

# Multivariate Analysis: Assignment 2

Harshad Kumar Elangovan - 19200349

15/04/2020

The votes of TDs are recorded and data on whether each TD voted yes or not for six of these votes are stored in file 32ndDail\_FifthSession\_BinaryVotes.Rdata. The data record has the value 2 if a TD was voted 'yes' or 1 if it is 'not' voted. This dataset is loaded in R and used for clustering the data.

```
library(miceadds)
library("poLCA")
library(ade4)
library(e1071)
library(vegan)
library(mclust)

load.Rdata(filename="32ndDail_FifthSession_BinaryVotes.Rdata","votingdata")
```

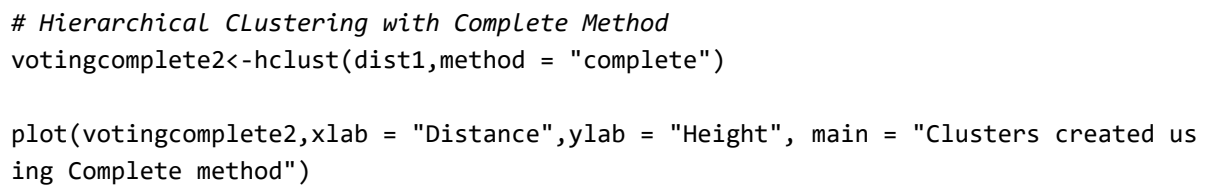
1 (a) Load the voting data into R. Which of the clustering methods that we have seen to date could be used to cluster the TDs based on these binary data? Apply your chosen clustering method to the binary voting data, detailing any decisions you make in the process. How many clusters of TDs do you think are present?

The dataset can be clustered using Hierarchical clustering method for the binary data. The dataset is converted to 0's and 1's (binary format) based on the existing data. This data is then used for calculating the dissimilarity matrix of this binary dataset.

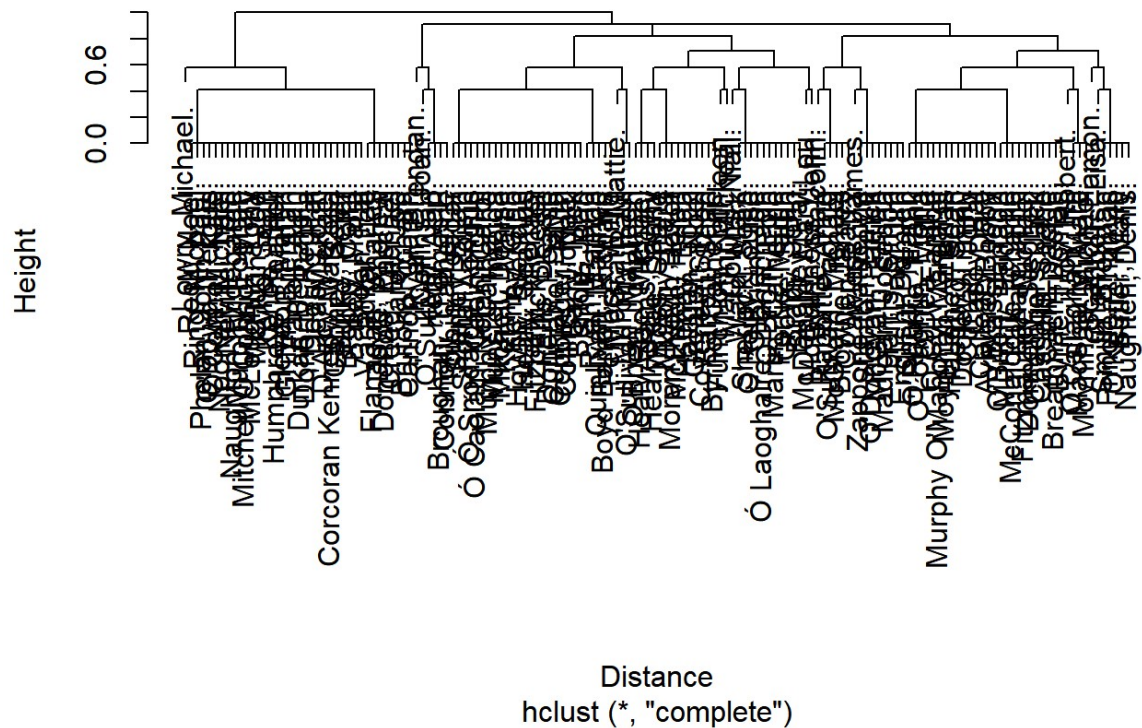
```
votingdata$Environment<-as.numeric(as.character(factor(votingdata$Environment,levels = c(1,2),labels = c(1,0))))
votingdata$RentFreeze<-as.numeric(as.character(factor(votingdata$RentFreeze,levels = c(1,2),labels = c(1,0))))
votingdata$SocialWelfare<-as.numeric(as.character(factor(votingdata$SocialWelfare,levels = c(1,2),labels = c(1,0))))
votingdata$GamingAndLotteries<-as.numeric(as.character(factor(votingdata$GamingAndLotteries,levels = c(1,2),labels = c(1,0))))
votingdata$HousingMinister<-as.numeric(as.character(factor(votingdata$HousingMinister,levels = c(1,2),labels = c(1,0))))
votingdata$FirstTimeBuyers<-as.numeric(as.character(factor(votingdata$FirstTimeBuyers,levels = c(1,2),labels = c(1,0))))
```

We can use "ade4" R package for calculating the dissimilarity matrix using 'Simple Matching' method. The dissimilarity matrix is used to construct the Hierarchical clustering of the data. The cluster with all three method is created and the best one is selected from it.

```
plot(votingSingle2,xlab = "Distance",ylab = "Height", main = "Clusters created using Single method")
```

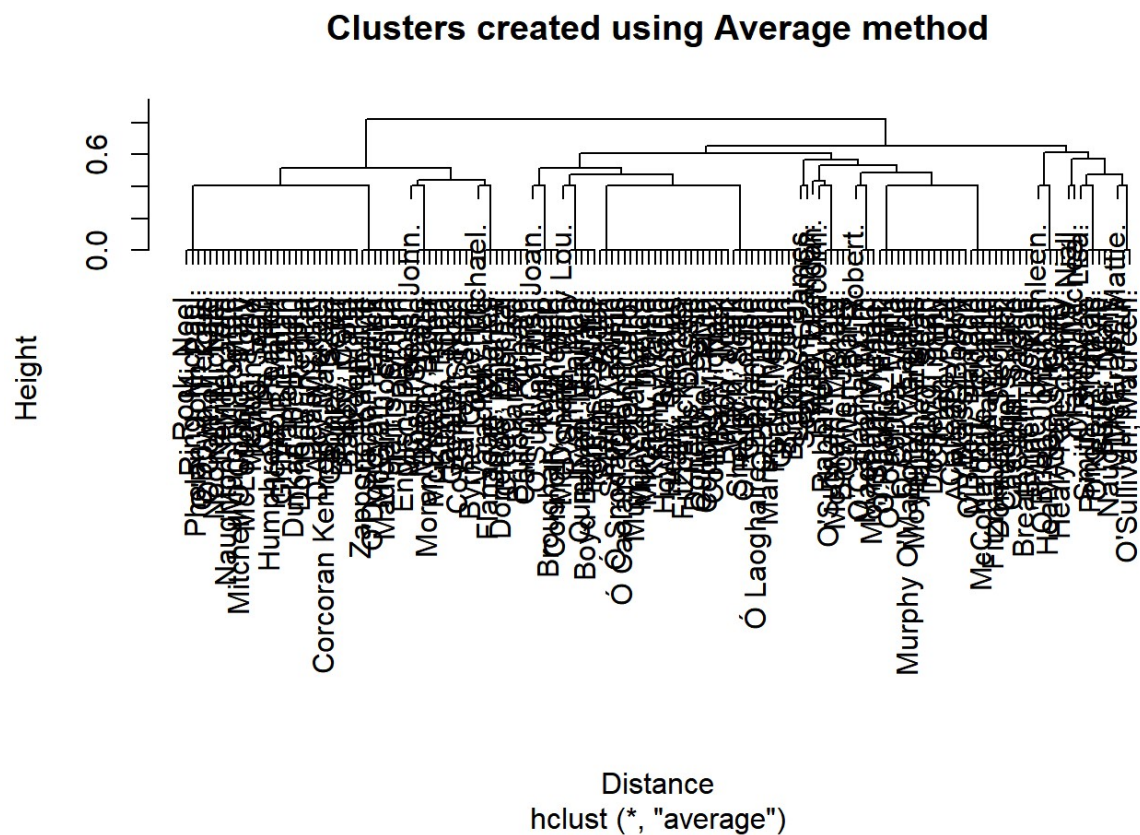


## Clusters created using Complete method



```
# Hierarchical Clustering with Average Method
votingavg2<-hclust(dist1,method = "average")
```

```
plot(votingavg2,xlab = "Distance",ylab = "Height", main = "Clusters created using A
verage method")
```



From the plots, the cluster constructed by 'Average' method visualizes the distance better when compared with other types of methods. We can clearly see that we can split the data into 2 clusters.

```
plot(votingavg2,xlab = "Distance",ylab = "Height", main = "Clusters created using Average method")
abline(h=0.7,col="red")
```

[illegible]

