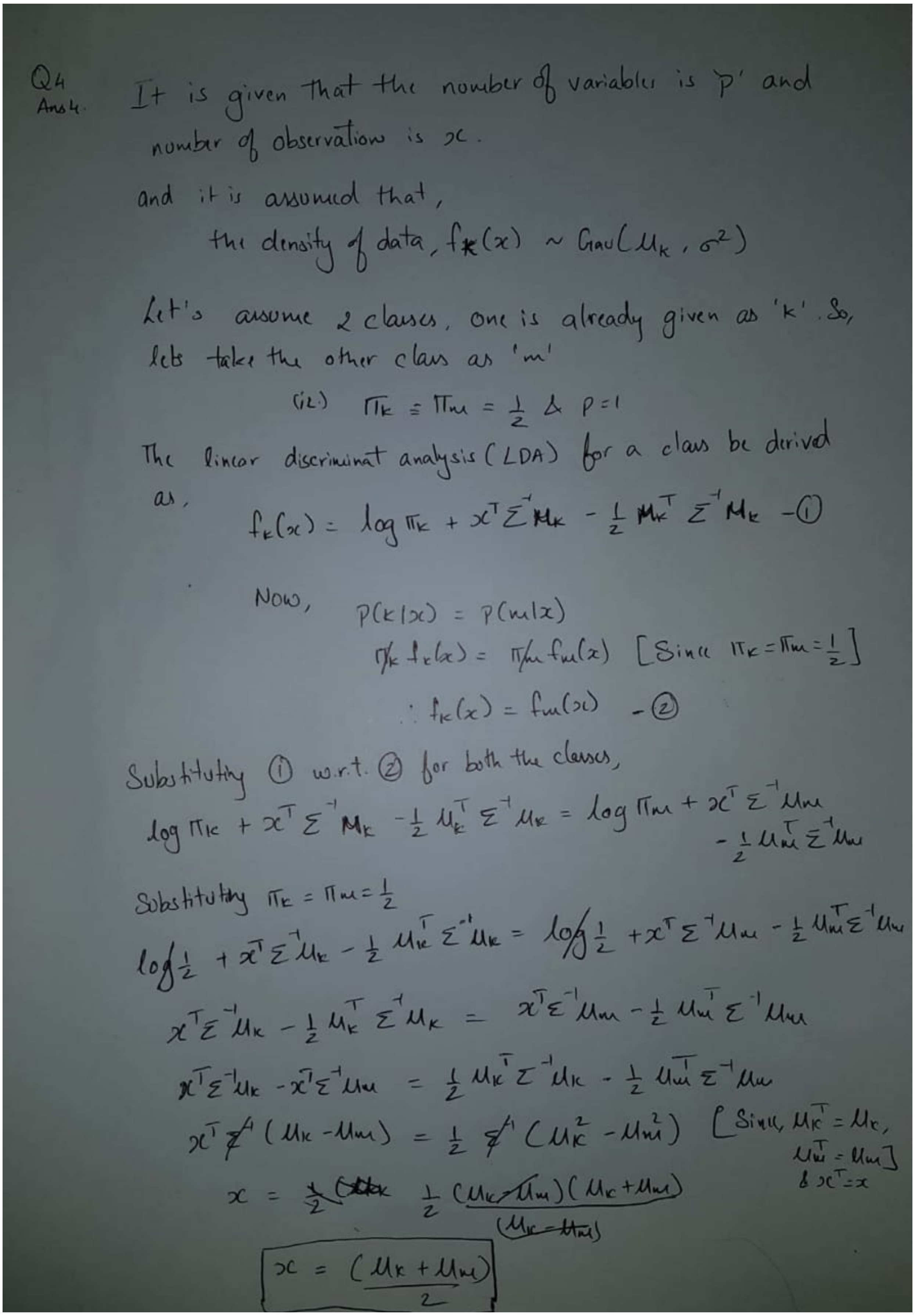
```
##
##       2  3  4
##    1  0 10 10
##    2 15  0  0
```

```
table(fitnew3$cluster,glassData[glassData$Group!="1","Group"])
```

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

From both the tables, we can see that the model with cluster 3 has better segregation of this new data and provides a clear view of the clusters after removing the Group1 and Pb0 variable. But,in a truly unsupervised and high-dimensional setting, we cannot confirm the actual groups and detecting the impact is very difficult for each variables.

