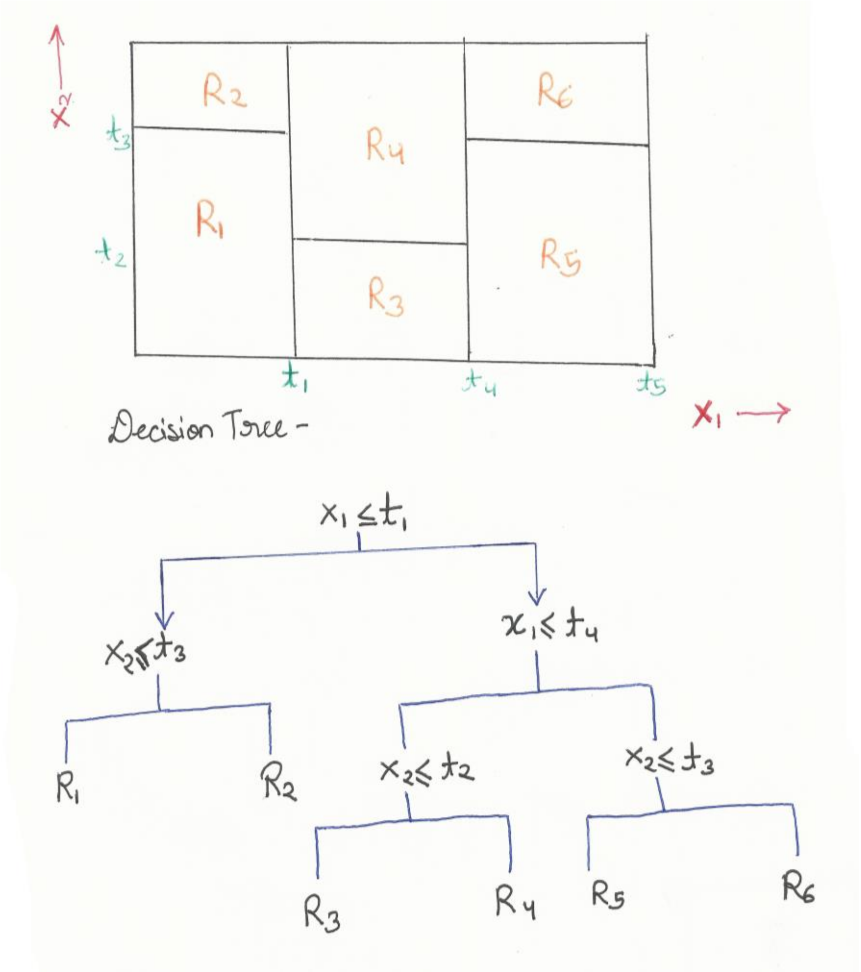
Assignment 4

1. Recitation Exercises:

1.1 Chapter 8

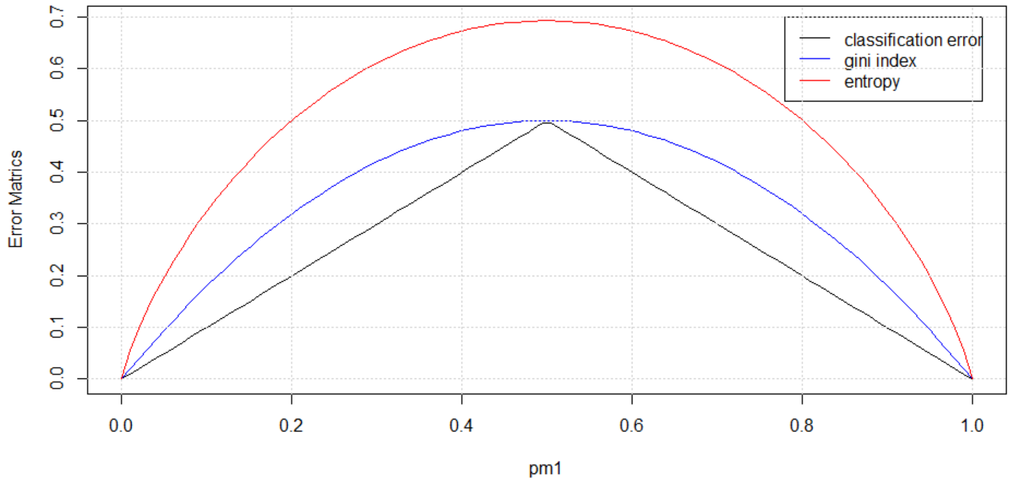
Que 1



Que 3

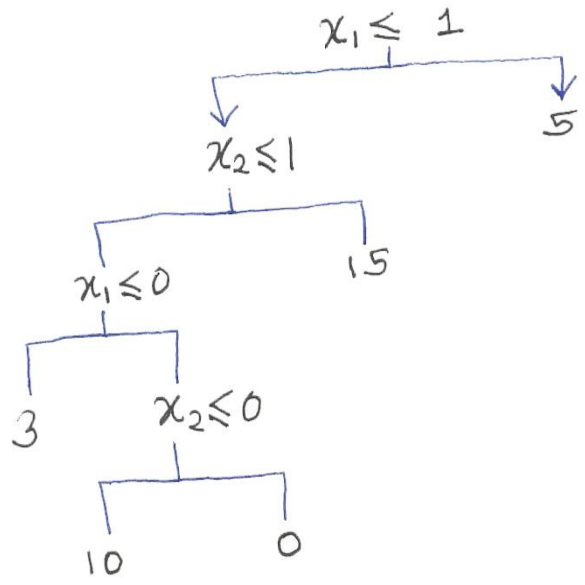
```
p1 = seq(0 + 1e-06, 1 - 1e-06,length.out = 100)
p2 = 1- p1

classification_error <- 1- apply(rbind(p1,p2),2, max)
gini_index<- p1 *(1 - p1) + p2 *(1 - p2)
entropy <- -(p1 * log(p1) + p2 * log(p2))
plot(p1, classification_error, type = "l", col="black",xlab = "pm1", ylab = "Error Matrics", ylim = c(min(c(classification_error,gini_index,entropy)),max(classification_error,gini_index,entropy)))
lines(p1,gini_index,col="blue")
lines(p1,entropy,col="red")
legend(0.78,0.7,c("classification error","gini index","entropy"),col = c("black","blue","red"),lty = c(1,1))
grid()
```

Assignment 4

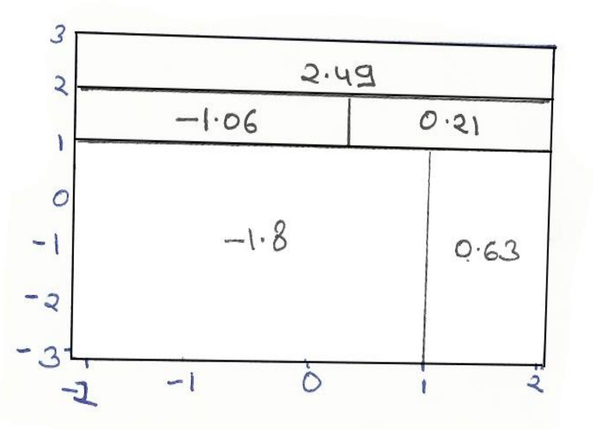


Que 4

a)



b)



Assignment 4

Que 5 Given:

We have two classes Red and Green

10 Estimates of P(Class is Red| X) are:

0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, .75

Different approaches to combine results are:

1. Majority Vote approach:

   We can see that out of 10 estimates, 6 estimates have p> 0.5 and 4 estimates have p<0.5, which suggest that majority of estimates classify X as Red.

2. Average Probability Approach:

   Here in this approach we will calculate the average of all the 10 estimates and depends on the result if the p is > 0.5 then class for X will be Red otherwise the class will be Green.

   $P = \frac{(0.1+0.15+0.2+0.2+0.55+0.6+0.6+0.65+0.7+0.75)}{10} = 0.45$

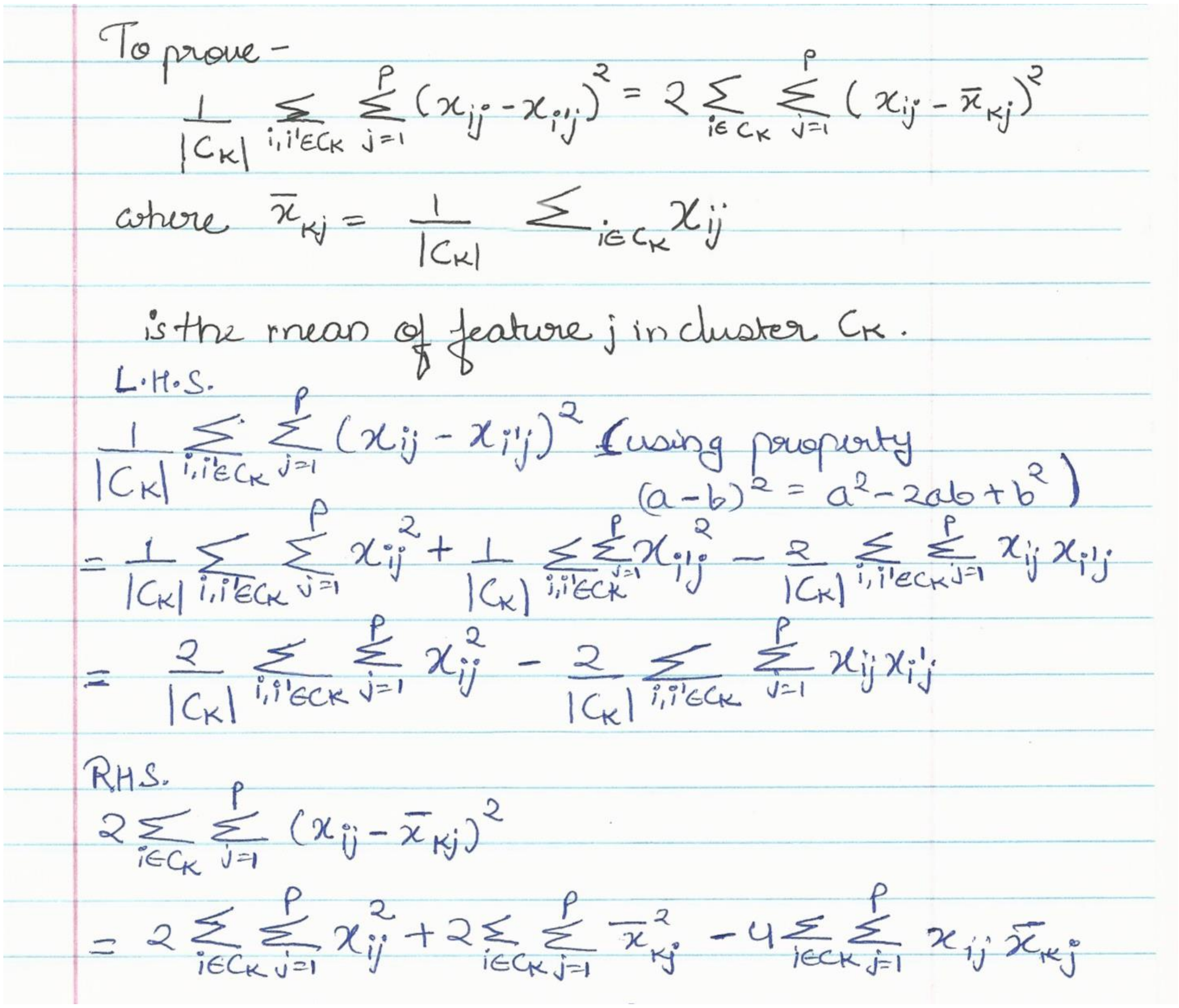
   Since the p<05

   ⇨ Class for X will be Green.

1.2 Chapter 10

Que 1

a)



To prove -

$$\frac{1}{|C_K|}\sum_{i,i'\in C_K}\sum_{j=1}^{p}(x_{ij}-x_{i'j})^2 = 2\sum_{i\in C_K}\sum_{j=1}^{p}(x_{ij}-\bar{x}_{Kj})^2$$

where $\bar{x}_{Kj} = \frac{1}{|C_K|}\sum_{i\in C_K} x_{ij}$

is the mean of feature j in cluster C_K.

