Big data Statistics
Exercise Set II

Efthymios Ioannis Kavour

June 25, 2023

Contents

1 Exercise 1 **3**

2 Exercise 2 **5**

3 Exercise 3 **17**

1 Exercise 1

(Gene expression measurements). Consider the Ramaswamy dataset available from

http://www-stat.stanford.edu/hastie/glmnet/glmnetData/,

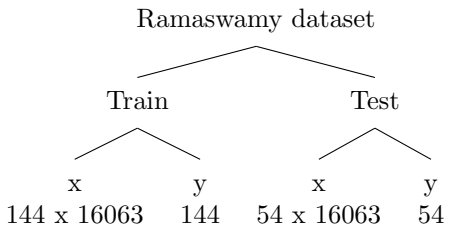
which contains 16063 gene expression measurements and 198 samples belonging to 14 distinct cancer subtypes. The aim is to construct a rule for predicting **cancer types** based on gene expression measurements. Evaluate the predictive performance of your proposed classifier by cross-validation.

Solution:

In order to complete this exercise, we will need the following packages :

- tidyverse
- e1071
- class
- tree
- randomForest
- HDclassif

By taking a quick look at our data, we can see that the initial format is a list that contains two sub-lists and each containing a matrix x (with the explanatory variables, *the genes*) and a vector y (that is the dependent variable, *cancer type*).



Our goal as stated in the definition of this exercise is to predict cancer types based on the dataset provided, as accurate as possible. Since we have labels (cancer type) of every observation, we are going to use Classification methods in order to find a preferred rule. In next steps we are going to implement the following algorithms :

- Naive Bayes (package : *e1071*)

- K Nearest Neighbors (package : *class*)
- Decision Tree (package : *tree*)
- Support Vector Machine (package : *e1071*)
- Random Forest (package : *randomForest*)
- High Dimensional Classification (package : *HDclassif*)

By using the afforementioned methods and while the data have been scaled, we have acquired the following results:

Classification Methods	
Method	Accuracy
Naive Bayes	42.59%
KNN	42.59%
Decision Trees	38.89%
SVM	48.15%
Ranbom Forest	55.56%
HDClassif	61.11%

We can see that the highest results are gained by High Dimensional Classification. We know that **Naive Bayes** Classification is a fine choice for large datasets but its assumption of features independent may not hold in all cases, which can affect its accuracy, as we can see in our case as well.

K Nearest Neighbors has several advantages, including simplicity and adaptability to different data distributions. However, it can be sensitive to irrelevant or noisy features, and its computational complexity increases as the dataset size grows which may affect accuracy and computational time. We may include in our preprocessing analysis, a deeper search for noise and correlation amongst the information provided in order to raise accuracy.

Decision Trees have advantages such as interpretability, handling both categorical and numerical features, and capturing non-linear relationships. On the other hand decision trees can be sensitive to data patters which results to overfitting and may struggle with complex relationships or datasets with many features which holds for our case. As one without much thinking would expect decision trees to have a better accuracy result.

Apart from decision tress one would expect **Support Vector Machines** to produce a somehow better result. But in order for support vector machine to produce better result, we need to tune its parameters (kernel etc).

As **Ranbom Forest** depends on producing many decision trees, and then take that information under consideration I think it is obvious why the results are better than Decision Trees. However one should not forget that the disadvantages of decision trees follow random forest up to a point. Apart from that it can be really expensive as we produce more and more decision trees.

High Dimensional Classification provides to us the best possible result given the low preprocessing level done. This is obvious as we have set the

argument *model* in *hdda* to "all" which implies that the function will compute all possible models based on

- Akj are the parameters of the classes subspaces
- Bk are the noises of the classes subspaces
- Qk is the orientation matrix of each class
- Dk is the intrinsic dimension of each class

Afterwards it will calculate its BIC and return to us the one with the highest BIC model, so we don't have to check its and everyone of them.