Question 4



Q4 Ans4. It is given that the number of variables is 'p' and number of observation is x.

and it is assumed that,

the density of data, $f_k(x) \sim Gau(\mu_k, \sigma^2)$

Let's assume 2 classes, one is already given as 'k'. So, lets take the other class as 'm'

(i.e.) $\pi_k = \pi_m = \frac{1}{2}$ & $p = 1$

The linear discriminat analysis (LDA) for a class be derived as,

$$f_k(x) = \log \pi_k + x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k \quad -①$$

Now, $P(k|x) = P(m|x)$

$\cancel{\pi_k} f_k(x) = \cancel{\pi_m} f_m(x)$ [Since $\pi_k = \pi_m = \frac{1}{2}$]

$\therefore f_k(x) = f_m(x) \quad -②$

Substituting ① w.r.t. ② for both the classes,

$$\log \pi_k + x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k = \log \pi_m + x^T \Sigma^{-1} \mu_m - \frac{1}{2} \mu_m^T \Sigma^{-1} \mu_m$$

Substituting $\pi_k = \pi_m = \frac{1}{2}$

$$\cancel{\log \frac{1}{2}} + x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k = \cancel{\log \frac{1}{2}} + x^T \Sigma^{-1} \mu_m - \frac{1}{2} \mu_m^T \Sigma^{-1} \mu_m$$

$$x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k = x^T \Sigma^{-1} \mu_m - \frac{1}{2} \mu_m^T \Sigma^{-1} \mu_m$$

$$x^T \Sigma^{-1} \mu_k - x^T \Sigma^{-1} \mu_m = \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_m^T \Sigma^{-1} \mu_m$$

$$x^T \cancel{\Sigma^{-1}} (\mu_k - \mu_m) = \frac{1}{2} \cancel{\Sigma^{-1}} (\mu_k^2 - \mu_m^2)$$ [Since, $\mu_k^T = \mu_k$, $\mu_m^T = \mu_m$ & $x^T = x$]

$$x = \cancel{\frac{1}{2}} \; \frac{1}{2} \frac{(\cancel{\mu_k - \mu_m})(\mu_k + \mu_m)}{(\cancel{\mu_k - \mu_m})}$$

$$\boxed{x = \frac{(\mu_k + \mu_m)}{2}}$$

Answer4

Question 3

Load the MASS library in R, and its fgl dataset. Use the help file to understand what the data set contains. Decide whether or not you need to scale the data. Write your own function to calculate the misclassification error for linear discriminant analysis applied to the first 9 variables in this dataset, using the final variable as the known class. Split the data such that floor(n*(2/3)) observations are in the training set and the remainder in the test set, compute the misclassification rate, and repeat this 100 times. Create a suitable plot to illustrate the misclassification rates for each class and the overall misclassification rates. What is the average overall misclassification rate?

```
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 3.6.2
```

```
data("fgl")
set.seed(19200349)
num1=floor(runif(1,min=1,max =nrow(fgl)))
fgldata<-fgl[-num1,]
nrow(fgl)
```

```
## [1] 214
```

```
nrow(fgldata)
```

```
## [1] 213
```

```
#help("fgl")
```

There are few data points in the dataset which are abnormally high compared with other variables.So, we have to standardize the dataset for unbiased calculation.

```
sdval1<-apply(fgldata[,-10],2,sd)
fgldata1<-sweep(fgldata[-10],2,sdval1,"/")
fgldata1<-data.frame(cbind(fgldata1,fgldata[,10]))
colnames(fgldata1)<-colnames(fgldata)
```

The data is now standardized and is further used for computing misclassification rate. LDA command will produce the prior and mean values that will be used for computing the misclassification rate.

```r
misclasification<- function(dataval1,iteration){
  type<-c()
  for(i in 1:iteration){
    dataval<-sort(sample(nrow(dataval1),floor(nrow(data)*(2/3))))
    traindata<-dataval1[dataval,]
    testdata<-dataval1[-dataval,]
    covval<-c()
    lda.res<-lda(dataval1$type~dataval1$RI + dataval1$Na + dataval1$Mg + dataval1$A
l + dataval1$Si + dataval1$K + dataval1$Ca + dataval1$Ba + dataval1$Fe)
    for(m in unique(traindata$type)){
      dataval2<-traindata[traindata$type==m,]
      covval<-c(covval,list(cov(dataval2[,-10])*nrow(dataval2)))
    }
    wgt_cov<-matrix(rep(0,81),nrow = 9,ncol = 9)
    for (n in 1:length(unique(traindata$type))) {
      wgt_cov<-wgt_cov + covval[[n]]
    }
    wgt_cov<-wgt_cov/nrow(traindata)
  }
}
#fglclass=misclasification(fgldata1,100)
```