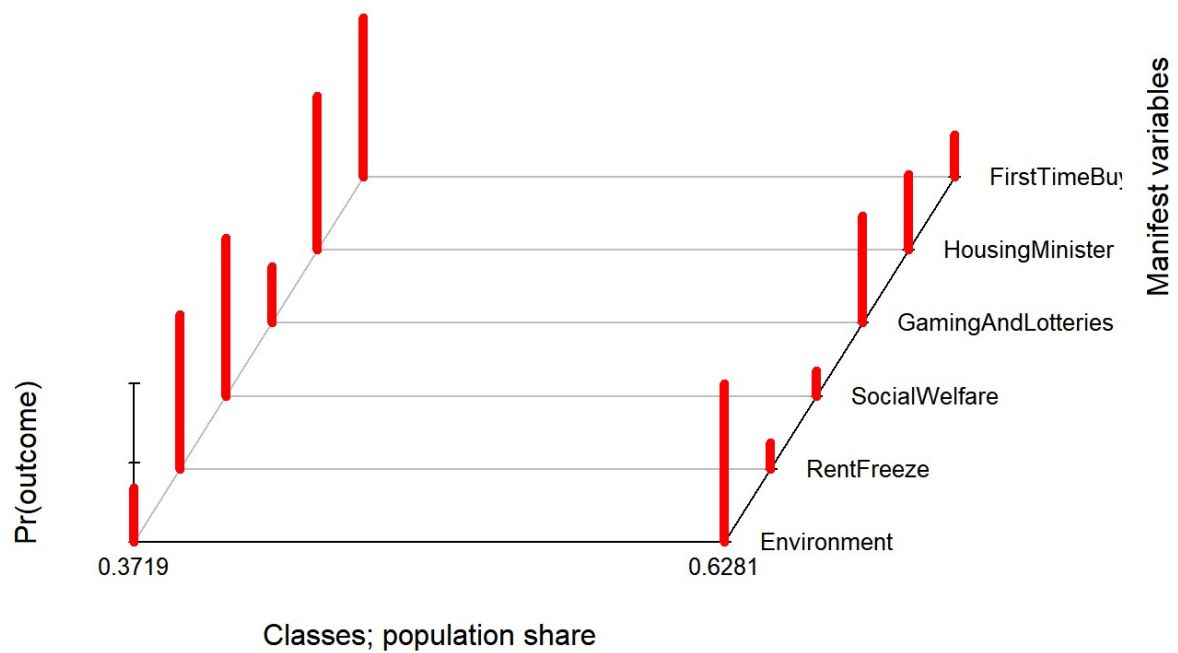
1 (b) Use the `poLCA` function in R to cluster the TDs based on their voting data. Detail any decisions you make in the process. How many clusters of TDs do you think are present now? On what basis did you make this decision? Include any output or plots which you use to motivate your decision

```
fx<-cbind(Environment,RentFreeze,SocialWelfare,GamingAndLotteries,HousingMinister,FirstTimeBuyers)~1
```

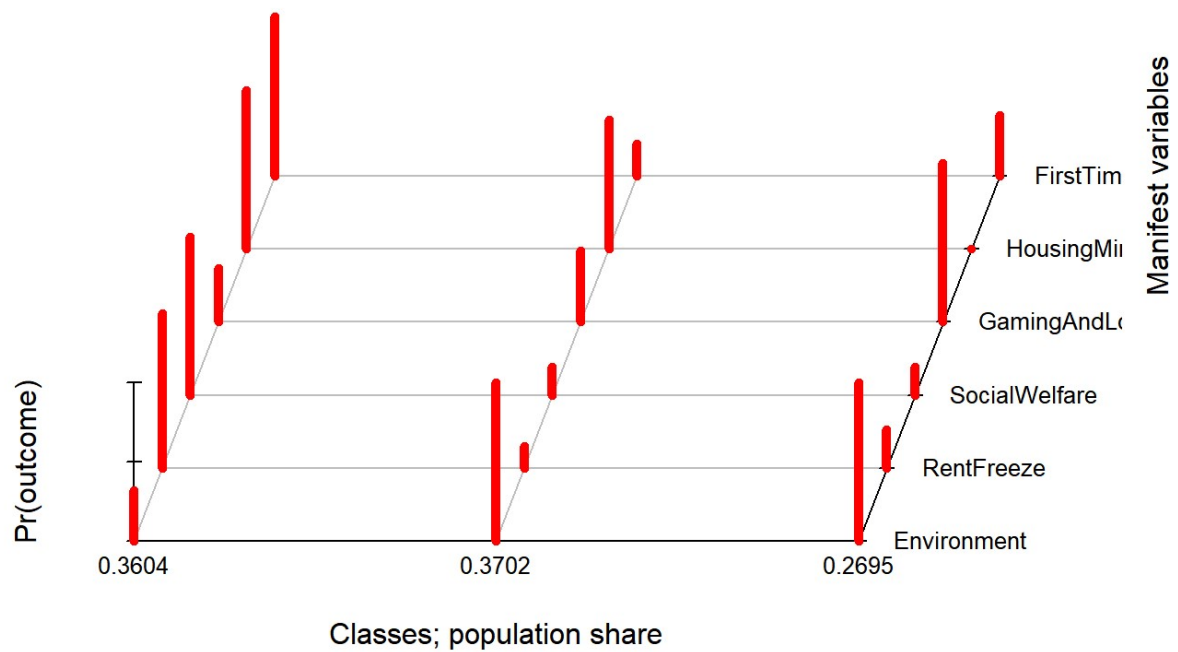
*# Three models are developed based on the classes and their BIC are compared to summarize the optimal model. Parameter Verbose can be used to restrict the function to print all the output to the screen.*

```
poLCA2<-poLCA(f,data = votingdata +1,nclass = 2,graphs = TRUE,na.rm = TRUE,maxiter = 1000,verbose = FALSE)
```

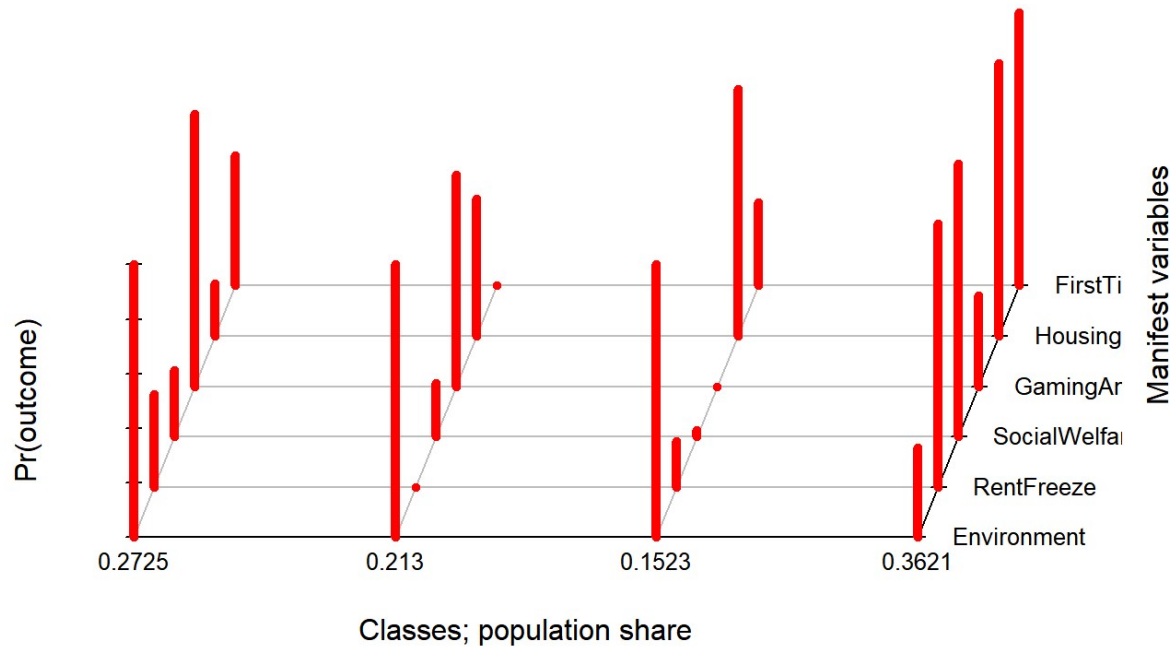




```
poLCA3<-poLCA(f,data = votingdata +1,nclass = 3,graphs = TRUE,na.rm = TRUE,maxiter
= 1000,verbose = FALSE)
```



```
polCA4<-polCA(f,data = votingdata +1,nclass = 4,graphs = TRUE,na.rm = TRUE,maxiter
= 1000,verbose = FALSE)
```



*#The AIC and BIC can be compared and the optimal model can be concluded.*

```
cat("The AIC and BIC values of model with class 2 is: ",polCA2$aic, " & ", polCA2$bic," respectively")
```

```
## The AIC and BIC values of model with class 2 is: 950.0332 & 989.6813 respectively
```

```
cat("The AIC and BIC values of model with class 3 is: ",polCA3$aic, " & ", polCA3$bic," respectively")
```

```
## The AIC and BIC values of model with class 3 is: 937.0493 & 998.0465 respectively
```

```
cat("The AIC and BIC values of model with class 4 is: ",polCA4$aic, " & ", polCA4$bic," respectively")
```

```
## The AIC and BIC values of model with class 4 is: 943.7951 & 1026.141 respectively
```

```
# Since the BIC value of class =2 is lower compared with other class 3 and 4, we can see that the class 2 is optimal.
```

The output shows that the AIC and BIC of class = 2 is optimal and the voting percentage is also clearly depicted by the plot in polCA2 model. So, the data for this dataset can be split into 2 clusters for achieving an optimal model.

1 (c) Compare the clustering from the polytomous analysis 1(b) to the clustering obtained by cutting the dendrogram you view as optimal from (a).

The hierarchical clustering using 'Average' method shows that the data can be split into two clusters. This is shown by cutting the tree at 0.7 height. This will show the dendrogram with two clusters.