2 Exercise 2

(Cluystering coffee samples). The coffee dataset available on the *pgmm* package contains data measuring the chemical composition of coffee samples collected from arounf the world, comprising 43 samples from 29 countries. Each sample is either of the Arabica or Robusta variety. The first two columns contain "Variety" and "Country, respectively. The purpose is to cluster the dataset (without of course taking the ground-truth classification into account) based on the chemical properties measures in the last 12 columns of the data frame. The number of clusters is assumed unknown. For this purpose you may use distance-based methods, partition methods as well as model-based clustering methods. For the model-based clustering methods use standard mixture of normals (*mclust*) as well as factor analytics ones (*pgmm, fabMix*) considering that the number of factors is at most equal to 2 (for pgmm use both random starting values as well as k-means starting values with option : *zstart = 1* and *zstart = 2*, respectively). Compare the estimated clustering and the ground-truth partition of the data in terms of the adjusted Rand Index. Obtain the dataset using the commands:

```
library(pgmm)
data(coffee)
x <- coffee[,-c(1,2)]
x <- scale(x)
```

(the input data for clustering is x)

Solution:

The libraries we are going to use in order to complete this assessment are the following:

- pgmm
- tidyverse

- corrplot
- stats
- cluster
- mclust

Initially, as requested, we are going to insert the data, and perform a scaling on the data. After those stpes are completed we are going to perform some simple Exploratory Data Analysis (EDA) in order to investigate what we have imported and understand he information we are going to work with. As a result we have the following.

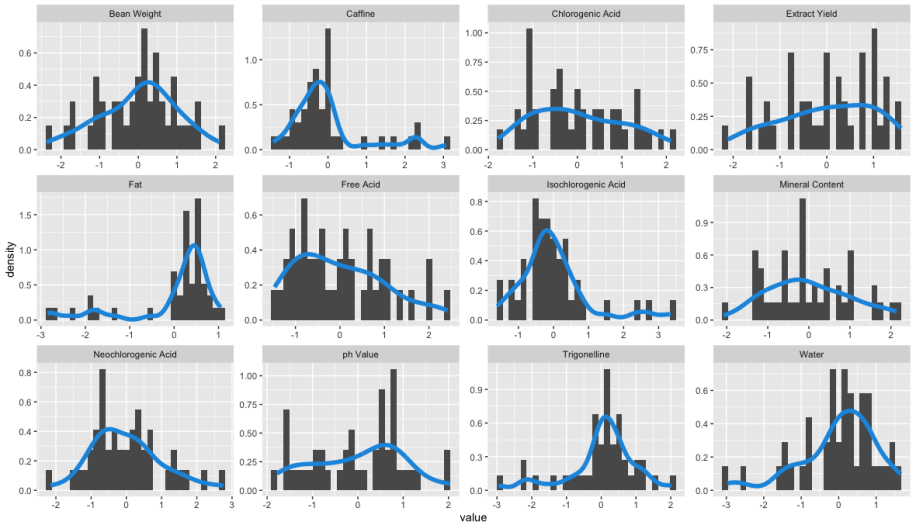


Figure 1: Distribution of variables in coffee dataset

From the plot generated above, we can see that the distribution of each variable may vary and is not (obviously) the same for every feature contained. As for the correlation of each variable in the dataset with each other, we have the following.

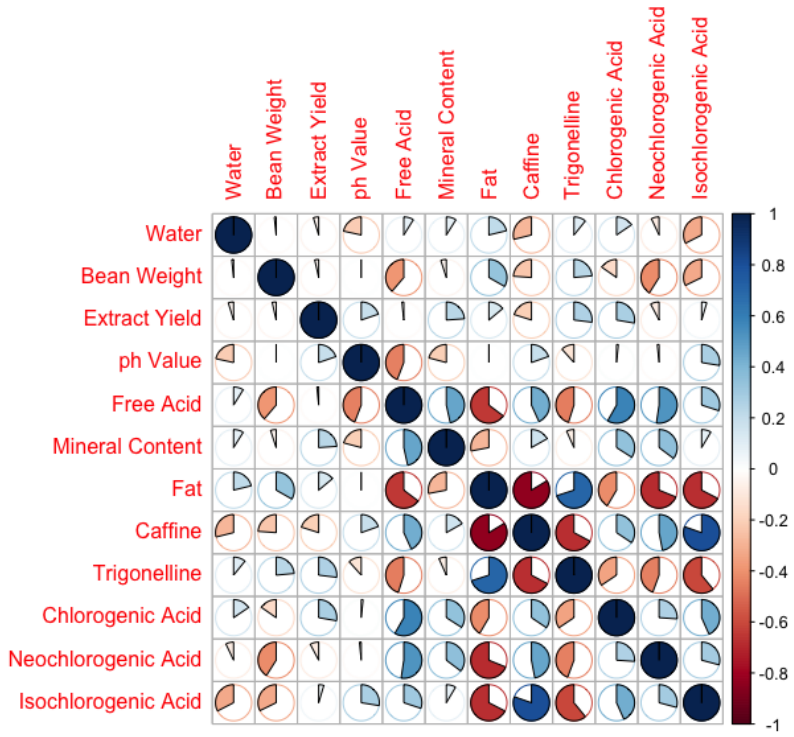


Figure 2: Correlation of variables in coffee dataset

We can see that the variables FAT - Caffine, Fat - Neochlorogenic Acid,FAT - Isochlorogenic Acid and Trigonelline - Isochlorogenic Acid seem to have high negative correlation, whereas, Caffine and Isochlorogenic Acid seem to have high positive correlation.