L.H.S.

$$\frac{1}{|C_K|}\sum_{i,i'\in C_K}\sum_{j=1}^{p}(x_{ij}-x_{i'j})^2$$ (using property $(a-b)^2 = a^2-2ab+b^2$)

$$= \frac{1}{|C_K|}\sum_{i,i'\in C_K}\sum_{j=1}^{p}x_{ij}^2 + \frac{1}{|C_K|}\sum_{i,i'\in C_K}\sum_{j=1}^{p}x_{i'j}^2 - \frac{2}{|C_K|}\sum_{i,i'\in C_K}\sum_{j=1}^{p}x_{ij}x_{i'j}$$

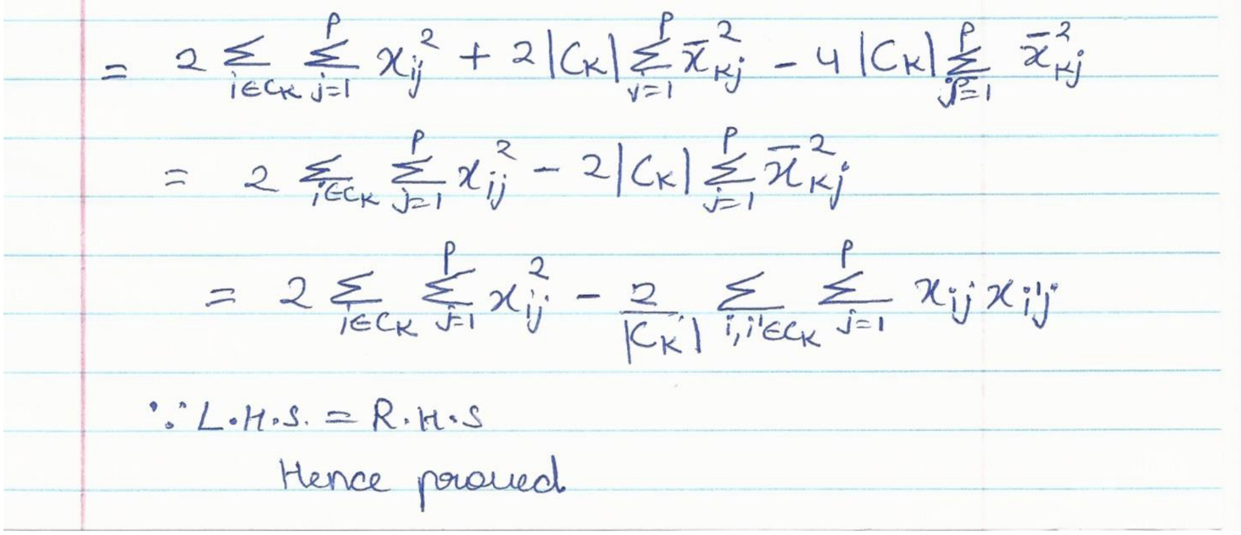
$$= \frac{2}{|C_K|}\sum_{i,i'\in C_K}\sum_{j=1}^{p}x_{ij}^2 - \frac{2}{|C_K|}\sum_{i,i'\in C_K}\sum_{j=1}^{p}x_{ij}x_{i'j}$$

RHS.

$$2\sum_{i\in C_K}\sum_{j=1}^{p}(x_{ij}-\bar{x}_{Kj})^2$$

$$= 2\sum_{i\in C_K}\sum_{j=1}^{p}x_{ij}^2 + 2\sum_{i\in C_K}\sum_{j=1}^{p}\bar{x}_{Kj}^2 - 4\sum_{i\in C_K}\sum_{j=1}^{p}x_{ij}\bar{x}_{Kj}$$

Assignment 4



$$= 2\sum_{i\in C_k}\sum_{j=1}^{p} x_{ij}^2 + 2|C_k|\sum_{j=1}^{p}\bar{x}_{kj}^2 - 4|C_k|\sum_{j=1}^{p}\bar{x}_{kj}^2$$

$$= 2\sum_{i\in C_k}\sum_{j=1}^{p} x_{ij}^2 - 2|C_k|\sum_{j=1}^{p}\bar{x}_{kj}^2$$

$$= 2\sum_{i\in C_k}\sum_{j=1}^{p} x_{ij}^2 - \frac{2}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p} x_{ij}x_{i'j}$$

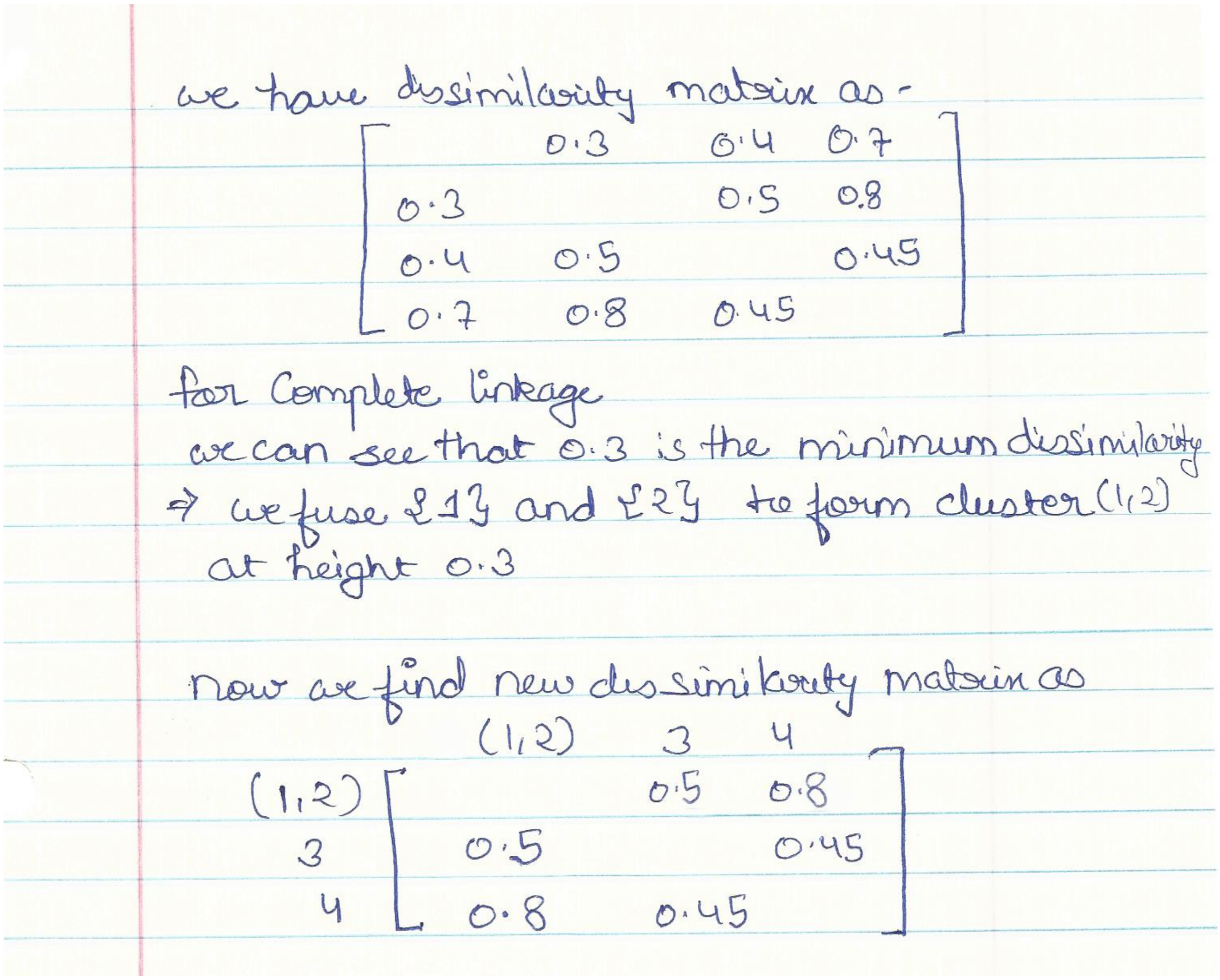
$\therefore$ L.H.S. = R.H.S

Hence proved

b) In K-means clustering algorithm, at each iteration, an observation is assigned to its nearest cluster. Due to which after each iteration the value of RHS will decreases as this quantity is sum of squared distance of each observation from the cluster mean. Hence, in this way the k-means will decrease the objective in each iteration.

Que 2

a)



we have dissimilarity matrix as -

$$\begin{bmatrix} & 0.3 & 0.4 & 0.7 \\ 0.3 & & 0.5 & 0.8 \\ 0.4 & 0.5 & & 0.45 \\ 0.7 & 0.8 & 0.45 & \end{bmatrix}$$

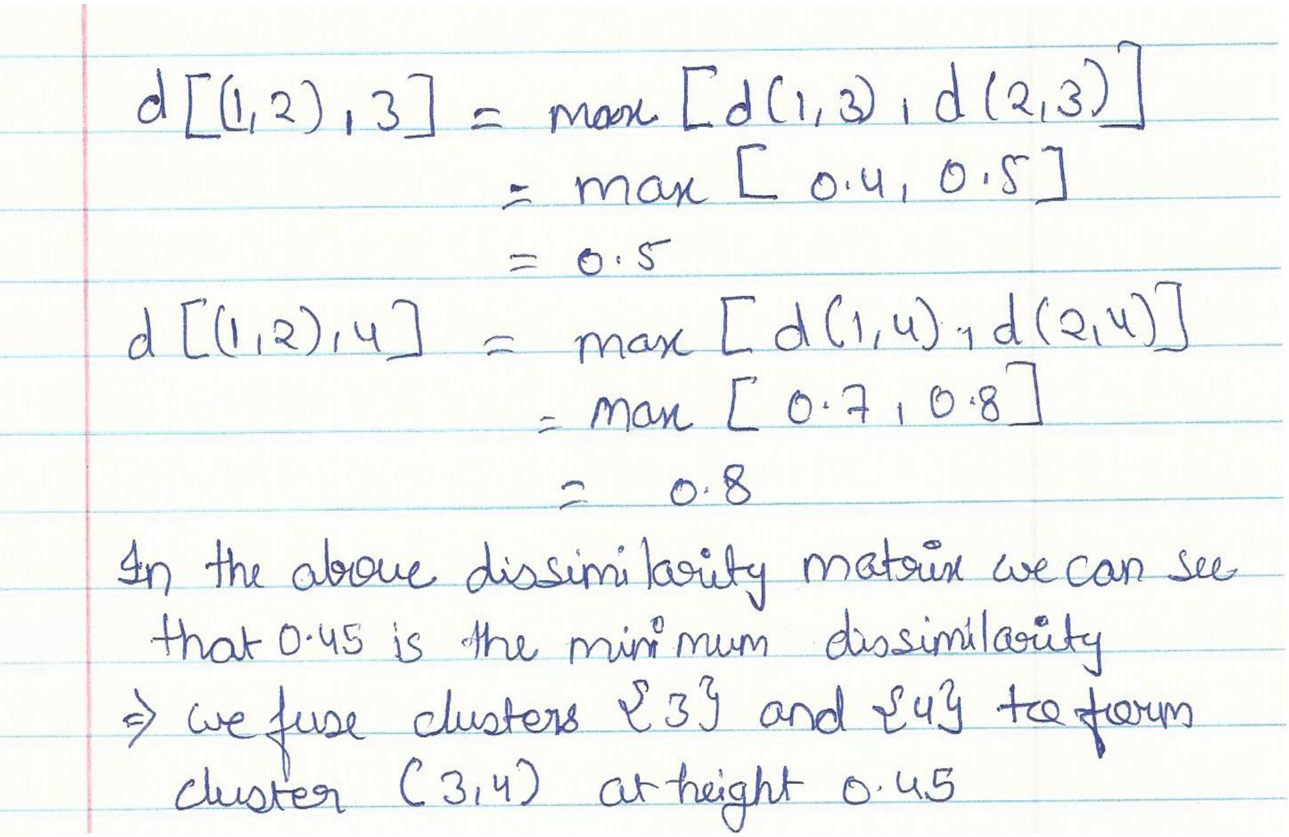
for Complete linkage

we can see that 0.3 is the minimum dissimilarity

⇒ we fuse {1} and {2} to form cluster (1,2) at height 0.3

now we find new dissimilarity matrix as

$$\begin{array}{c|ccc} & (1,2) & 3 & 4 \\ \hline (1,2) & & 0.5 & 0.8 \\ 3 & 0.5 & & 0.45 \\ 4 & 0.8 & 0.45 & \end{array}$$

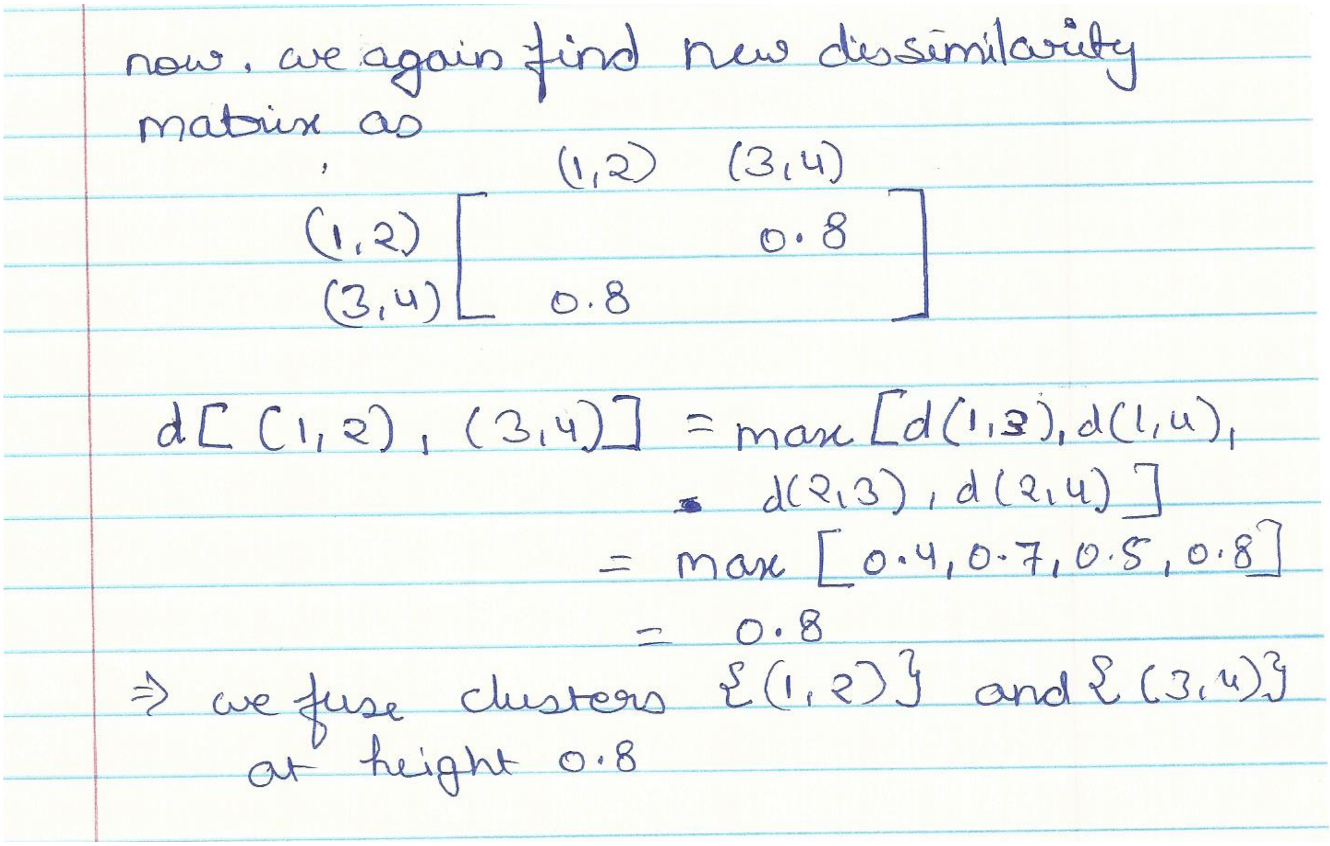


$$d[(1,2),3] = \max[d(1,3), d(2,3)]$$
$$= \max[0.4, 0.5]$$
$$= 0.5$$

$$d[(1,2),4] = \max[d(1,4), d(2,4)]$$
$$= \max[0.7, 0.8]$$
$$= 0.8$$

In the above dissimilarity matrix we can see that 0.45 is the minimum dissimilarity