Also from the Polytomous analysis, we come into conclusion that the model with class = 2 is an optimal model. These two data can be compared and the clustering can be confirmed using "classAgreement".

```
#The dendrogram is cut using cutree and is used for comparing with polytomous analysis.  
votinghcl2<-cutree(votingavg2, k = 2)  
  
#Library(e1071)  
  
#The output of the cutree is stored in votinghcl2 which is compared with the vector of predicted class memberships and then used for external validation.  
tab<-table(votinghcl2,polCA2$predclass)  
classAgreement(tab)
```

```
## $diag  
## [1] 0.006410256  
##  
## $kappa  
## [1] -0.8469294  
##  
## $rand  
## [1] 0.9871795  
##  
## $crand  
## [1] 0.9742328
```

The output of external validation with adjusted RandIndex value as 0.97 shows that the cluster values derived from both the methods is optimal.

1 (d) Write a report outlining your research and conclusions, based on your LCA analysis.



## Voting patterns in the 32nd Dail Eireann

Ireland is a parliamentary democracy and consists of the President and two houses, Dail Eireann (House of Representatives) and Seanad Eireann (the Senate). The members of Dail Eireann, called Teachtaí Dála (TDs), are directly elected by the people. During the 5th session of the 32nd Dail Eireann, the votes of TDs are recorded in a dataset and is used for analyse the voting pattern of the TDs.

The voting patten of the TDs can be analyzed by clustering the TDs and splitting the groups with similar voting patterns. This clustering can be carried out using Latent Class Analysis (LCA) approach and votes pattern can be derived using Polytomous Analysis.

### Latent Class Analysis approach

Latent Class Analysis(LCA) is a statistical technique that can be used for analysis of multivariate categorical data. It analyzes the clustering the observations in a multi-way tables of these categorical variables. LCA is part of “poLCA” package in R and it has several functions that can be used for clustering and calculating the posterior and likelihood probabilities of the variables.

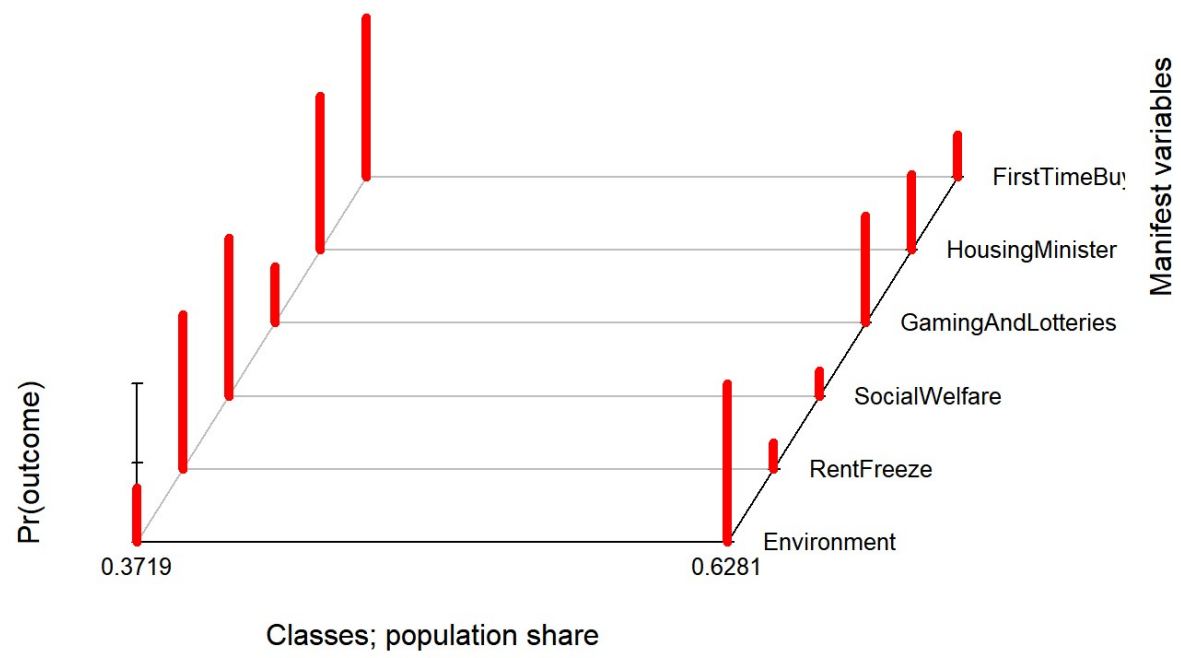
One such function of LCA is ‘poLCA’. Polytomous Latent Class Analysis(poLCA) is a function in LCA package and is uded in predicting the latent class memberships by one or more covariates. poLCA estimates the latent class model by maximizing the log-likelihood function using the Expectation-Maximization (EM) algorithm. It estimates the standard errors of the estimated class-conditional response probabilities and mixing parameters. It also calculates the parsimony measures while estimating the latent class model. These parameters include Akaike information criterion (AIC) and Bayesian information criterion (BIC), which are used for analyzing goodness of fit. Preferred models are those that minimize values of the BIC and/or AIC. The BIC will usually be more appropriate for basic latent class models because of their relative simplicity.

### Modelling the dataset

The voting pattern of 32nd Dail Eireann can be estimated using this polytomous analysis, by fitting the models with serveral classes and arriving to an optimal class. The model is fit using poLCA function with variables formula, data, nclass, graphs. Data variable will have the dataset assigned and nclass represents the number of latent classes to assume in the model. Variable graphs depicts the probability of variables with respect to number of latent classes.

```
## The AIC and BIC values of model with class 2 is: 950.0332 & 989.6813 respect  
ively.  
## The AIC and BIC values of model with class 3 is: 937.0493 & 998.0465 respec  
tively.  
## The AIC and BIC values of model with class 4 is: 943.7951 & 1026.141 respec  
tively.
```

From the values, we can see that the BIC value of class =2 is lower compared with other class 3 and 4, so, the model with class 2 gives an optimal result. So, we can use the latent class model with nclass as 2 for estimating the voting pattern of TDs and examining the membership of the clusters.



```

## Conditional item response (column) probabilities,
## by outcome variable, for each class (row)
##
## $Environment
##           Pr(1) Pr(2)
## class 1:  0.655 0.345
## class 2:  0.000 1.000
##
## $RentFreeze
##           Pr(1) Pr(2)
## class 1:  0.0261 0.9739
## class 2:  0.8316 0.1684
##
## $SocialWelfare
##           Pr(1) Pr(2)
## class 1:  0.0000 1.0000
## class 2:  0.8369 0.1631
##
## $GamingAndLotteries
##           Pr(1) Pr(2)
## class 1:  0.6429 0.3571
## class 2:  0.3235 0.6765
##
## $HousingMinister
##           Pr(1) Pr(2)
## class 1:  0.0330 0.9670
## class 2:  0.5214 0.4786
##
## $FirstTimeBuyers
##           Pr(1) Pr(2)
## class 1:  0.0000 1.0000
## class 2:  0.7348 0.2652
##
## Estimated class population shares
##  0.3719 0.6281
##
## Predicted class memberships (by modal posterior prob.)
##  0.359 0.641
##
## =====
## Fit for 2 latent classes:
## =====
## number of observations: 156
## number of estimated parameters: 13
## residual degrees of freedom: 50
## maximum log-likelihood: -462.0166
##
## AIC(2): 950.0332
## BIC(2): 989.6813
## G^2(2): 58.95794 (Likelihood ratio/deviance statistic)
## X^2(2): 67.89259 (Chi-square goodness of fit)
##

```

The summary of model 2 shows the probability of each class or cluster for its respective categorical variable. From the graph, we can see that the predicted class memberships is 0.641 and 0.359. We can compare the predicted class of each categorical variable with the political affiliation of each TD and then interpret the voting pattern of each group.

##	AAA-PBP	FF	FG	Green	I4C	Ind	Lab	SD	SF
## 1	0	0	48	0	1	7	0	0	0
## 2	6	43	1	3	2	11	7	3	24

From the above table, we can see that all the members of 'AAA-PBP', 'FF', 'Green', 'Lab', 'SD', 'SF' are all contained in class 1 of the latent class model and members of 'FG', 'I4C' and 'Ind' are split among both the classes with 'FG' falling almost completely on class 2. This proves their predicted class memberships of 0.641 & 0.359 respectively.

**Interpretation of the estimated optimal model** By interpreting the estimated class conditional response probabilities for each categorical variable, we can see that, 100% of the members in class 1 votes 'Yes' for bills related to 'Environment' when compared to the voting ratio of class 2 members which is approximately 35%. For 'Rent freeze bill', almost 84 % of members of class 1 voted 'No' when compared with members of second cluster(class 2), which was only 2%. So, TDs of class 2 were all positive (approx 98%) for 'Rent Freeze'. Similarly, for 'Social Welfare' bill, majority of class 1 members, opposed the bill (approx 84%) where as all the class 2 members voted 'Yes' for 'Social Welfare'.

'Gaming and Lotteries' bill was welcomed by class 1 members since approximately 68% of them voted 'Yes' for this bill when compared to class 2 members who majorly opposed the bill (approx 64% of TDs from class 2). Almost 52% of class 1 members opposed the bill related to 'Housing Minister' and rest voted yes. In class 2, almost all the members supported the 'Housing Minister' as approximately 97% of them voted 'Yes' for it. For 'First Time Buyers' bill, approx. 74% of members of class 1 voted on a motion of no confidence. But this bill was completely welcomed by class 2 members as all the members (100%) voted 'Yes' for 'First Time Buyers'.

## Conclusion

The Latent Class Analysis, a model based approach, works well with categorical variables that are binary in nature. For the voting dataset, an optimal model is constructed using polCA function and the data is segregated into two clusters. The optimal model for the dataset is achieved by parsimony measure, Bayesian information criterion (BIC). From the clusters, we see that most of the TDs fall under first cluster and they account to majority decisions making on each bill.