In order to implement Hieratchical clustering (by using different linkages) we need to calculate the distances of each variable. Initially, we are going to calculate Euclidean Distance and by moving forward, we are going to calculate other distances in order to compare and check if a specific calculated distances works better than other.

Distances are calculated with the help of the function *dist* with a selectionn of the specific method required. For example in order to calculated the euclidean distance, we need to run the following command :

```
dist(x, method = "euclidian")
```

After having calculated the distances with the preferred method, we can proceed to implement hierarchical clustering with the help of the function *hclust* and the *plot* the result acquired. With no further ado, we decided to use all posible linkage methods (*single*, *complete*, *average*, *ward.D*, *ward.D2*, *mcquitty*, *median*, *centroid*). After having calculated everything, we have the following

results :

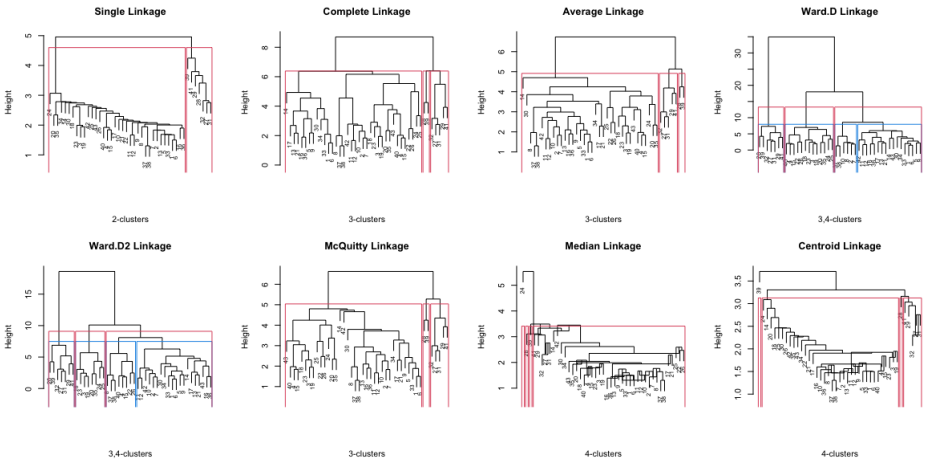


Figure 3: Hierarchical clustering using different linkage methods, with Euclidean Distances used

Let us try to comment on the results, provided above. We know that the optimal number of groups is 2, as we have **Arabica** and **Robusta** variety. With red squares there have been marked the first possible split distinguished by the clustering method. For some there have been a second pair of rectangles (the blue ones), which are the next possible splitting point. The second pair of rectangles have been added in order to check how an overfilled group would be divided.

Initially, one can distinguish without much concentration, that by using Complete, Average, McQuitty, Median, Centroid linkage methods, we have certain clusters with only one observation which does not seem to be in our case correct. As a result, we are confident to drop these cases. Moving forward, we can see that by using Ward.D and Ward.D2 linkage methods, we get 3 clusters, one of which contains a greater number of observations. We are not going to drop these two but they are not going to be my favourite choices. As for the Single linkage method, we can see that there are initially two clusters, though the number of observations are not balanced within the groups. Balance is not something we aim to get, nor it should be. We know that we aim to get two groups so at this point Single linkage method seems to work the best. Of course we are going to use certain tests in order to get a decision upon which is the best but given the knowledge we have until now, Single linkage methods is our go-to clustering method.

We repeat the process by using *Manhattan* distances. We have the following results.