⇒ we fuse clusters {3} and {4} to form cluster (3,4) at height 0.45



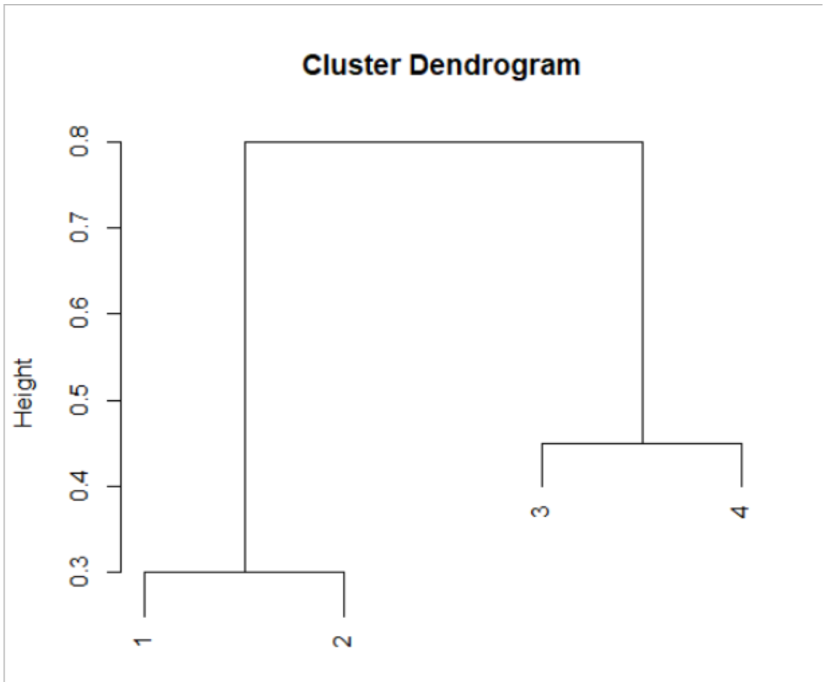
now, we again find new dissimilarity matrix as

$$\begin{array}{c|cc} & (1,2) & (3,4) \\ \hline (1,2) & & 0.8 \\ (3,4) & 0.8 & \end{array}$$

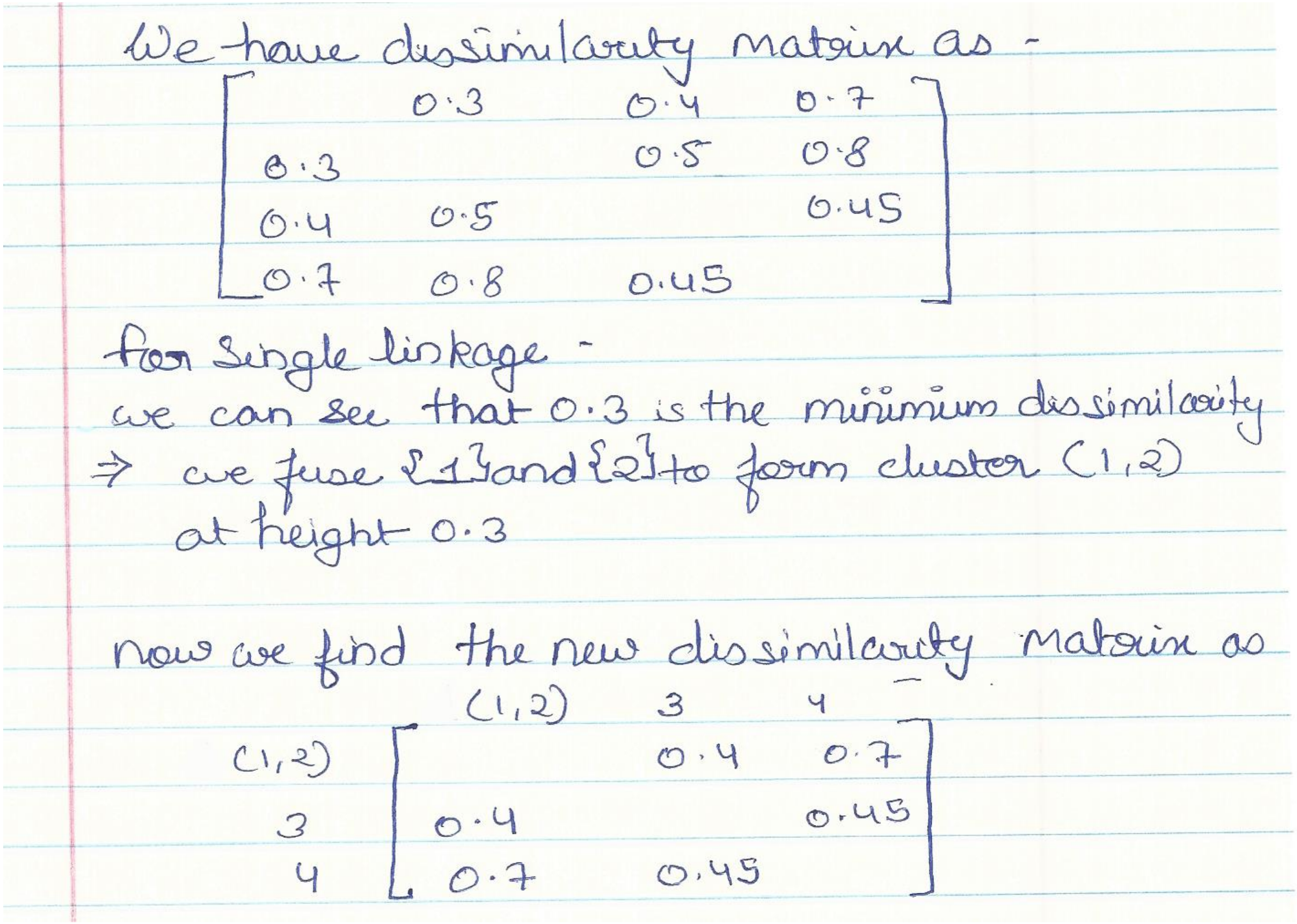
$$d[(1,2),(3,4)] = \max[d(1,3), d(1,4), d(2,3), d(2,4)]$$
$$= \max[0.4, 0.7, 0.5, 0.8]$$
$$= 0.8$$

⇒ we fuse clusters {(1,2)} and {(3,4)} at height 0.8

Assignment 4



b)



We have dissimilarity matrix as -

$$\begin{bmatrix} & 0.3 & 0.4 & 0.7 \\ 0.3 & & 0.5 & 0.8 \\ 0.4 & 0.5 & & 0.45 \\ 0.7 & 0.8 & 0.45 & \end{bmatrix}$$

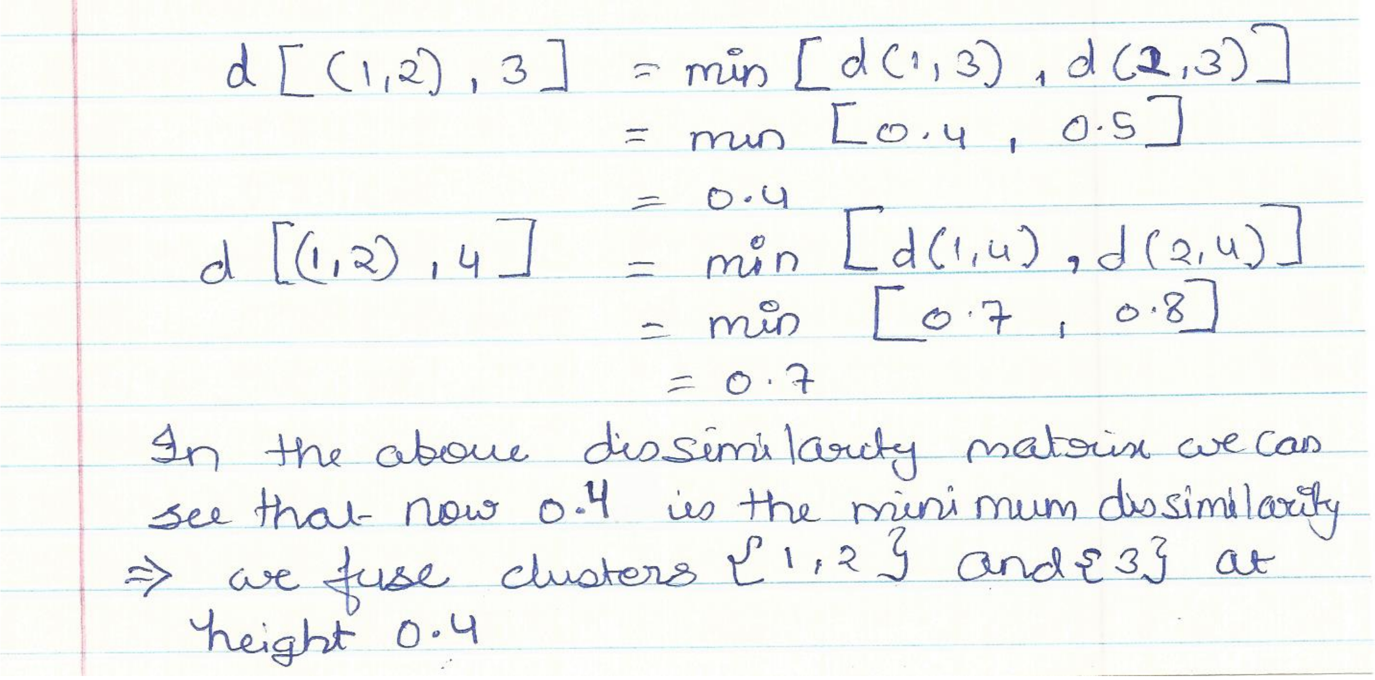
For single linkage -

we can see that 0.3 is the minimum dissimilarity

⇒ we fuse {1} and {2} to form cluster (1,2) at height 0.3

now we find the new dissimilarity matrix as

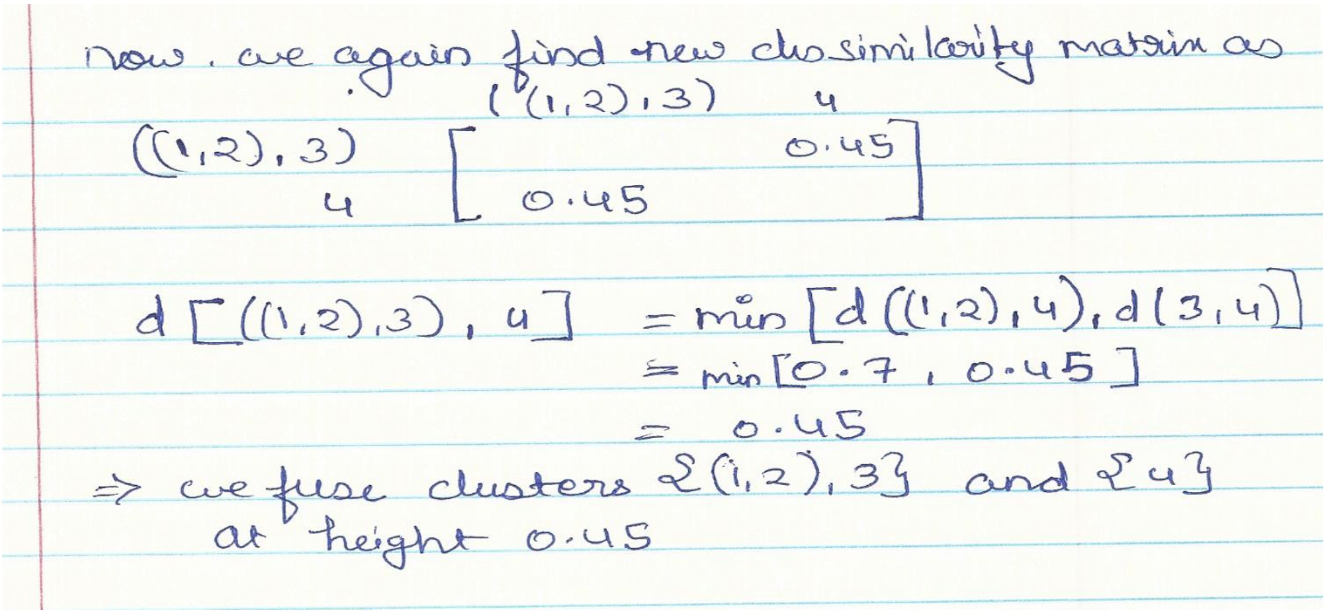
$$\begin{array}{c|ccc} & (1,2) & 3 & 4 \\ \hline (1,2) & & 0.4 & 0.7 \\ 3 & 0.4 & & 0.45 \\ 4 & 0.7 & 0.45 & \end{array}$$



$$d[(1,2),3] = \min[d(1,3), d(2,3)]$$
$$= \min[0.4, 0.5]$$
$$= 0.4$$

$$d[(1,2),4] = \min[d(1,4), d(2,4)]$$
$$= \min[0.7, 0.8]$$
$$= 0.7$$

In the above dissimilarity matrix we can see that now 0.4 is the minimum dissimilarity
⇒ we fuse clusters {1,2} and {3} at height 0.4