## Appendix

```

#A formula expression of the form response ~ predictors
f<-cbind(Environment,RentFreeze,SocialWelfare,GamingAndLotteries,HousingMinister,FirstTimeBuyers)~1

# Three models are developed based on the classes and their BIC are compared to summarize the optimal model. Parameter Verbose can be used to restrict the function to print all the output to the screen.
poLCA2<-poLCA(f,data = votingdata +1,nclass = 2,graphs = FALSE,na.rm = TRUE,maxiter = 1000,verbose = FALSE)
poLCA3<-poLCA(f,data = votingdata +1,nclass = 3,graphs = FALSE,na.rm = TRUE,maxiter = 1000,verbose = FALSE)
poLCA4<-poLCA(f,data = votingdata +1,nclass = 4,graphs = FALSE,na.rm = TRUE,maxiter = 1000,verbose = FALSE)

#The AIC and BIC can be compared and the optimal model can be concluded.
cat("The AIC and BIC values of model with class 2 is: ",poLCA2$aic, " & ", poLCA2$bic," respectively. \n","The AIC and BIC values of model with class 3 is: ",poLCA3$aic, " & ", poLCA3$bic," respectively.\n","The AIC and BIC values of model with class 4 is: ",poLCA4$aic, " & ", poLCA4$bic," respectively.")

#The party data is added and is used for comparing the class data
TDnames<-read.csv("TDs_names_parties.csv",header = T)

#Predicted class membership and the party data is compared to check the number of TDs added to each class.
table(poLCA2$predclass,TDnames$Party)

```

## Question 2

A sample of 30 wild growing and flowering *Hyptis suaveolens* plants were collected in El Salvador, and the concentrations of 7 terpenes (sabinene,  $\alpha$ -pinene, 1.8-cineole,  $\beta$ -terpinene, fenchone,  $\gamma$ -terpinolene and fenchol) were measured (data *Hyptis.csv* available in Brightspace). Interest lies in uncovering different chemotypes of plant (if any) using the terpene measures. The geographical region from which each plant was collected was also recorded, either north, south or east. For the eastern plants, a distinction was made whether the plants grew at low or high altitude.

The *Hyptis* dataset is loaded to a data frame 'hyptisdata'. This data frame will be used for applying various Multidimensional scaling methods.

```
hyptisdata<-read.csv("Hyptis.csv",header = T)
```

2 (a) Apply classical metric scaling to the data, explaining any decisions you make in the process. Choose a suitable number of dimensions required to represent the resulting configuration. Explain, using a graphic to support your argument, your reasoning behind your choice of the required number of dimensions. Plot the two dimensional configuration resulting from the application of classical metric scaling. Label each point in your plot using the geographical location of each plant.

For applying Classical Metric Scaling, we have to first calculate the dissimilarity matrix and use `cmdscale` for calculating the eigen values and then use it determining the dimensions.

```

# Computing the Dissimilarity Matrix
D<- dist(hyptisdata[,1:7], method="euclidean")

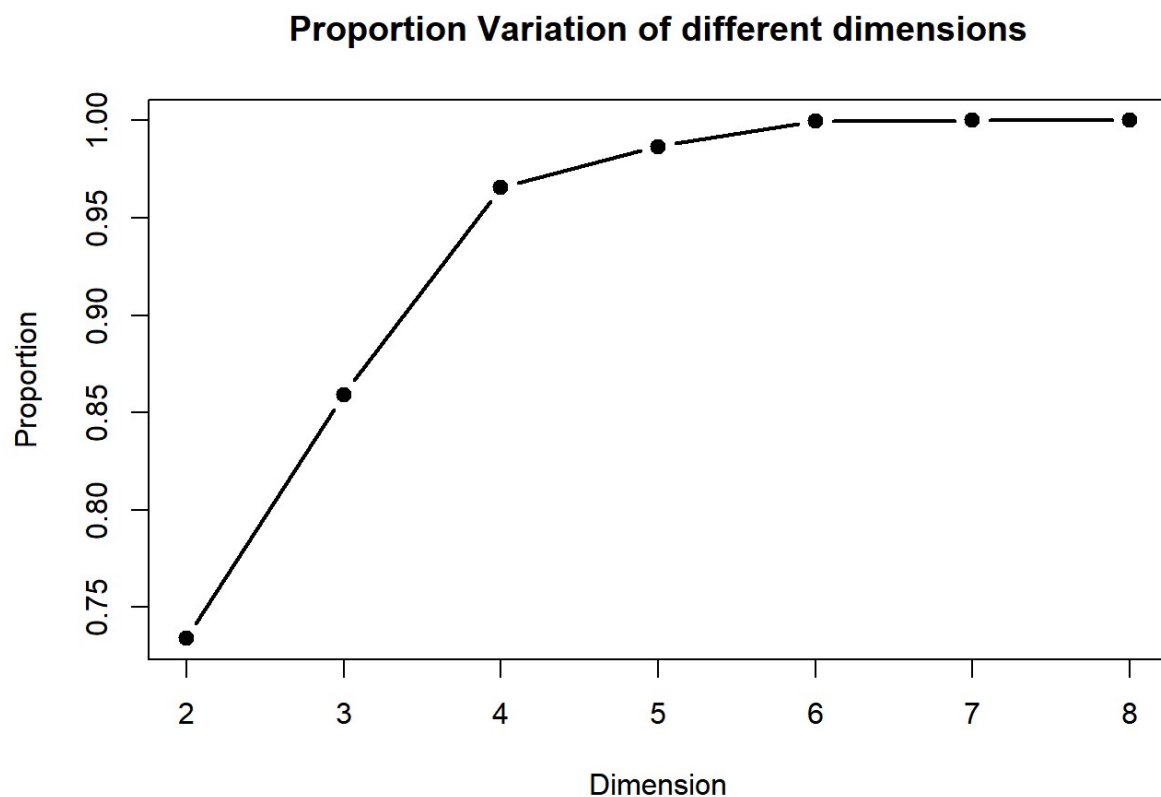
#metric classical scaling for q=2
mds2<-cmdscale(D,k=2,eig = TRUE)

#proportion variation for storing several dimensions
variation <- c()

for(i in 2:8){
  variation <- c(variation,sum(abs(mds2$eig[1:i]))/sum(abs(mds2$eig)))
}

#plotting of the variance with respect to their dimensions
plot(2:8,variation,main="Proportion Variation of different dimensions", xlab="Dimension", ylab="Proportion", type="b",lwd=2, pch=19)

```

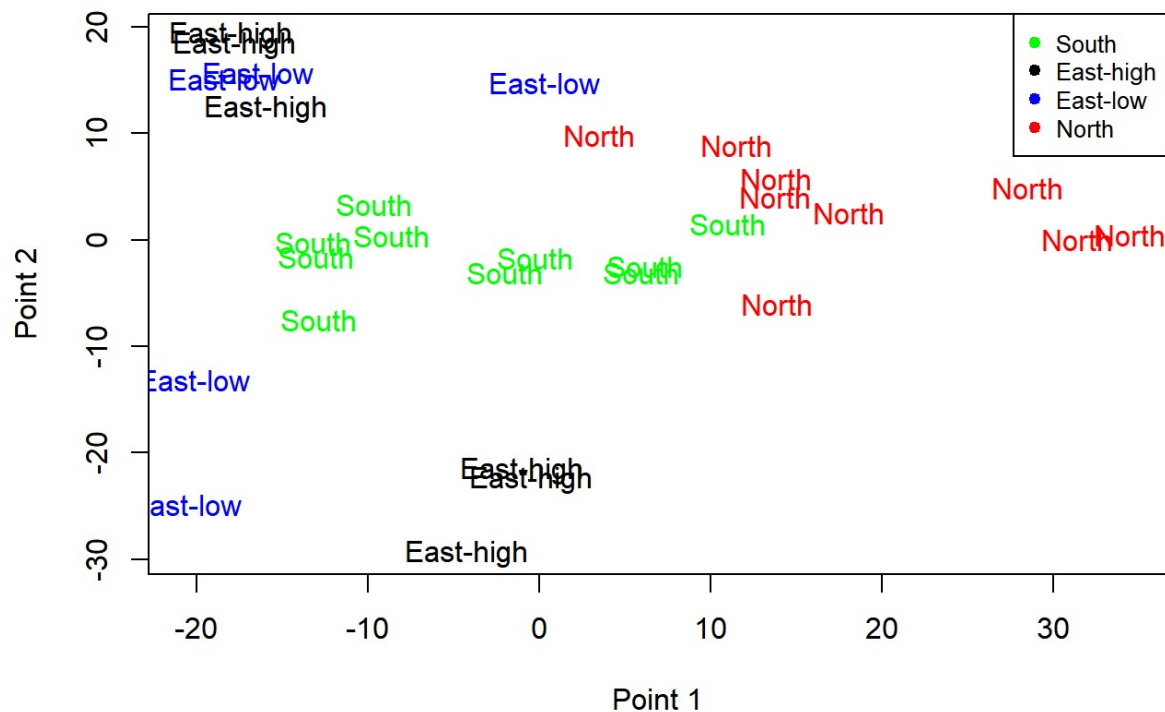


While comparing the proportion of each dimension  $q$ , we see that there is not much of variation between  $q=2$  with other  $q$  proportions. So, we can choose the dimension as 2 for classical metric dimension.

```
plot(mds2$points[,1],mds2$points[,2],type = "n",xlab = "Point 1",ylab = "Point 2",main="Classical metric Scaling")
text(mds2$points[,1],mds2$points[,2],hyptisdata[,8],cex=1,col = c("black","blue","red","green")[hyptisdata[,8]])

legend("topright", legend=unique(hyptisdata[,8]),
      col=c("green","black","blue","red"), pch=19, cex=0.8)
```

## Classical metric Scaling



2 (b) Apply Sammon's metric least squares scaling and Kruskal's non-metric scaling to the data. Overlay the resulting two dimensional configurations on your plot of the classical scaling configuration. Label each point using the observation number of the associated plant.

The Sammon's metric least squares scaling and Kruskal's non-metric scaling can be applied using the 'Sammon' function and 'isoMDS' function respectively of MASS package.

```
#Sammon stress variable
sammonstress<-c()

#Kruskal stress variable
kruskalstress<-c()
k<-8
for(i in 1:k){
  sammonstress<-c(sammonstress,sammon(D,k=i)$stress)
  kruskalstress<-c(kruskalstress,isoMDS(D,k=i)$stress)
}
```



```
## Initial stress          : 0.26140
## stress after 10 iters: 0.16214, magic = 0.500
## stress after 20 iters: 0.16212, magic = 0.500
## initial value 40.484914
## final value 35.156076
## converged
## Initial stress          : 0.06787
## stress after 10 iters: 0.02647, magic = 0.500
## stress after 20 iters: 0.02612, magic = 0.500
## stress after 30 iters: 0.02546, magic = 0.500
## stress after 40 iters: 0.02464, magic = 0.500
## stress after 50 iters: 0.02446, magic = 0.500
## stress after 60 iters: 0.02445, magic = 0.500
## initial value 19.099008
## iter 5 value 18.422428
## iter 10 value 17.550163
## iter 15 value 12.874160
## iter 20 value 12.185171
## iter 25 value 12.021814
## iter 25 value 12.016702
## iter 25 value 12.015710
## final value 12.015710
## converged
## Initial stress          : 0.02787
## stress after 10 iters: 0.01018, magic = 0.500
## stress after 20 iters: 0.00891, magic = 0.500
## stress after 30 iters: 0.00752, magic = 0.500
## stress after 40 iters: 0.00726, magic = 0.500
## stress after 50 iters: 0.00641, magic = 0.500
## stress after 60 iters: 0.00621, magic = 0.500
## stress after 70 iters: 0.00616, magic = 0.500
## initial value 11.588897
## iter 5 value 8.245610
## final value 8.211046
## converged
## Initial stress          : 0.00216
## stress after 10 iters: 0.00081, magic = 0.461
## stress after 20 iters: 0.00068, magic = 0.500
## stress after 30 iters: 0.00068, magic = 0.500
## initial value 2.008442
## iter 5 value 1.479018
## iter 10 value 1.335232
## iter 15 value 1.310778
## final value 1.303042
## converged
## Initial stress          : 0.00040
## stress after 10 iters: 0.00010, magic = 0.500
## stress after 20 iters: 0.00009, magic = 0.500
## initial value 0.850290
## iter 5 value 0.626153
## iter 10 value 0.522565
## iter 15 value 0.456890
```

```

## iter 20 value 0.423685
## iter 25 value 0.403645
## iter 30 value 0.388613
## iter 35 value 0.378595
## iter 40 value 0.374870
## final value 0.373954
## converged
## Initial stress      : 0.00000
## stress after 10 iters: 0.00000, magic = 0.150
## initial value 0.012549
## final value 0.005433
## converged
## Initial stress      : 0.00000
## stress after 9 iters: 0.00000
## initial value 0.000000
## final value 0.000000
## converged
## Initial stress      : 0.00000
## stress after 10 iters: 0.00000, magic = 0.150
## initial value 0.000000
## final value 0.000000
## converged

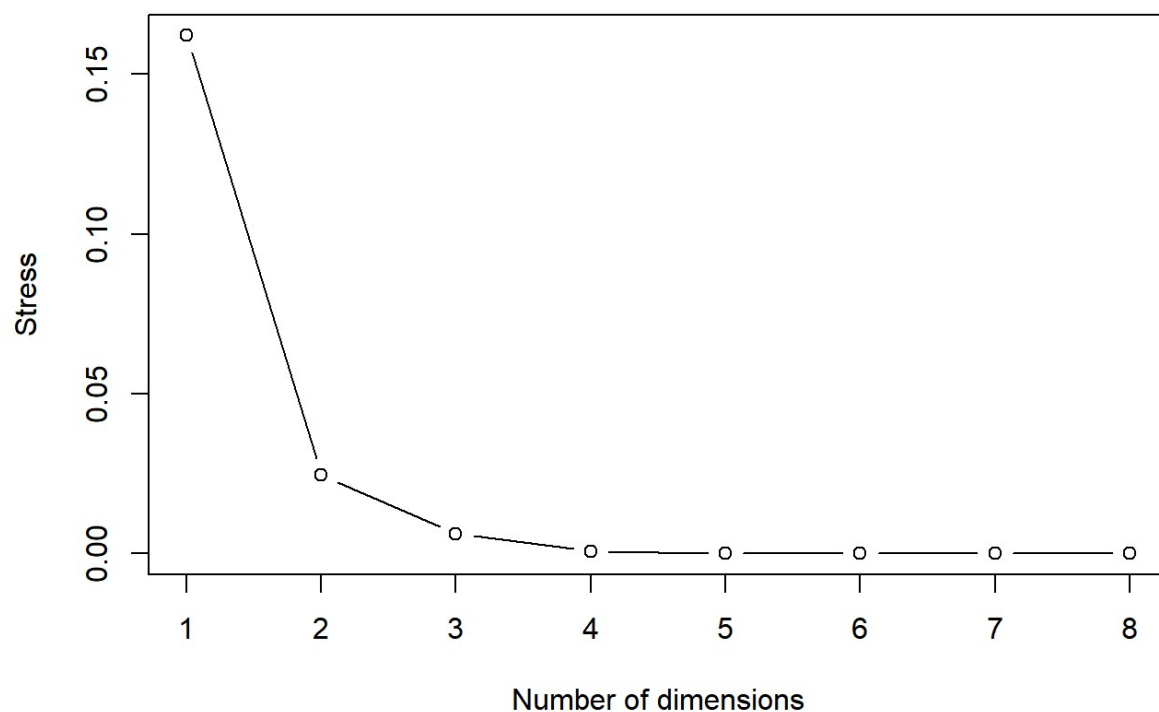
```

```

#Plotting the stress of Sammon metric Least squares scaling
plot(x=c(1:k),y=sammonstress,type = "b", xaxp=c(1,k, k-1), ylab="Stress", xlab="Number of dimensions",main = "Sammon Stress for 8 Dimensions")

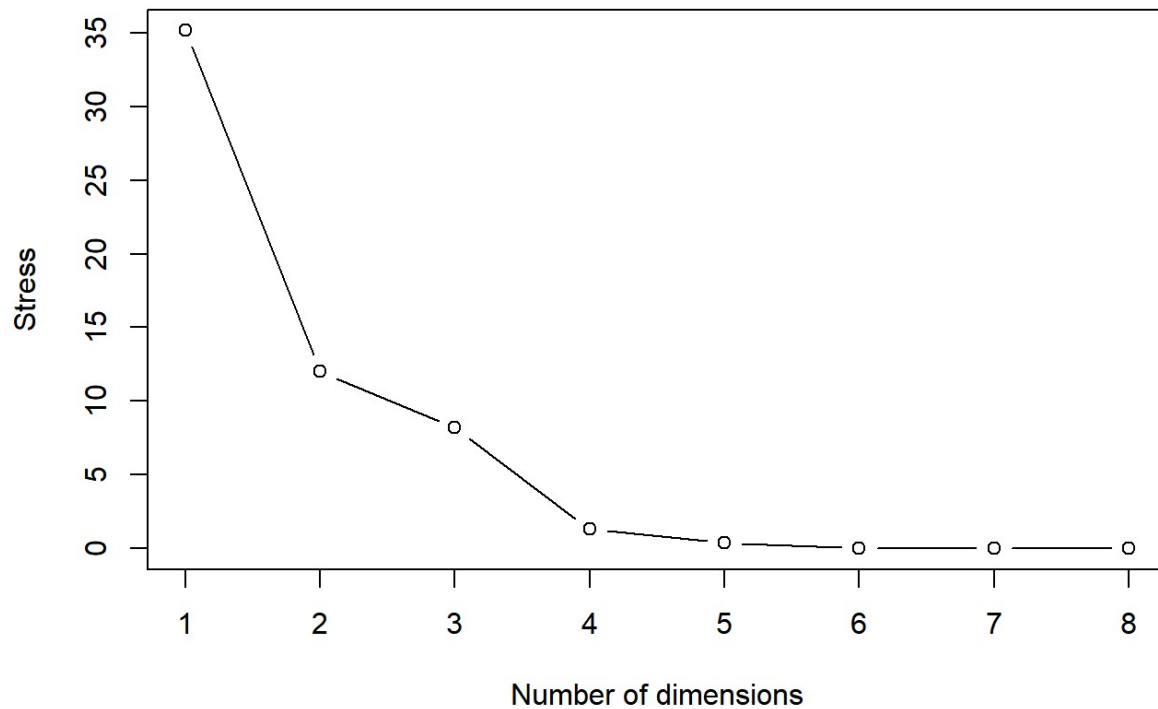
```

### Sammon Stress for 8 Dimensions



```
#Plotting the stress of Kruskal's non-metric scaling
plot(x=c(1:k),y=kruskalstress,type = "b", xaxp=c(1,k, k-1), ylab="Stress", xlab="Number of dimensions",main = "Kruskal Stress for 8 Dimensions")
```

### Kruskal Stress for 8 Dimensions



From the plot we can see that both the scaling methods points a steep reduction in stress at  $q = 2$ , i.e., in two dimension. The stress gradually decreases after this dimension. So, we can use the dimension as 2 and apply the scaling methods on the data.

```
# Applying Sammon and Kruskal function to the data with q=2
Sammon2<-sammon(D,k=2)
```

```
## Initial stress      : 0.06787
## stress after 10 iters: 0.02647, magic = 0.500
## stress after 20 iters: 0.02612, magic = 0.500
## stress after 30 iters: 0.02546, magic = 0.500
## stress after 40 iters: 0.02464, magic = 0.500
## stress after 50 iters: 0.02446, magic = 0.500
## stress after 60 iters: 0.02445, magic = 0.500
```