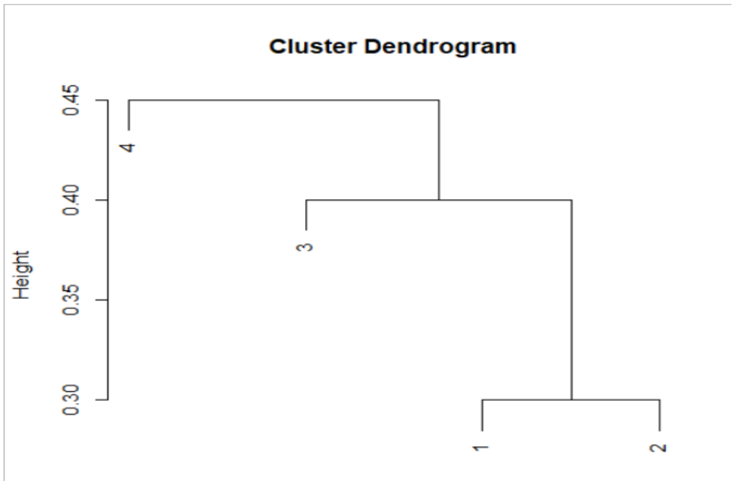


now, we again find new dissimilarity matrix as

$$\begin{array}{c|cc} & ((1,2),3) & 4 \\ \hline ((1,2),3) & & 0.45 \\ 4 & 0.45 & \end{array}$$

$$d[((1,2),3), 4] = \min[d((1,2),4), d(3,4)]$$
$$= \min[0.7, 0.45]$$
$$= 0.45$$

⇒ we fuse clusters {(1,2),3} and {4} at height 0.45



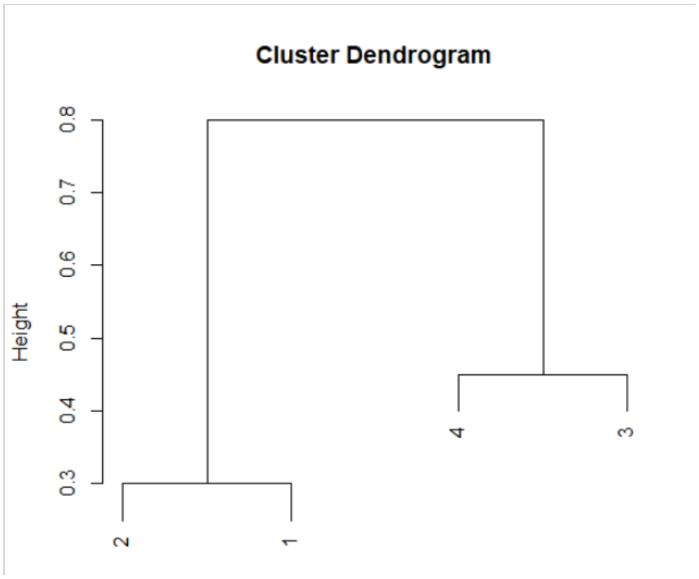
c) When we cut the dendrogram obtained in (a) such that it results in two clusters then the observations in each cluster will be:

Cluster 1:	(1,2)
Cluster 2:	(3,4)

d) When we cut the dendrogram obtained in (b) such that it results in two clusters then the observations in each cluster will be:

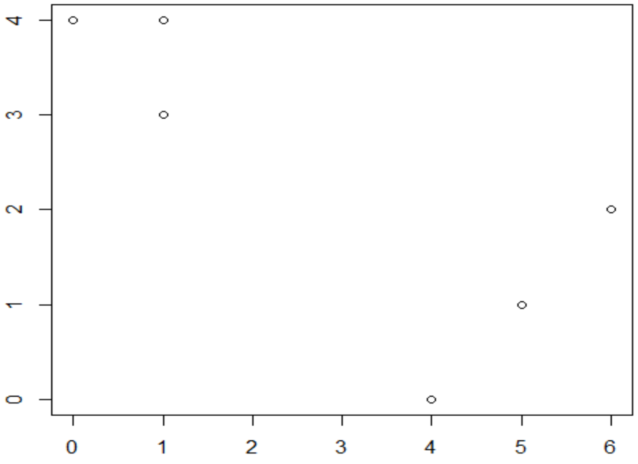
Cluster 1:	((1,2),3)
Cluster 2:	(4)

e)



Que 3

a)

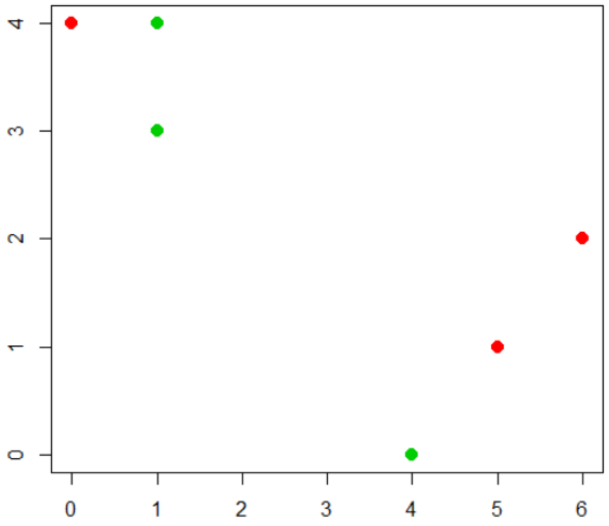


b)

```
set.seed(10)
labels <- sample(2, nrow(x), replace = T)
labels
```

```
## [1] 2 2 1 1 1 2
```

Assignment 4



c) centroid for cluster 1:

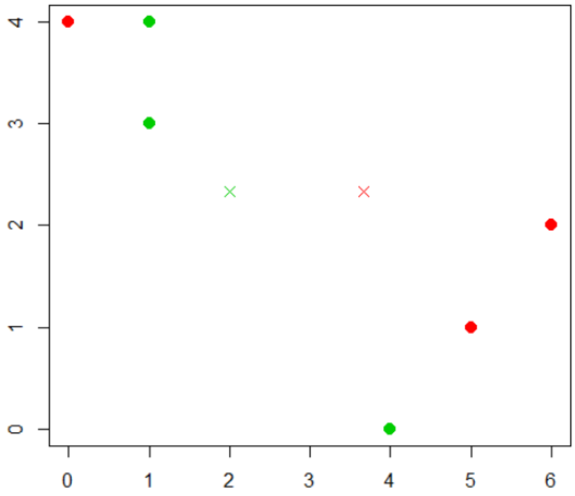
$$X1= \frac{(x1+x2+x3)}{3} = \frac{(0+5+6)}{3} = 3.67$$

$$Y1= \frac{(y1+y2+y3)}{3} = \frac{(4+1+2)}{3} = 2.33$$

centroid for cluster 2:

$$X1= \frac{(x1+x2+x3)}{3} = \frac{(1+1+4)}{3} = 2$$

$$Y1= \frac{(y1+y2+y3)}{3} = \frac{(4+3+0)}{3} = 2.33$$



d) Calculating Euclidean distance for each point from centroid of both the clusters and assigning them to the cluster which has smaller distance.

For point (1,4)

Distance from centroid 1 = $\sqrt{(1 - 3.67)^2 + (4 - 2.33)^2}$ = 3.149

Distance from centroid 2 = $\sqrt{(1 - 2)^2 + (4 - 2.33)^2}$ = 1.946

⇨ Point (1,4) belongs to cluster 2

For point (1,3)

Distance from centroid 1 = $\sqrt{(1 - 3.67)^2 + (3 - 2.33)^2}$ = 2.752

Distance from centroid 2 = $\sqrt{(1 - 2)^2 + (3 - 2.33)^2}$ = 1.203

⇨ Point (1,3) belongs to cluster 2

For point (0,4)

Distance from centroid 1 = $\sqrt{(0 - 3.67)^2 + (4 - 2.33)^2}$ = 4.032

Distance from centroid 2 = $\sqrt{(0 - 2)^2 + (4 - 2.33)^2}$ = 3.284

⇨ Point (0,4) belongs to cluster 2

For point (5,1)

Distance from centroid 1 = $\sqrt{(5 - 3.67)^2 + (1 - 2.33)^2}$ = 1.880

Distance from centroid 2 = $\sqrt{(5 - 2)^2 + (1 - 2.33)^2}$ = 3.218

⇨ Point (5,1) belongs to cluster 1

For point (6,2)

Distance from centroid 1 = $\sqrt{(6 - 3.67)^2 + (2 - 2.33)^2}$ = 2.353

Distance from centroid 2 = $\sqrt{(6 - 2)^2 + (2 - 2.33)^2}$ = 4.013

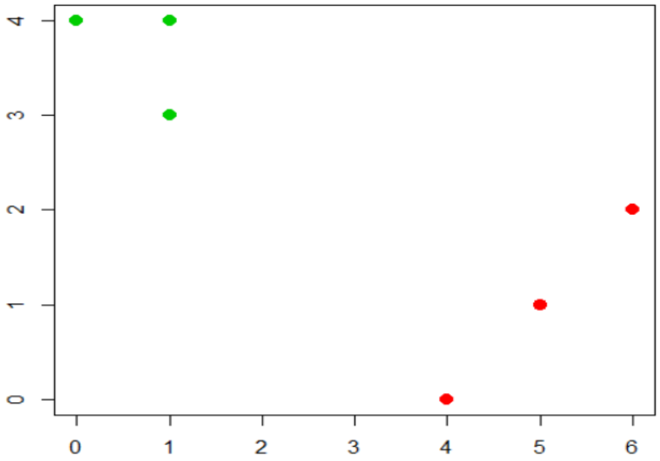
⇨ Point (6,2) belongs to cluster 1

For point (4,0)

Distance from centroid 1 = $\sqrt{(4 - 3.67)^2 + (0 - 2.33)^2}$ = 2.353

Distance from centroid 2 = $\sqrt{(4 - 2)^2 + (0 - 2.33)^2}$ = 9.428

⇨ Point (4,0) belongs to cluster 1



e) centroid for cluster 1:

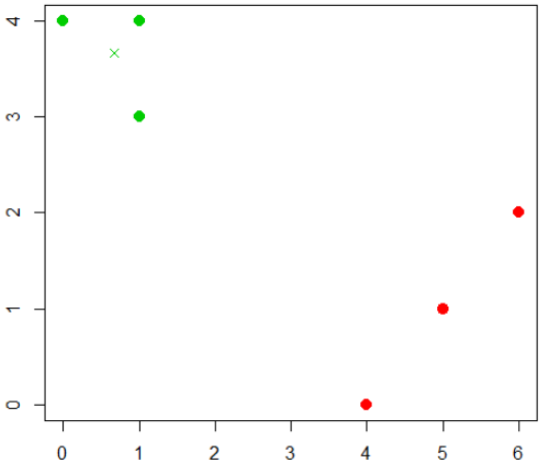
$X1 = \frac{(x1+x2+x3)}{3} = \frac{(5+6+4)}{3} = 5$

$Y1 = \frac{(y1+y2+y3)}{3} = \frac{(1+2+0)}{3} = 1$

centroid for cluster 2:

$X1 = \frac{(x1+x2+x3)}{3} = \frac{(1+1+0)}{3} = 0.667$

$Y1 = \frac{(y1+y2+y3)}{3} = \frac{(4+3+4)}{3} = 3.667$



Now from the new centroids we can see that if we assign each observation to its new centroid, the label of each point will remain the same. Hence the algorithm will terminate here as nothing will change.

Que 4

a) given:
two clusters are {1,2,3} and {4,5}
According to the question, these two clusters will fuse at certain point for both single linkage dendrogram and for complete linkage dendrogram but **there is not enough information** to tell which fusion will occur higher on tree because it totally depends on the inter-observations distance.
For example,
Suppose the inter-observations distance is given as d(1,4) =2, d(1,5) =3, d(2,4) =1, d(2,5) =3, d(3,4) =4 and d(3,5) =1, then the single linkage dissimilarity between {1,2,3} and {4,5} will be equal to 1 and the complete linkage dissimilarity will be equal to 4. Hence the complete linkage will occur higher on the tree.
Now take another example,
Suppose the inter-observation distance is same for all the observations and is equal to 2, then the both single linkage dissimilarity and complete linkage dissimilarity between {1,2,3} and {4,5} will be equal to 2. Hence both the clusters will fuse at same height.
So, from the above two examples we can say that **we will require more information to derive the exact results**.

b) given:
two clusters are {5} and {6}
According to the question, these two clusters will fuse at certain point for both single linkage dendrogram and for complete linkage dendrogram. And we can say that **both clusters will fuse at same height** because inter-observation distance will be same for both the cases**.**
For example,
suppose d(5,6) = 3
then for both single and complete linkage, dissimilarities between {5} and {6} will be equal to 3. So, we can fuse both the clusters at height of 3.