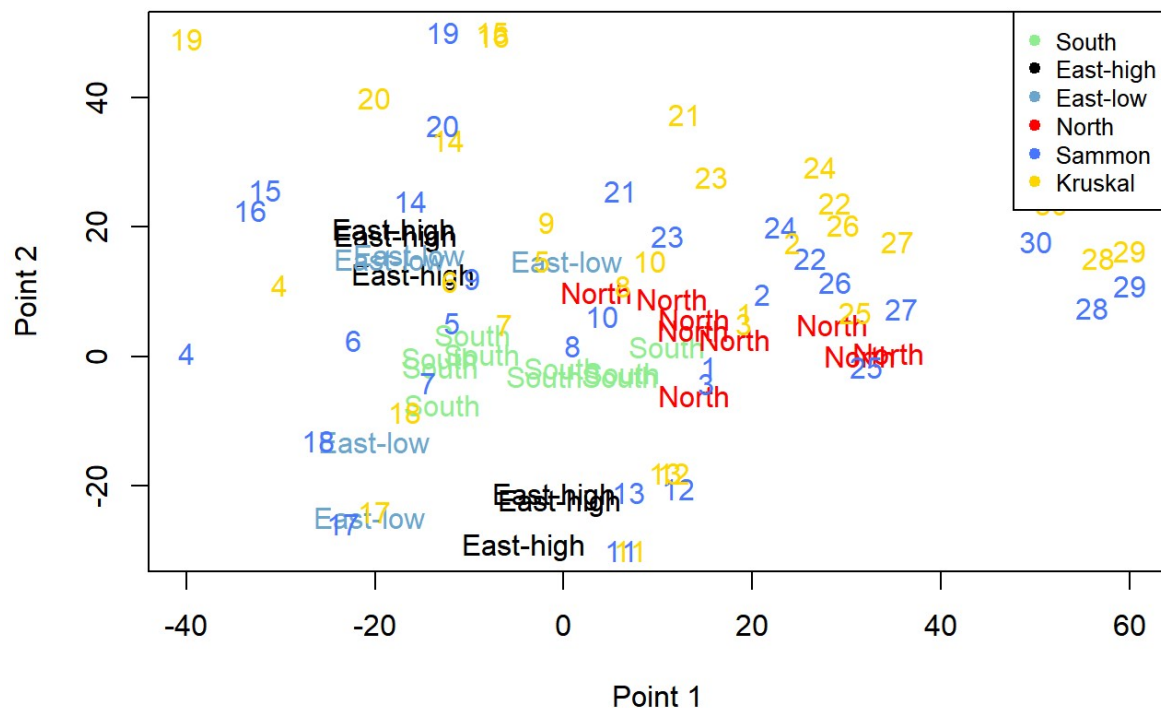
```
Kruskal2<-isoMDS(D,k=2)
```

```
## initial value 19.099008
## iter 5 value 18.422428
## iter 10 value 17.550163
## iter 15 value 12.874160
## iter 20 value 12.185171
## iter 25 value 12.021814
## iter 25 value 12.016702
## iter 25 value 12.015710
## final value 12.015710
## converged
```

*# The calculated values are then plotted on the plot of classical scaling configuration*

```
plot(mds2$points[,1],mds2$points[,2],type = "n",xlim = c(-40,60),ylim = c(-30,50),
xlab = "Point 1",ylab = "Point 2",main="Multi Dimentional Scaling of Hyptis")
text(mds2$points[,1],mds2$points[,2],hyptisdata[,8],cex=1,col = c("black","skyblue
3","red","palegreen2")[hyptisdata[,8]])
par(new=TRUE,col.axis = "transparent")
plot(Sammon2$points[,1],Sammon2$points[,2],type = "n",xlab = "",ylab = "",axes=FALSE)
text(Sammon2$points[,1],Sammon2$points[,2],row.names(hyptisdata),cex=1,col = "royalblue1")
par(new=TRUE,col.axis = "transparent")
plot(Kruskal2$points[,1],Kruskal2$points[,2],type = "n",xlab = "",ylab = "",axes=FALSE)
text(Kruskal2$points[,1],Kruskal2$points[,2],row.names(hyptisdata),cex=1,col = "gold1")
legend("topright", legend=c(as.character(unique(hyptisdata[,8])), "Sammon", "Kruskal1"),
      col=c("palegreen2","black","skyblue3","red","royalblue1","gold1"), pch=19, cex=0.8)
```

## Multi Dimensional Scaling of Hyptis



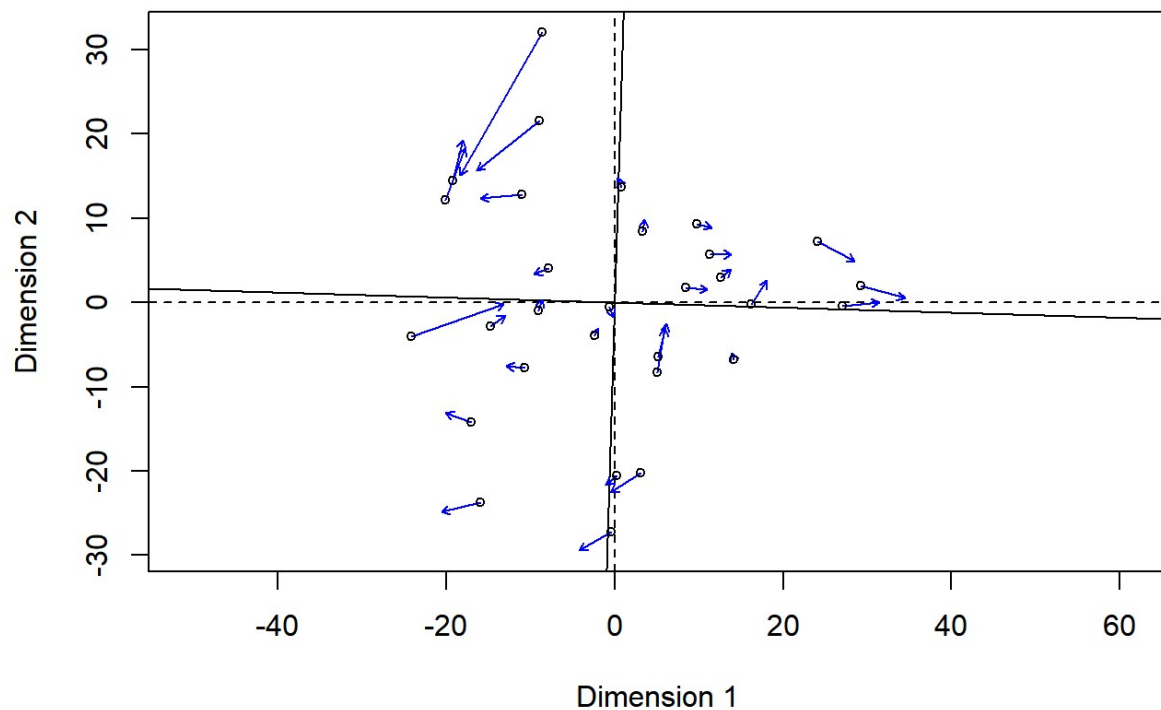
2 (c) Use Procrustes analysis to match the three resulting configurations of the plants. Which configurations match best? Suggest a reason for your conclusion

Procrustes Analysis is a technique that matches one configuration to another and produces a measure of the match.

```
# Matching the configurations of Classical, Sammon and Kruskal scaling method.
proc12<-procrustes(mds2$points,Sammon2$points)
proc23<-procrustes(Sammon2$points,Kruskal2$points)
proc31<-procrustes(Kruskal2$points,mds2$points)

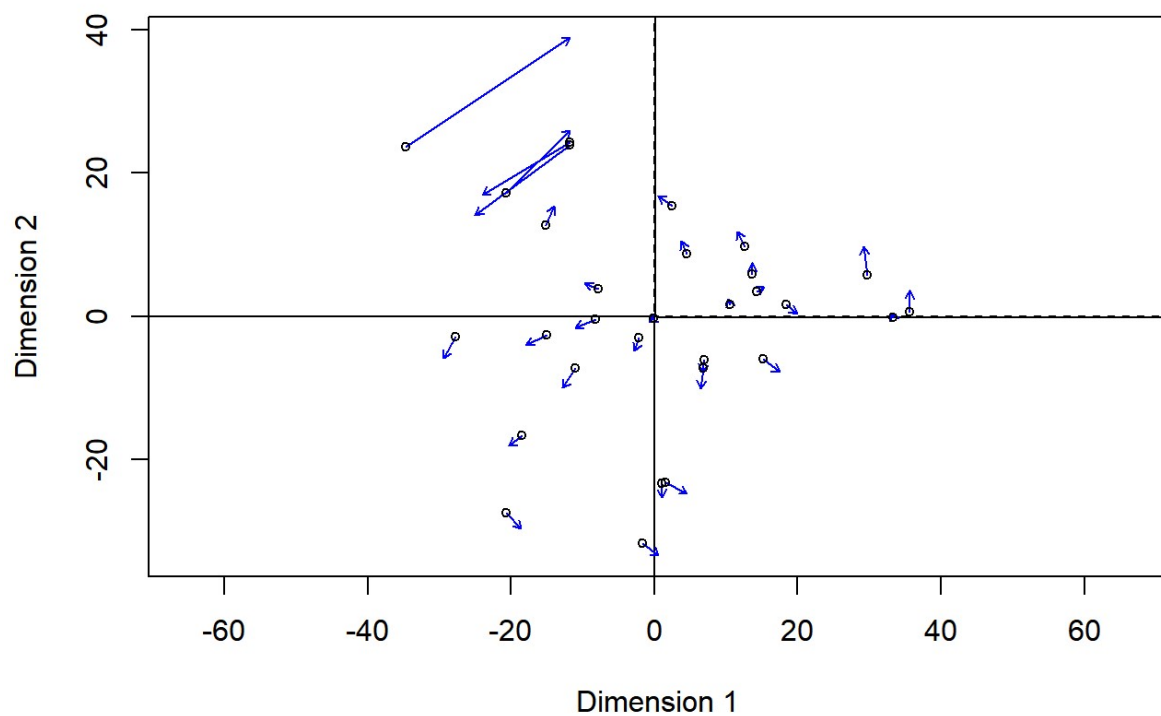
# Plotting each match with procrustes error between the two configuration
plot(proc12,xlab = "Dimension 1",ylab = "Dimension 2", main = "Procrustes Errors for Classical MDS vs Sammon MDS on Hyptis data")
```