Que 6

a) The first principle component "explains 10% of the variation" means that only the 10% of the information in the gene dataset is explained by projecting the tissue sample observations onto the first principle component. It also means that 90% of the information in the gene dataset is lost. Because the first principle component explains the maximum information of the dataset and in this case only 10% of the information is explained.

b) I will suggest including the machine used (A vs B) as a feature of the dataset. This will increase the proportion of variance explained by the first principal component before applying the 2-sample t-test.

c)
```
set.seed(123)
control <- matrix(rnorm(50 * 1000), ncol = 50)
treatment <- matrix(rnorm(50 * 1000), ncol = 50)
data <- cbind(control, treatment)
data[1, ] <- seq(-18, 18 - .36, .36)
result <- prcomp(scale(data))
summary(result)$importance[, 1]
```

```
##     Standard deviation Proportion of Variance  Cumulative Proportion
##               3.159123               0.099800               0.099800
```

Here in this case, 9.98% of variance is explained by first principal component.
Now adding in A vs B via 10 vs 0 encoding

```
n_data <- rbind(data, c(rep(10, 50), rep(0, 50)))
n_result <- prcomp(scale(n_data))
summary(n_result)$importance[, 1]
```

```
##     Standard deviation Proportion of Variance  Cumulative Proportion
##               3.404544               0.115910               0.115910
```

Now we can see that 11.59% of variance is explained by first principal component. Which means that 1.611% more variance is explained by the first principal component than the previous one.

===

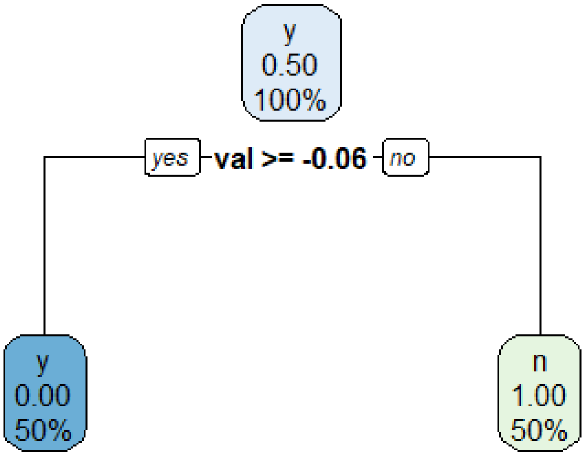
2. Practicum Problems:

2.1 Problem 1

```
library(rpart)
library(rpart.plot)

#function Definition
gini <- function(p)
{
  gini.index = 2 * p * (1 - p)
  return (gini.index)
}


entropy <- function(p)
{
  entropy = (p * log(p) + (1 - p) * log(1 - p))
  return (entropy)
}

set.seed(123)
a<-rnorm(n=150,mean=5,sd=2)
b<-rnorm(n=150,mean=-5,sd=2)
data1 <- data.frame(val = a,label=rep("y",150))
data2 <- data.frame(val = b,label=rep("n",150))
data <- rbind(data1,data2)
```

Assignment 4

```
data$label <- as.factor(data$label)
d_tree <- rpart(label~val,data,method="class")
rpart.plot(d_tree)
```



From the above tree we can see that threshold value for the first split will be -0.06. The tree has one root node and two leaf nodes. Also, tree is able to classify both classes separately which clearly shows empirical distribution.

```
#Calculating Gini and Entropy for Each Node:
#p=probability of each node
p=c(.5, 0, 1)

gini_values=sapply(p, gini)
gini_values

## [1] 0.5 0.0 0.0
```

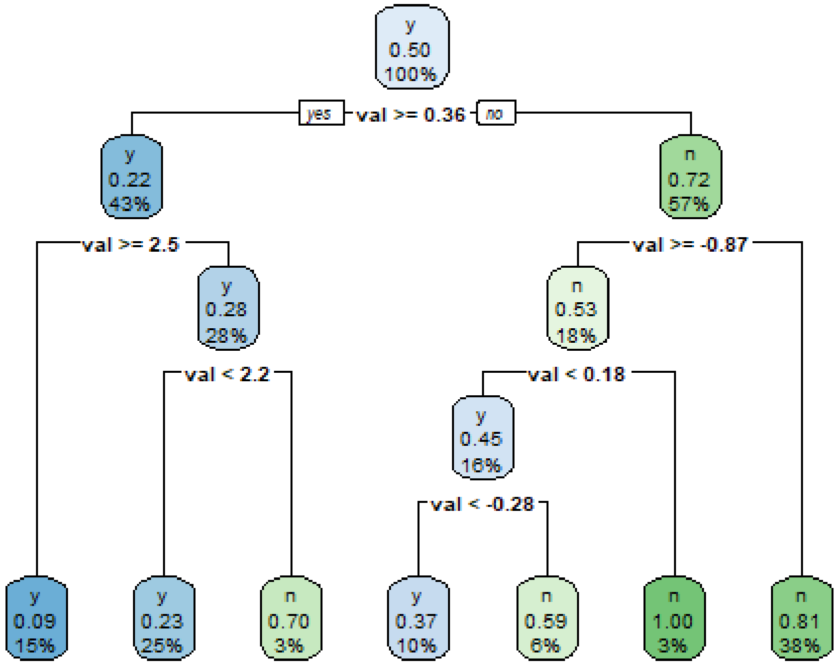
The gini values for above tree will be 0.5, 0.0, 0.0

```
entropy_values=sapply(p, entropy)
entropy_values

## [1] -0.6931472        NaN        NaN
```

The entropy values for above tree will be -0.6931472, NaN, NaN

```
set.seed(150)
a<-rnorm(n=150,mean=1,sd=2)
b<-rnorm(n=150,mean=-1,sd=2)
data1 <- data.frame(val = a,label=rep("y",150))
data2 <- data.frame(val = b,label=rep("n",150))
data <- rbind(data1,data2)
data$label <- as.factor(data$label)
d_tree <- rpart(label~val,data,method="class")
rpart.plot(d_tree)
```

Assignment 4



From the above tree we can see that threshold value for the first split will be 0.36. The tree has total of 13 nodes in which one of the nodes is root node and has total of 7 leaf nodes. Large tree size shows presence of more different labels in node, which resulted in a large tree. So, this tree has more overlapping of labels in nodes.

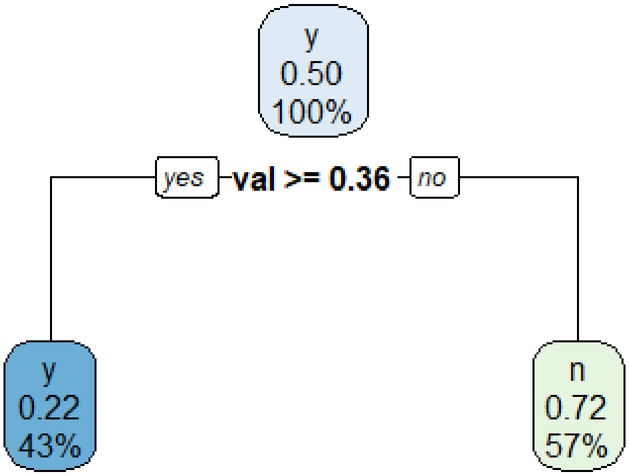
```
#Calculating Gini and Entropy for Each Node:
#p=probability of each node
p=c(.5,0.22,0.72,0.28,0.53,0.45,0.09,0.23,0.70,0.37,0.59,1.0,0.81)
gini_values=sapply(p, gini)
gini_values

##  [1] 0.5000 0.3432 0.4032 0.4032 0.4982 0.4950 0.1638 0.3542 0.4200 0.4662
## [11] 0.4838 0.0000 0.3078
```

The gini values for above tree will be 0.5000, 0.3432, 0.4032, 0.4032, 0.4982, 0.4950, 0.1638, 0.3542, 0.4200, 0.4662,0.4838, 0.0000, 0.3078

```
entropy_values=sapply(p, entropy)
entropy_values

##  [1] -0.6931472 -0.5269080 -0.5929533 -0.5929533 -0.6913461 -0.6881388
##  [7] -0.3025378 -0.5392763 -0.6108643 -0.6589557 -0.6768585        NaN
## [13] -0.4862230
```

The entropy values for above tree will be -0.6931472, -0.5269080, -0.5929533, -0.5929533, -0.6913461, -0.6881388, -0.3025378, -0.5392763, -0.6108643, -0.6589557, -0.6768585, NaN, -0.4862230

```
new_d_tree <- prune.rpart(d_tree,cp=0.1)
rpart.plot(new_d_tree)
```



From the above tree we can see that threshold value for the first split will be 0.36. The tree has one root node and 2 leaf nodes. Also, this pruned tree is much better than the previous as this has only two leaf nodes with less overlapping labels.

```
#Calculating Gini and Entropy for Each Node:
#p=probability of each node
p=c(.5,0.22,0.72)
gini_values=sapply(p, gini)
gini_values

## [1] 0.5000 0.3432 0.4032
```

The gini values for above tree will be 0.5000, 0.3432, 0.4032

```
entropy_values=sapply(p, entropy)
entropy_values

## [1] -0.6931472 -0.5269080 -0.5929533
```

The entropy values for above tree will be -0.6931472, -0.5269080, -0.5929533

==

2.2 Problem 2

```
URL <- "https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data"
data <- read.table(URL,sep=",")
colnames(data) <- c("class","alcohol","malic_acid","ash","alcalinity","magnesium","total_
phenols","flavanoids",
                    "nonfalvanoid","roanthocyanins","color_intensity","hue","OD280/OD315"
,"proline")
#display top six rows
head(data)

##   class alcohol malic_acid  ash alcalinity magnesium total_phenols
## 1     1   14.23       1.71 2.43       15.6       127          2.80
## 2     1   13.20       1.78 2.14       11.2       100          2.65
## 3     1   13.16       2.36 2.67       18.6       101          2.80
## 4     1   14.37       1.95 2.50       16.8       113          3.85
## 5     1   13.24       2.59 2.87       21.0       118          2.80
## 6     1   14.20       1.76 2.45       15.2       112          3.27
##   flavanoids nonfalvanoid roanthocyanins color_intensity  hue OD280/OD315
## 1       3.06         0.28           2.29            5.64 1.04        3.92
## 2       2.76         0.26           1.28            4.38 1.05        3.40
## 3       3.24         0.30           2.81            5.68 1.03        3.17
## 4       3.49         0.24           2.18            7.80 0.86        3.45
## 5       2.69         0.39           1.82            4.32 1.04        2.93
## 6       3.39         0.34           1.97            6.75 1.05        2.85
##   proline
## 1    1065
## 2    1050
## 3    1185
## 4    1480
## 5     735
## 6    1450
```

```
#check the means of predictors
apply(data[,-1],2,mean)
```

```
##         alcohol      malic_acid             ash      alcalinity 
##      13.0006180       2.3363483       2.3665169      19.4949438 
##       magnesium   total_phenols      flavanoids    nonfalvanoid 
##      99.7415730       2.2951124       2.0292697       0.3618539 
##  roanthocyanins color_intensity             hue     OD280/OD315 
##       1.5908989       5.0580899       0.9574494       2.6116854 
##         proline 
##     746.8932584
```

```
#check the variance of the predictors
apply(data[,-1],2,var)
```

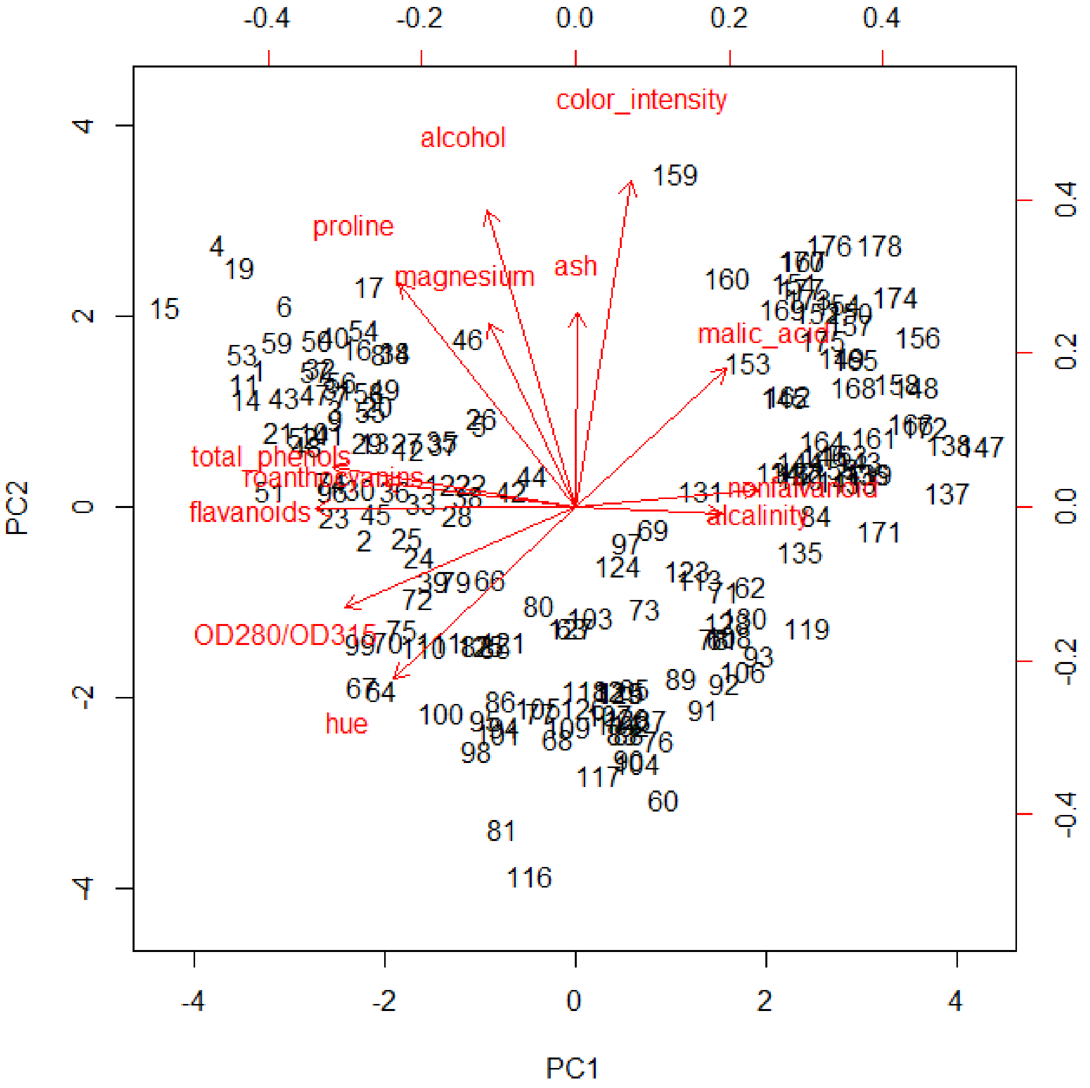
```
##         alcohol      malic_acid             ash      alcalinity 
##    6.590623e-01    1.248015e+00    7.526464e-02    1.115269e+01 
##       magnesium   total_phenols      flavanoids    nonfalvanoid 
##    2.039893e+02    3.916895e-01    9.977187e-01    1.548863e-02 
##  roanthocyanins color_intensity             hue     OD280/OD315 
##    3.275947e-01    5.374449e+00    5.224496e-02    5.040864e-01 
##         proline 
##    9.916672e+04
```