## Procrustes Errors for Classical MDS vs Sammon MDS on Hyptis data



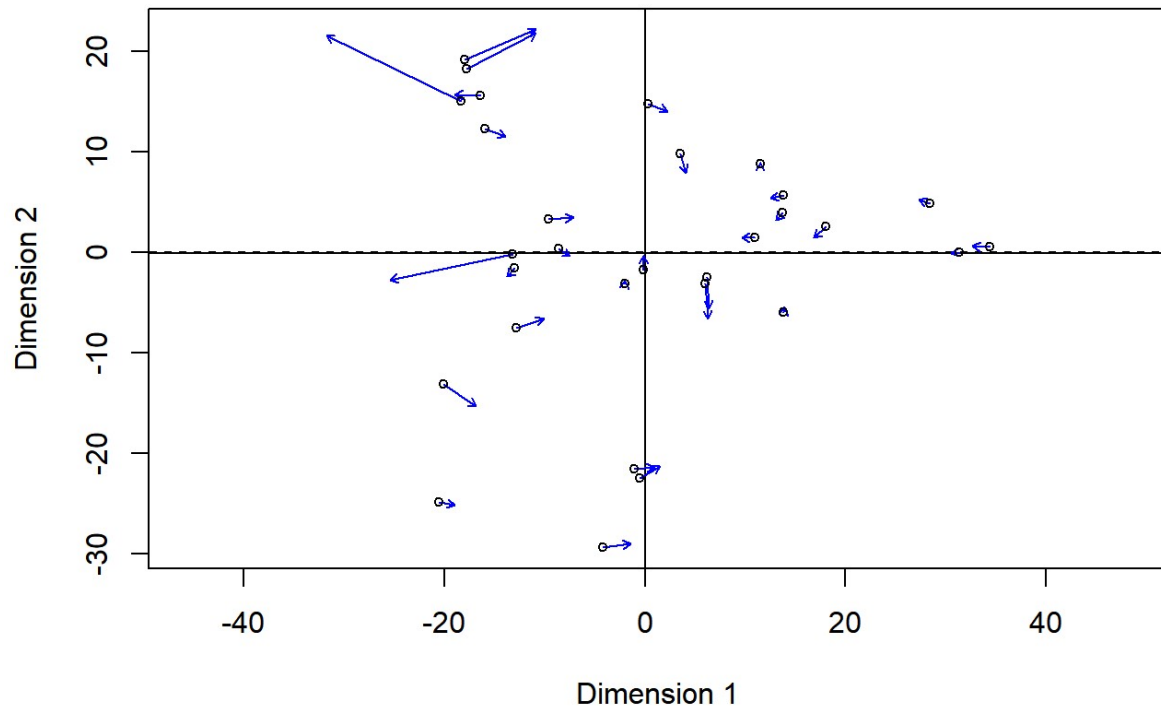
```
plot(proc23,xlab = "Dimension 1",ylab = "Dimension 2", main = "Procrustes Errors for Sammon MDS vs Kruskal MDS on Hyptis data")
```

## Procrustes Errors for Sammon MDS vs Kruskal MDS on Hyptis data



```
plot(proc31,xlab = "Dimension 1",ylab = "Dimension 2", main = "Procrustes Errors for Kruskal MDS vs Classical MDS on Hyptis data")
```

### Procrustes Errors for Kruskal MDS vs Classical MDS on Hyptis data

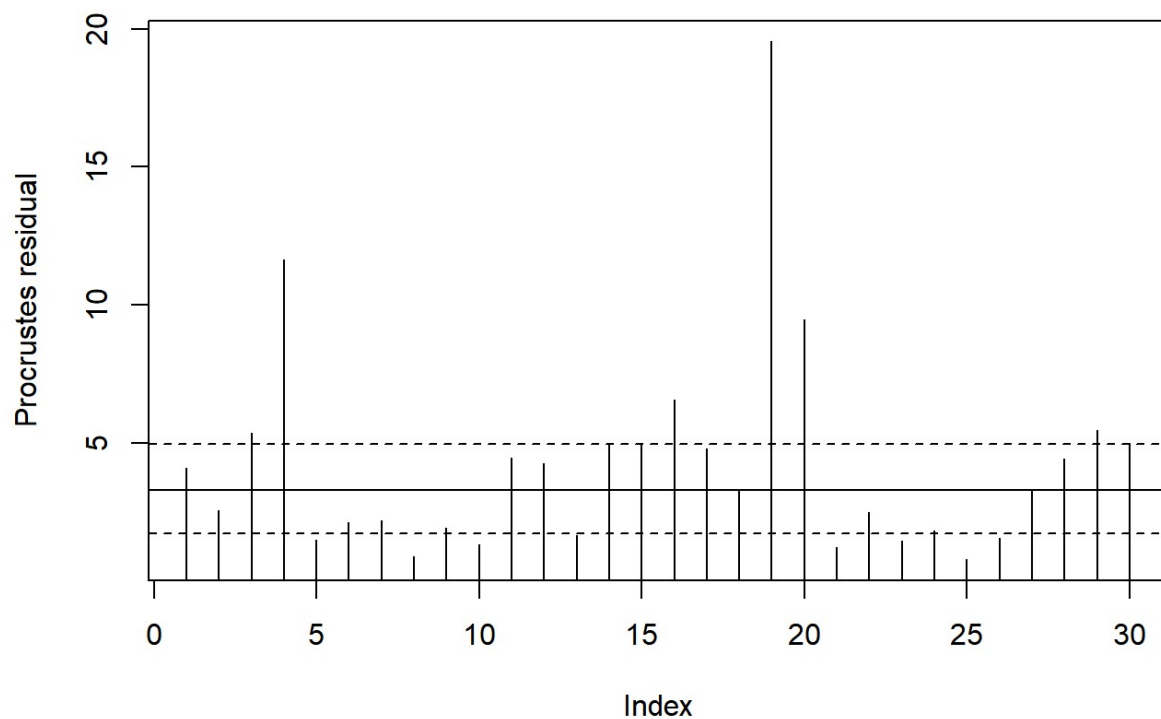


From the above plots, we see that the points in all three configurations has procrustes Errors in them. But Kruskal configuration has less procrustes error compared with other plots. So it can be considered as a good configuration. We can confirm it by verifying the Residual errors aswell.

```
# Procrustes residual errors
plot(proc12,kind = 2, main = "Procrustes Errors for Classical MDS vs Sammon MDS on Hyptis data")
```