From the above mean and variance values it is clear that values are on different scale. So, we need to perform scaling before applying PCA to our dataset.

```
#using prcomp to perform PCA
output <- prcomp(data[,-1],scale=TRUE)
output$rotation
```

```
##                          PC1          PC2         PC3         PC4
## alcohol         -0.144329395  0.483651548 -0.20738262  0.01785630
## malic_acid       0.245187580  0.224930935  0.08901289 -0.53689028
## ash              0.002051061  0.316068814  0.62622390  0.21417556
## alcalinity        0.239320405 -0.010590502  0.61208035 -0.06085941
## magnesium        -0.141992042  0.299634003  0.13075693  0.35179658
## total_phenols    -0.394660845  0.065039512  0.14617896 -0.19806835
## flavanoids       -0.422934297 -0.003359812  0.15068190 -0.15229479
## nonfalvanoid      0.298533103  0.028779488  0.17036816  0.20330102
## roanthocyanins   -0.313429488  0.039301722  0.14945431 -0.39905653
## color_intensity   0.088616705  0.529995672 -0.13730621 -0.06592568
## hue              -0.296714564 -0.279235148  0.08522192  0.42777141
## OD280/OD315      -0.376167411 -0.164496193  0.16600459 -0.18412074
## proline          -0.286752227  0.364902832 -0.12674592  0.23207086
##                          PC5         PC6         PC7         PC8
## alcohol         -0.26566365  0.21353865 -0.05639636  0.39613926
## malic_acid       0.03521363  0.53681385  0.42052391  0.06582674
## ash             -0.14302547  0.15447466 -0.14917061 -0.17026002
## alcalinity       0.06610294 -0.10082451 -0.28696914  0.42797018
## magnesium        0.72704851  0.03814394  0.32288330 -0.15636143
## total_phenols   -0.14931841 -0.08412230 -0.02792498 -0.40593409
## flavanoids      -0.10902584 -0.01892002 -0.06068521 -0.18724536
## nonfalvanoid    -0.50070298 -0.25859401  0.59544729 -0.23328465
## roanthocyanins   0.13685982 -0.53379539  0.37213935  0.36822675
## color_intensity -0.07643678 -0.41864414 -0.22771214 -0.03379692
## hue             -0.17361452  0.10598274  0.23207564  0.43662362
## OD280/OD315     -0.10116099  0.26585107 -0.04476370 -0.07810789
## proline         -0.15786880  0.11972557  0.07680450  0.12002267
##                          PC9        PC10        PC11        PC12
## alcohol         -0.50861912  0.21160473  0.22591696 -0.26628645
## malic_acid       0.07528304 -0.30907994 -0.07648554  0.12169604
## ash              0.30769445 -0.02712539  0.49869142 -0.04962237
## alcalinity      -0.20044931  0.05279942 -0.47931378 -0.05574287
## magnesium       -0.27140257  0.06787022 -0.07128891  0.06222011
## total_phenols   -0.28603452 -0.32013135 -0.30434119 -0.30388245
## flavanoids      -0.04957849 -0.16315051  0.02569409 -0.04289883
## nonfalvanoid    -0.19550132  0.21553507 -0.11689586  0.04235219
## roanthocyanins   0.20914487  0.13418390  0.23736257 -0.09555303
## color_intensity -0.05621752 -0.29077518 -0.03183880  0.60422163
## hue             -0.08582839 -0.52239889  0.04821201  0.25921400
## OD280/OD315     -0.13722690  0.52370587 -0.04642330  0.60095872
## proline          0.57578611  0.16211600 -0.53926983 -0.07940162
##                         PC13
## alcohol          0.01496997
## malic_acid       0.02596375
## ash             -0.14121803
## alcalinity       0.09168285
## magnesium        0.05677422
## total_phenols   -0.46390791
## flavanoids       0.83225706
## nonfalvanoid     0.11403985
## roanthocyanins  -0.11691707
## color_intensity -0.01199280
## hue             -0.08988884
## OD280/OD315     -0.15671813
## proline          0.01444734
```

```
#biplot
biplot(output,scale=0)
```

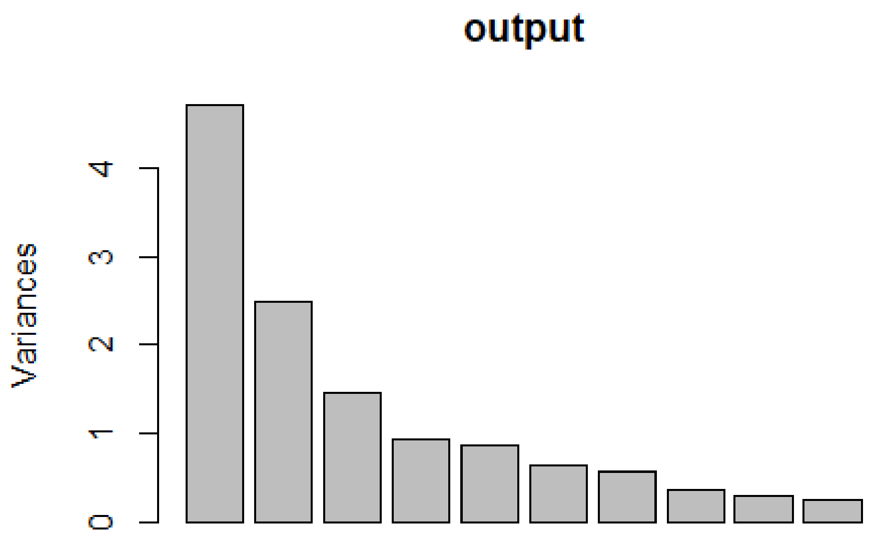


From the above plot we can see that feature malic_acid is pointed in opposite direction to the feature hue.

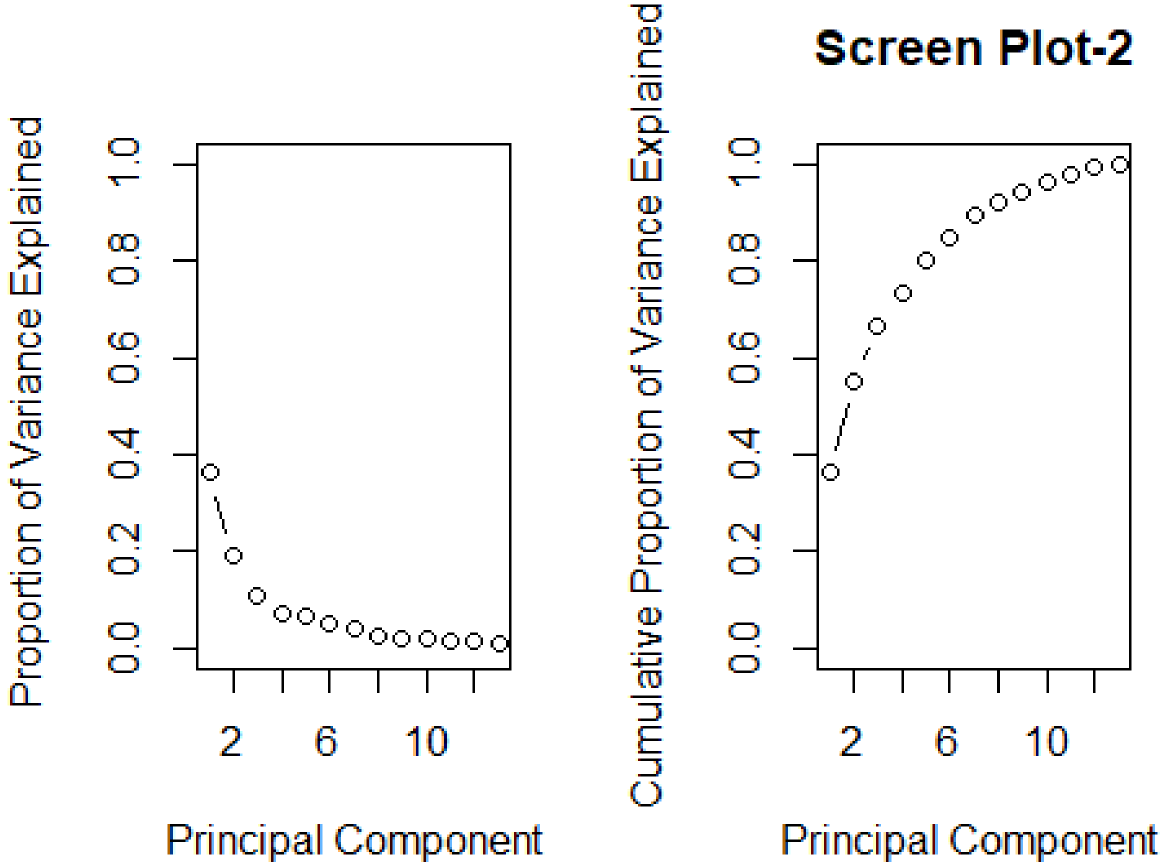
```
#calculating correlation between malic_acid and hue
cor(data$malic_acid,data$hue)

## [1] -0.5612957
```

From the correlation value between feature hue and malic acid it is clear that as the one variable increases the other variable decreases with the almost same extent.

```
#screeplot
screeplot(output)
```



```
summary(output)

## Importance of components:
##                          PC1    PC2    PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.169 1.5802 1.2025 0.95863 0.92370 0.80103 0.74231
## Proportion of Variance 0.362 0.1921 0.1112 0.07069 0.06563 0.04936 0.04239
## Cumulative Proportion  0.362 0.5541 0.6653 0.73599 0.80162 0.85098 0.89337
##                            PC8     PC9   PC10    PC11    PC12    PC13
## Standard deviation     0.59034 0.53748 0.5009 0.47517 0.41082 0.32152
## Proportion of Variance 0.02681 0.02222 0.0193 0.01737 0.01298 0.00795
## Cumulative Proportion  0.92018 0.94240 0.9617 0.97907 0.99205 1.00000

#calculating proportion of varianve for each principle component
variance <- output$sdev^2
pve <- variance/sum(variance)

#screenplot
par(mfrow=c(1,2))
plot(pve, xlab="Principal Component", ylab="Proportion of Variance Explained ",ylim=c(0,1
),type='b')
plot(cumsum(pve), xlab="Principal Component ", ylab=" Cumulative Proportion of Variance E
xplained ",main="Screen Plot-2", ylim=c(0,1), type='b')
```



```
#Proportion of variance expalined by PC1 and PC2
temp<-pve[1:2]*100
temp

## [1] 36.19885 19.20749

sum(temp)

## [1] 55.40634
```

Thus, from the above results it is clear that PC1 and PC2 has explained total of 55.40% of variance.

==

2.3 Problem 3

```
library("factoextra")

## Loading required package: ggplot2

## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/1
3EFCZ

library(tidyverse)

## -- Attaching packages --------------------------------------------------------------- tid
yverse 1.2.1 --
```

```
## v tibble  2.1.3     v purrr   0.3.3
## v tidyr   1.0.0     v dplyr   0.8.3
## v readr   1.3.1     v stringr 1.4.0
## v tibble  2.1.3     v forcats 0.4.0

## -- Conflicts --------------------------------------------------------------- tidyverse
_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

#load the dataset
data("USArrests")

#convert the dataset to a dataframe
data <- data.frame(USArrests)
head(data)

##            Murder Assault UrbanPop Rape
## Alabama      13.2     236       58 21.2
## Alaska       10.0     263       48 44.5
## Arizona       8.1     294       80 31.0
## Arkansas      8.8     190       50 19.5
## California    9.0     276       91 40.6
## Colorado      7.9     204       78 38.7

#dimensions of dataset
dim(data)

## [1] 50  4

#structure of dataset
str(data)

## 'data.frame':    50 obs. of  4 variables:
##  $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
##  $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
##  $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
##  $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...

#checking the mean of the predictors
apply(data,2,mean)

##   Murder  Assault UrbanPop     Rape
##    7.788  170.760   65.540   21.232

#checking the variance of the predictors
apply(data,2,var)

##     Murder    Assault   UrbanPop       Rape
##   18.97047 6945.16571  209.51878   87.72916
```

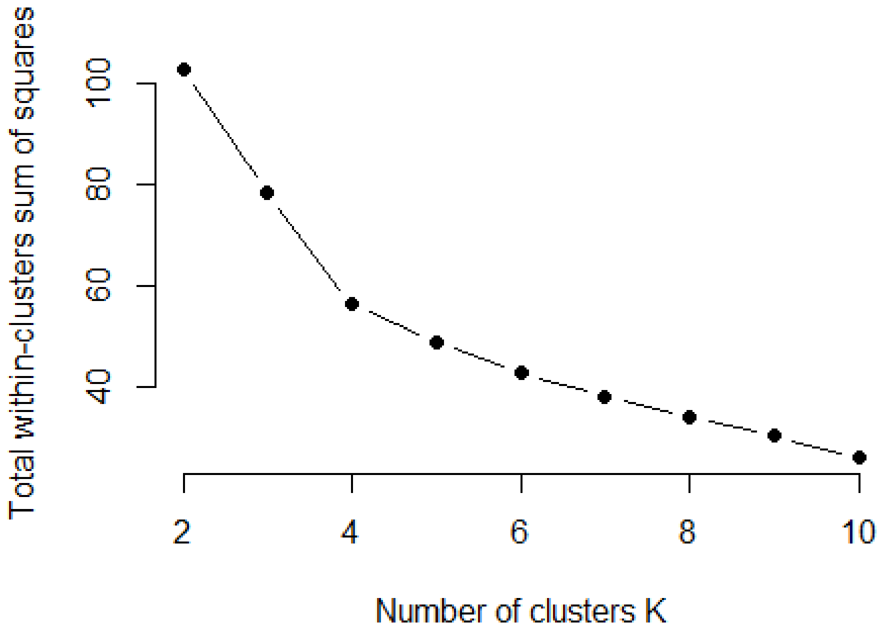
In the above mean and variance values it is clear that values are on different scale. So, we need to perform scaling before applying k-means to our dataset.

Assignment 4

```
#scaling the dataset
n_data <- scale(data, center = TRUE, scale = TRUE)


#Applying K-Means
result <- function(k)
{
  kmeans(n_data,centers=k,nstart=20)$tot.withinss
}
# values of k form 2 to 10
k <- 2:10

#compute total within-cluster sum of square values of k from 2 to 10
wss_val <- map_dbl(k, result)
wss_val
```

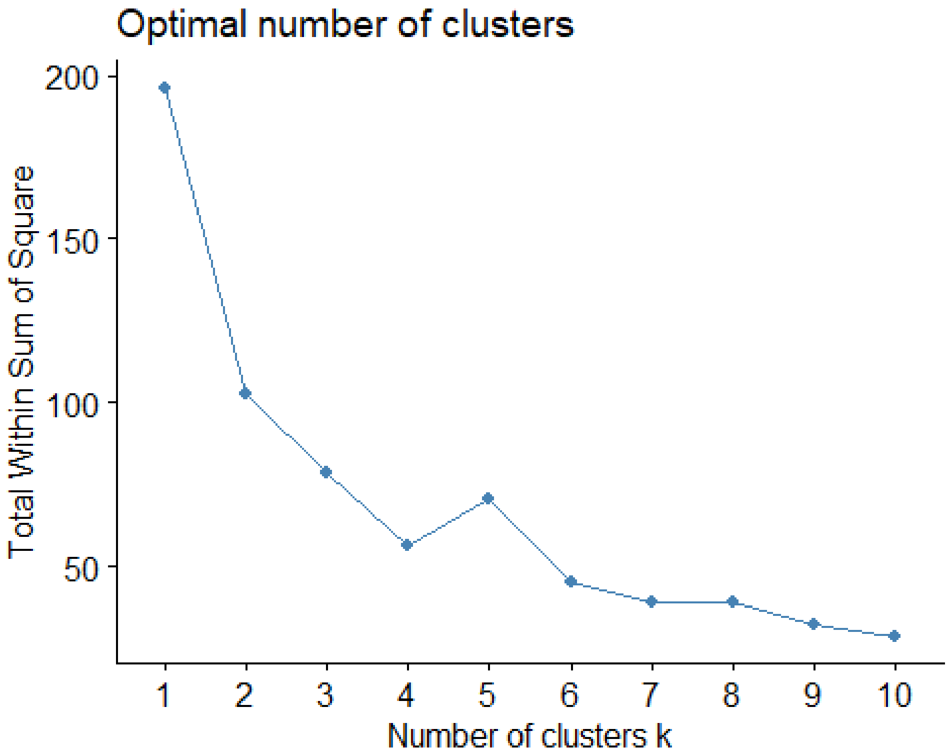
```
## [1] 102.86240  78.32327  56.40317  48.94420  42.83303  38.25764  34.10865
## [8]  30.42425  26.18348
```

```
#elbow method to find optimal K value
plot(k, wss_val,
     type="b", pch = 19, frame = FALSE,
     xlab="Number of clusters K",
     ylab="Total within-clusters sum of squares")
```



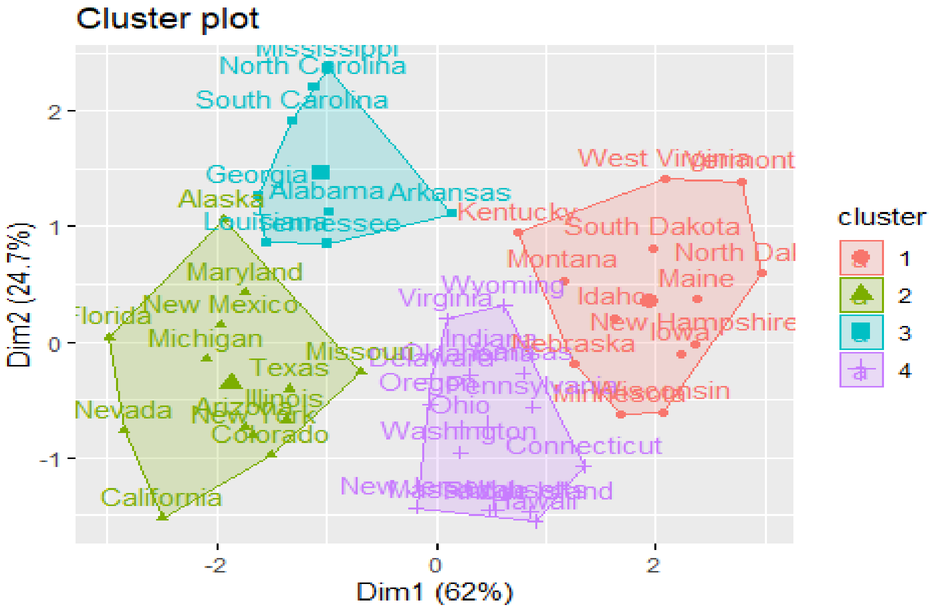
```
#another Method
fviz_nbclust(n_data, kmeans, method = "wss")
```

Assignment 4



From the above two graph it is clear that if we consider major drop in total within-clusters sum of square values then the optimal value of k in this case will be 4.

```
#plot for optimal clustering
optimal <- kmeans(n_data, centers = 4, nstart = 20)
fviz_cluster(optimal, data = n_data)
```



==

2.4 Problem 4

```
library(dplyr)
```

Assignment 4

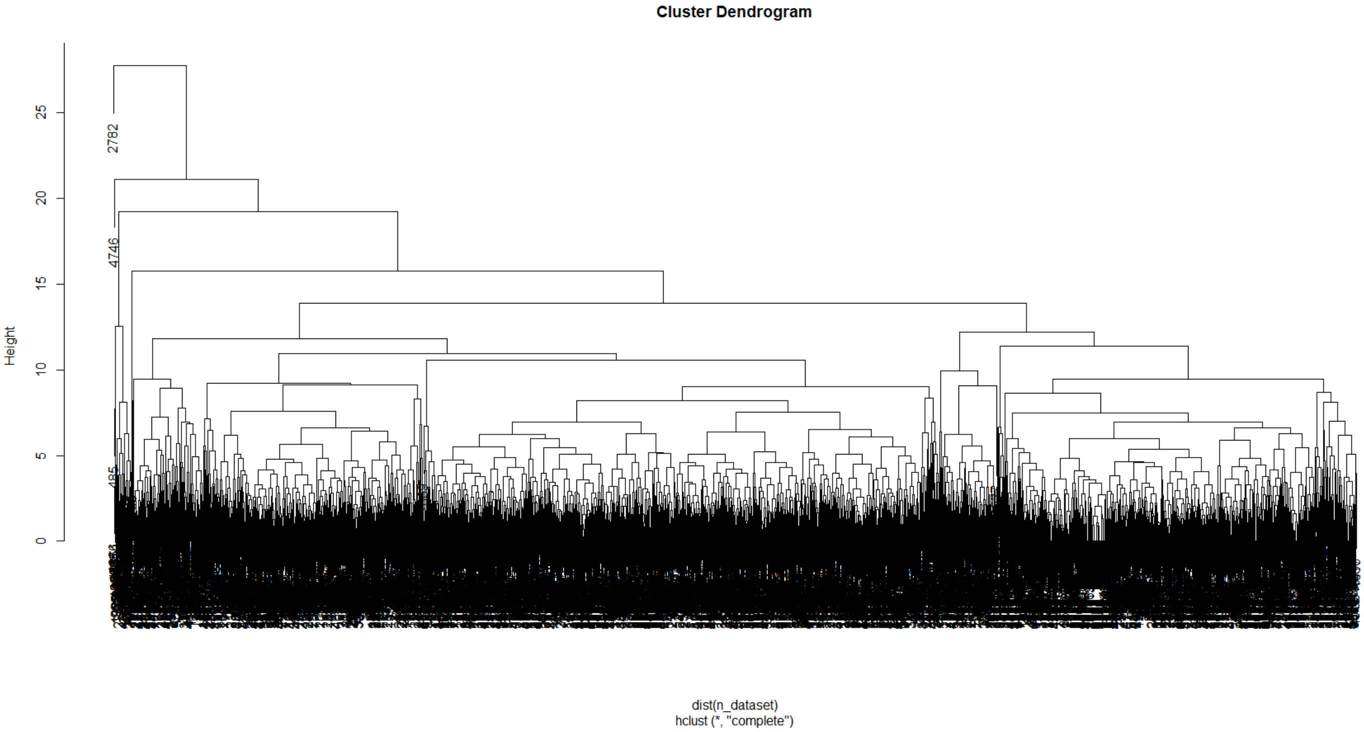
```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

#Importing the dataset
URL <- "https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequalit
y-white.csv"
wine <- read.csv(URL,sep=";")

#display dataset
head(wine)

##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 1           7.0             0.27        0.36           20.7     0.045
## 2           6.3             0.30        0.34            1.6     0.049
## 3           8.1             0.28        0.40            6.9     0.050
## 4           7.2             0.23        0.32            8.5     0.058
## 5           7.2             0.23        0.32            8.5     0.058
## 6           8.1             0.28        0.40            6.9     0.050
##   free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## 1                  45                  170  1.0010 3.00      0.45     8.8
## 2                  14                  132  0.9940 3.30      0.49     9.5
## 3                  30                   97  0.9951 3.26      0.44    10.1
## 4                  47                  186  0.9956 3.19      0.40     9.9
## 5                  47                  186  0.9956 3.19      0.40     9.9
## 6                  30                   97  0.9951 3.26      0.44    10.1
##   quality
## 1       6
## 2       6
## 3       6
## 4       6
## 5       6
## 6       6

#excluding quality variable
dataset <- wine[,-12]

#check mean of predictors
apply(dataset,2,mean)

##        fixed.acidity     volatile.acidity          citric.acid
##           6.85478767           0.27824112           0.33419151
##       residual.sugar            chlorides  free.sulfur.dioxide
##           6.39141486           0.04577236          35.30808493
## total.sulfur.dioxide              density                   pH
##         138.36065741           0.99402738           3.18826664
##            sulphates             alcohol
##           0.48984688         10.51426705
```

```
#check variance of predictors
apply(dataset,2,var)
```

```
##        fixed.acidity     volatile.acidity          citric.acid
##         7.121136e-01         1.015954e-02         1.464579e-02
##       residual.sugar            chlorides  free.sulfur.dioxide
##         2.572577e+01         4.773337e-04         2.892427e+02
## total.sulfur.dioxide              density                   pH
##         1.806085e+03         8.945524e-06         2.280118e-02
##            sulphates              alcohol
##         1.302471e-02         1.514427e+00
```