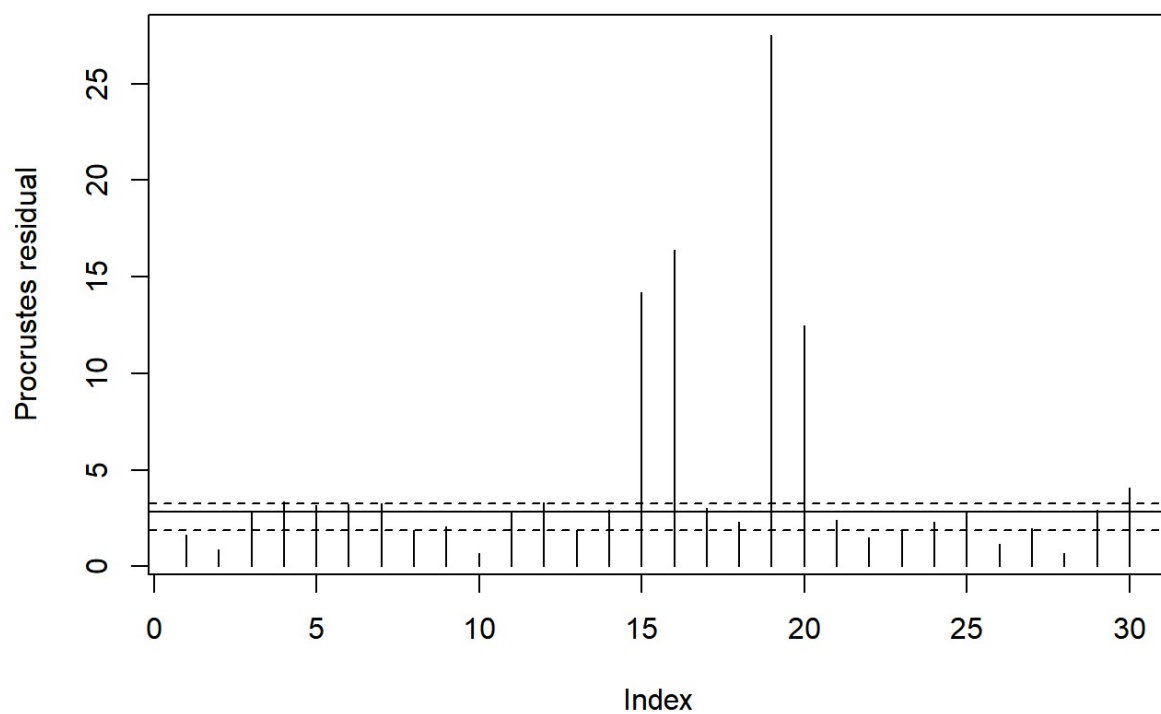


### Procrustes Errors for Classical MDS vs Sammon MDS on Hyptis data

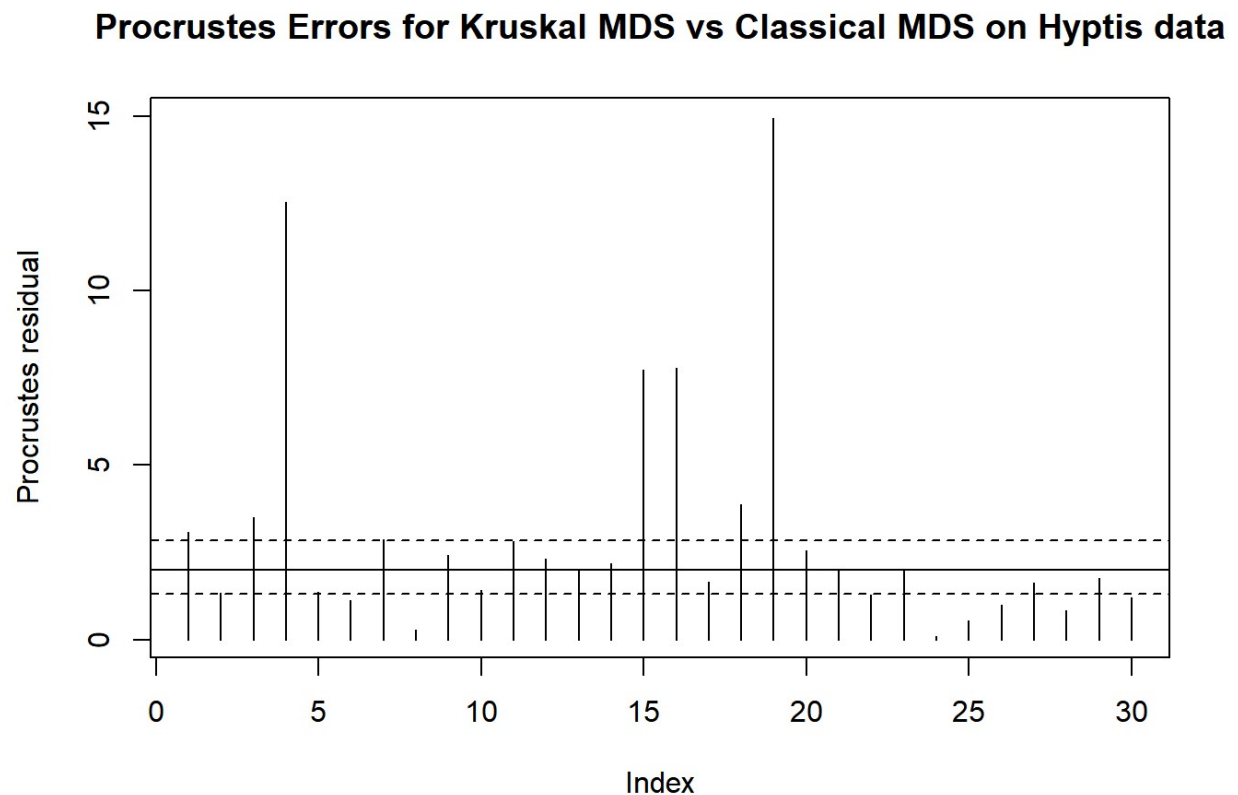


```
plot(proc23,kind = 2, main = "Procrustes Errors for Sammon MDS vs Kruskal MDS on Hyptis data")
```

### Procrustes Errors for Sammon MDS vs Kruskal MDS on Hyptis data



```
plot(proc31, kind = 2, main = "Procrustes Errors for Kruskal MDS vs Classical MDS on Hyptis data")
```



From the Procrustes Residual plots, we can see that the Kruskal versus Classical MDS plot has less Procrustes residual error compared with other plots. From this, we can suggest that Kruskal MDS configuration fits better compared with the other two configurations.

2 (d) Perform model-based clustering on these data. How many clusters are chosen as optimal? Explain how you arrived at your answer.

The Model Based Clustering can be done using `mclust` function.

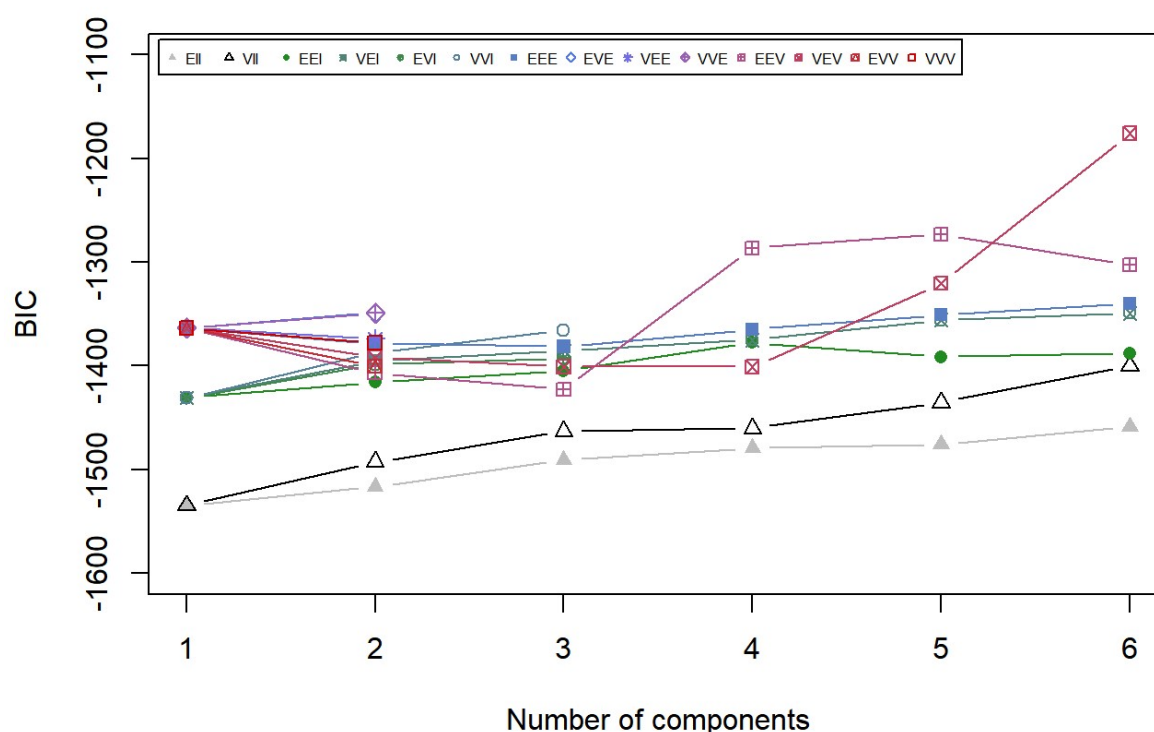
```
res1<-Mclust(hyptisdata[,1:7],G=1:6)
summ<-summary(res1)

res1$BIC
```

```
## Bayesian Information Criterion (BIC):
##      EII      VII      EEI      VEI      EVI      VVI      EEE
## 1 -1534.671 -1534.671 -1430.731 -1430.731 -1430.731 -1430.731 -1363.418
## 2 -1516.394 -1492.713 -1416.033 -1395.465 -1398.577 -1387.067 -1378.315
## 3 -1490.860 -1463.129 -1404.612 -1385.553 -1392.777 -1365.842 -1381.151
## 4 -1479.260 -1460.120 -1377.685 -1374.947      NA      NA -1364.810
## 5 -1475.484 -1435.108 -1391.356 -1355.610      NA      NA -1350.639
## 6 -1458.831 -1400.271 -1387.737 -1349.284      NA      NA -1339.974
##      EVE      VEE      VVE      EEV      VEV      EVV      VVV
## 1 -1363.418 -1363.418 -1363.418 -1363.418 -1363.418 -1363.418 -1363.418
## 2 -1348.799 -1373.693 -1349.190 -1406.801 -1391.472 -1400.549 -1377.730
## 3      NA      NA      NA -1422.666 -1400.862      NA      NA
## 4      NA      NA      NA -1286.380 -1400.790      NA      NA
## 5      NA      NA      NA -1273.187 -1320.322      NA      NA
## 6      NA      NA      NA -1302.723 -1175.816      NA      NA
##
## Top 3 models based on the BIC criterion:
##      VEV,6      EEV,5      EEV,4
## -1175.816 -1273.187 -1286.380
```

From the BIC values, we can see that G=6 is the optimal group(cluster) in the function. We can confirm this by plotting the values to a graph.

```
plot(res1,what = "BIC",legendArgs=list(x="topleft",cex=0.5,horiz=T),ylim = c(-1600, -1100))
```



This plot also suggests that the VEV model with G = 6, has the lowest BIC value among the other

models.

```
#Cross Tabulation  
table(hyptisdata[,8],summ$classification)
```

```
##  
##           1 2 3 4 5 6  
## East-high 0 0 3 3 0 0  
## East-low  2 1 0 0 2 0  
## North     0 6 0 0 0 3  
## South     8 1 1 0 0 0
```

```
#Calculating the adjusted Rand Index  
adjustedRandIndex(hyptisdata[,8],summ$classification)
```

```
## [1] 0.4403259
```

From the cross tabulation and adjusted Rand Index value, we can see that the data points are fairly clustered and has a Adjusted Rand Index value as 0.44. Therefore, the optimal clusters for this dataset is 6.