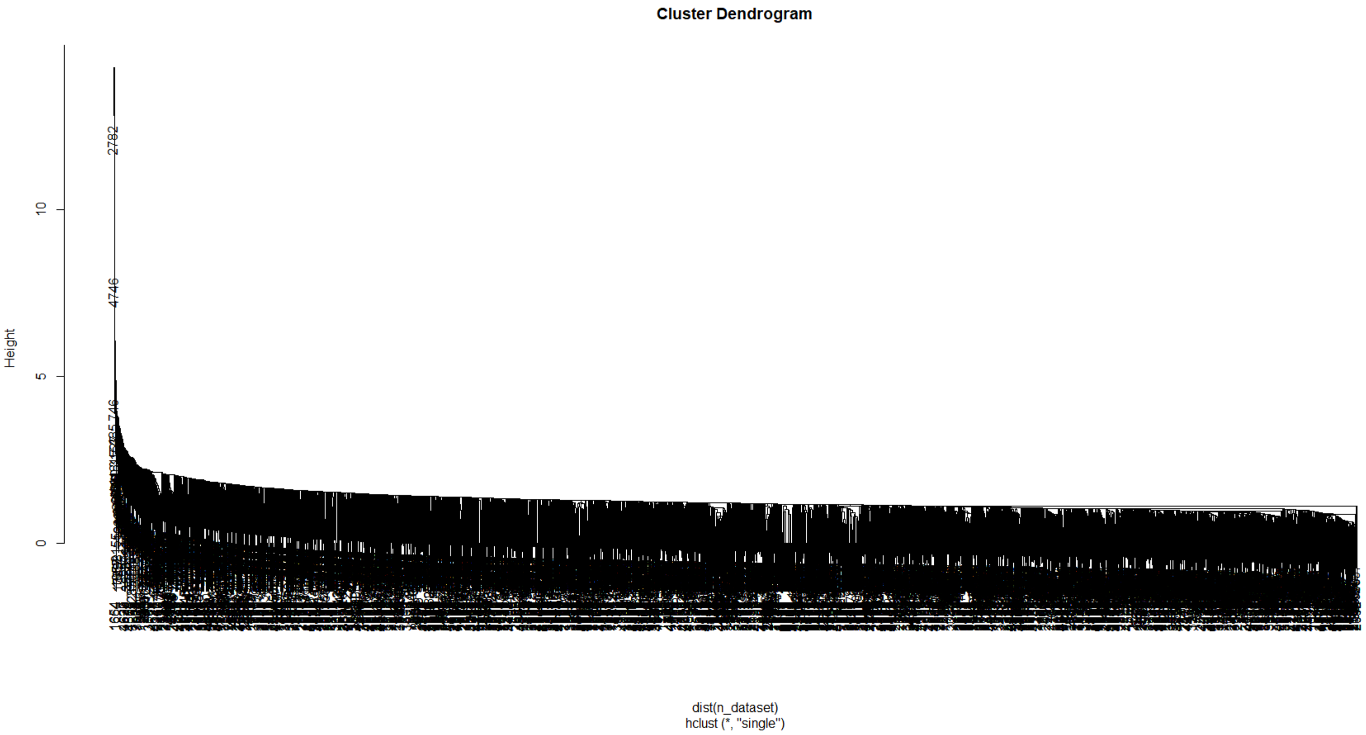
In the above mean and variance values it is clear that values are on different scale. So, we need to perform scaling before applying hclust to our dataset.

```
#scaling the model
n_dataset <- scale(dataset,center = TRUE,scale=TRUE)

#Performing hierarchical clustering using complete linkage
hc.complete <- hclust(dist(n_dataset),method="complete")
#dendogram of complete linkage
plot(hc.complete)
```



```
#Performing hierarchical clustering using single linkage
hc.single <- hclust(dist(n_dataset),method="single")
#dendogram of single linkage
plot(hc.single)
```

Assignment 4



```
#for complete linkage
tail(hc.complete$height,1)

## [1] 27.73476
```

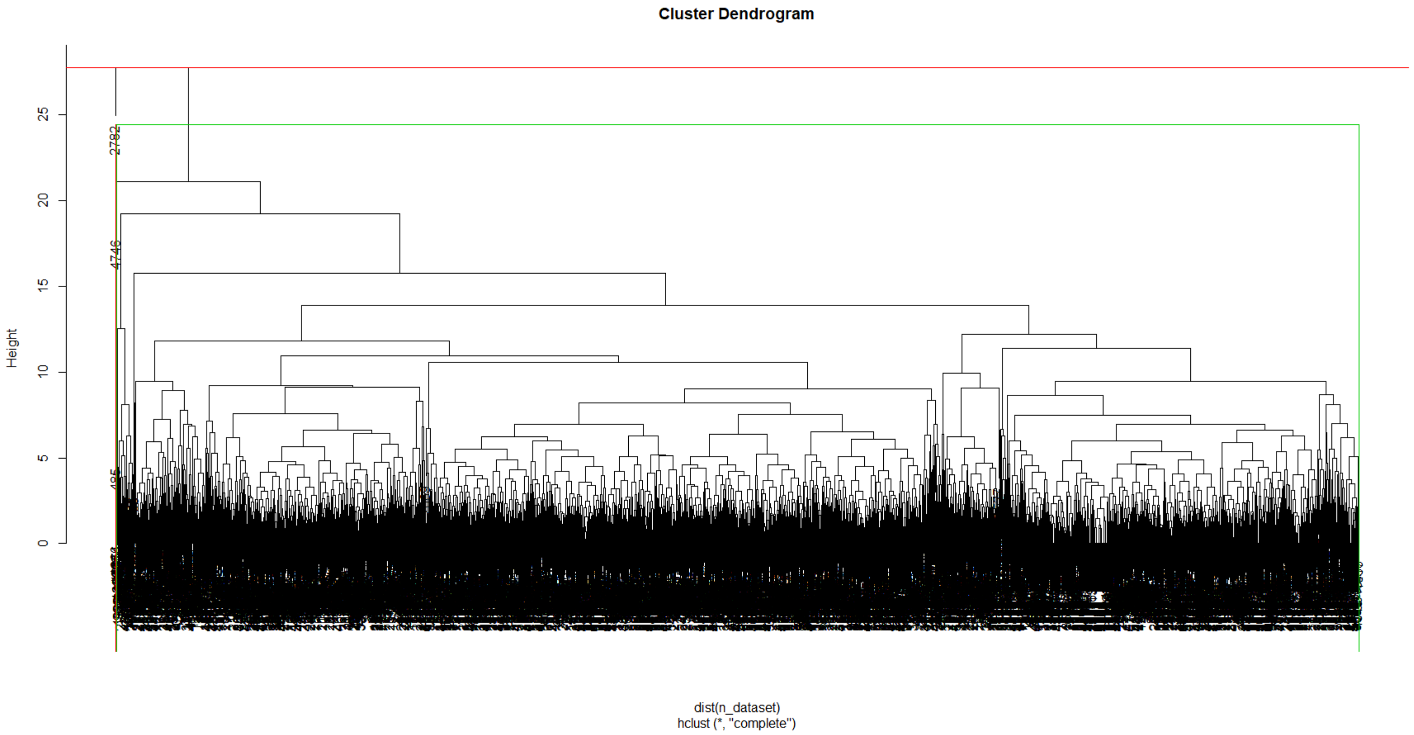
For single linkage two penultimate clusters will merge a 27.73476

```
#for single linkage
tail(hc.single$height,1)

## [1] 14.25323
```

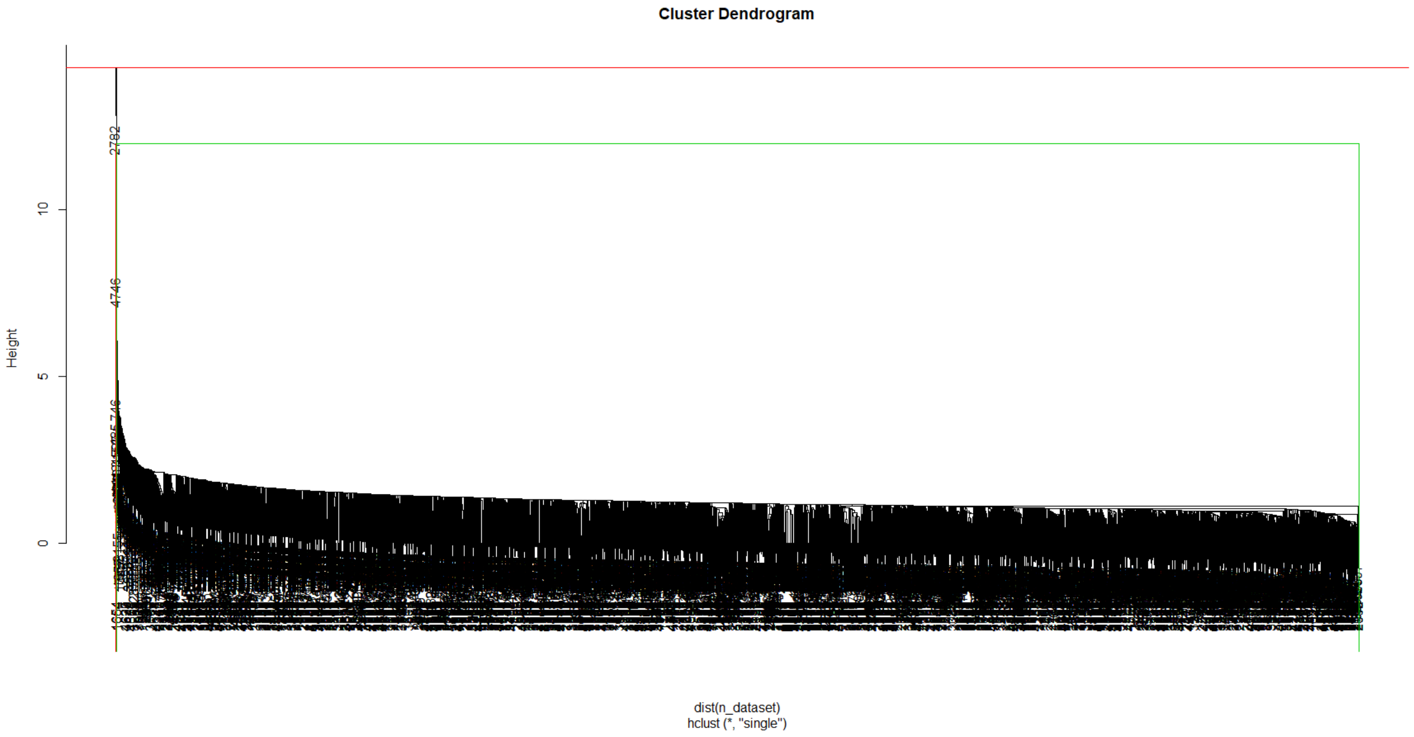
For single linkage two penultimate clusters will merge a 14.25325

```
#applying cutree method on complete linkage
cut.complete <- cutree(hc.complete,h=27.73476)
#Number of clusters formed
table(cut.complete)

## cut.complete
##    1    2
## 4897    1

plot(hc.complete)
rect.hclust(hc.complete ,h=27.73476, border = 2:6)
abline(h =27.73476, col = 'red')
```

Assignment 4



```
#applying cutree method on single linkage
cut.single <- cutree(hc.single,h=14.25323)
#Number of clusters formed
table(cut.single)

## cut.single
##    1    2
## 4897    1

plot(hc.single)
rect.hclust(hc.single ,h=14.25323, border = 2:6)
abline(h =14.25323, col = 'red')
```



```
#summary Statistics for complete linkage
dataset$Clusters <- cut.complete
unique(dataset$Clusters)

## [1] 1 2

dataset <- dplyr::group_by(dataset,Clusters)
a <- dplyr::summarise_each(dataset, funs(mean))

## Warning: funs() is soft deprecated as of dplyr 0.8.0
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once per session.

print.data.frame(a)

##   Clusters fixed.acidity volatile.acidity citric.acid residual.sugar
## 1        1      6.854595        0.2781009   0.3341372       6.379283
## 2        2      7.800000        0.9650000   0.6000000      65.800000
##    chlorides free.sulfur.dioxide total.sulfur.dioxide   density       pH
## 1 0.04576659            35.31366             138.3562 0.9940182 3.188225
## 2 0.07400000             8.00000             160.0000 1.0389800 3.390000
##   sulphates  alcohol
## 1  0.489806 10.51402
## 2  0.690000 11.70000

#Difference in feature means for complete linkage
abs(a[2,-1]-a[1,-1])

##   fixed.acidity volatile.acidity citric.acid residual.sugar  chlorides
## 1     0.9454054        0.6868991   0.2658628       59.42072 0.02823341
##   free.sulfur.dioxide total.sulfur.dioxide   density        pH sulphates
## 1            27.31366             21.64376 0.0449618 0.2017746  0.200194
##    alcohol
## 1 1.185975

#summary Statistics for single linkage
dataset$Clusters <- cut.single
unique(dataset$Clusters)

## [1] 1 2

dataset <- dplyr::group_by(dataset,Clusters)
b <- dplyr::summarise_each(dataset, funs(mean))
print.data.frame(b)

##   Clusters fixed.acidity volatile.acidity citric.acid residual.sugar
## 1        1      6.854595        0.2781009   0.3341372       6.379283
## 2        2      7.800000        0.9650000   0.6000000      65.800000
##    chlorides free.sulfur.dioxide total.sulfur.dioxide   density       pH
## 1 0.04576659            35.31366             138.3562 0.9940182 3.188225
## 2 0.07400000             8.00000             160.0000 1.0389800 3.390000
##   sulphates  alcohol
## 1  0.489806 10.51402
## 2  0.690000 11.70000

#Difference in feature means
abs(b[2,-1]-b[1,-1])

##   fixed.acidity volatile.acidity citric.acid residual.sugar  chlorides
## 1     0.9454054        0.6868991   0.2658628       59.42072 0.02823341
##   free.sulfur.dioxide total.sulfur.dioxide   density        pH sulphates
## 1            27.31366             21.64376 0.0449618 0.2017746  0.200194
##    alcohol
## 1 1.185975
```

From the above results we can see that feature residual.sugar has maximum means difference. Also, from the above two plots of Complete and Single linkage we can conclude that Complete linkage produces more balanced clustering.

==