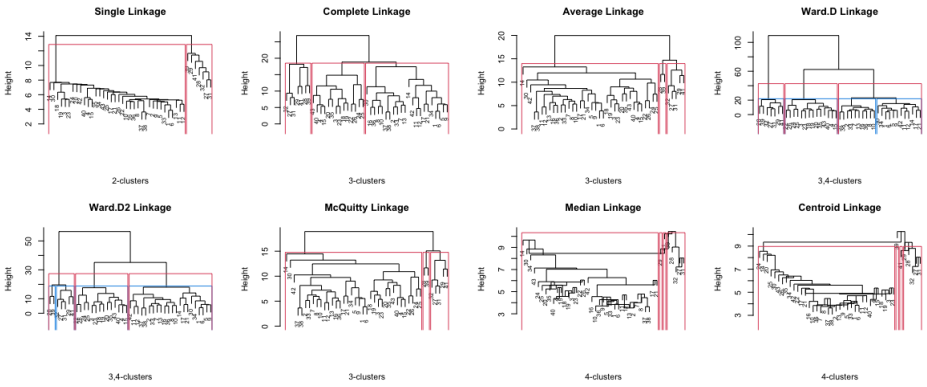


Figure 4: Hierarchical clustering using different linkage methods, with Manhattan Distances used

We can initially, drop the one by using Average, McQuitty, Median, Centroid linkage methods for the same reason as before. Furthermore, we can see that Single linkage method has divided the observations into two groups, which fits our case. On the other hands, we can clearly see that Complete, Ward.D and Ward.D2 has returned a 3 way division. I have decided to provided also the next level of division for the last two linkage methods, as the third of the 3 groups generated appears to contain many more observations.

Lastly we are going to repeat the process by using Minkowski distances with the proceeding results :

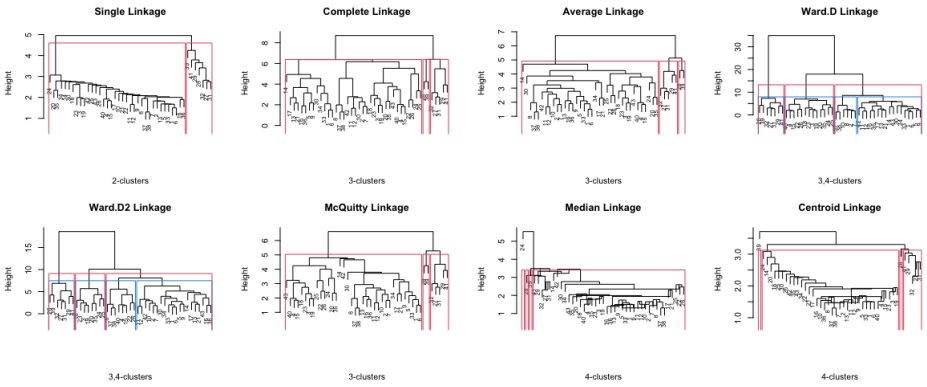


Figure 5: Hierarchical clustering using different linkage methods, with Minkowski Distances used

For this group of plots, we can initially say that Complete, Average, Mc-

Quitty, Median, Centroid have groups containing only one observation and this is a case that we do not want. Moving forward, once again Single linkage method divides our group of data into two subplots and once again Ward.D and Ward.D2 divided that into 3 subplots.

Now given the methods, we have used, we are going to use the function *cutree* and produce Contingency tables for each case give the ground truth in order to see in depth our results with the help of function *table*. To sum up by using Euclidean distance we will use, Single, Ward.D and Ward.D2 linkage methods, Manhattan distance we will use Singe, Complete, Ward.D and Ward.D2 linkage methods and finally for the Minkowski Method, we shall use Single, Ward.D and Ward.D2 method.

Euclidean Distance								
Single			Ward.D			Ward.D2		
	1	2		1	2		1	2
1	36	0	1	23	0	1	28	0
2	0	7	2	13	0	2	8	0
			3	0	7	3	0	7

Manhattan Distance												
Single			Complete			Ward.D			Ward.D2			
	1	2		1	2		1	2		1	2	
1	36	0	1	22	0	1	22	0	1	22	0	
2	0	7	2	14	0	2	14	0	2	14	0	
			3	0	7	3	0	7	3	0	7	

Minkowski Distance								
Single			Ward.D			Ward.D2		
	1	2		1	2		1	2
1	36	0	1	23	0	1	28	0
2	0	7	2	13	0	2	8	0
			3	0	7	3	0	7

We can see that Euclidean and Minkowski distances provided the exact same results, for the selected linkage methods. But first things first. With blue color, the ground truth is indicated, and with red the groups generated by the clustering methods used. Here we are going to comment on the first distance used (Euclidean) as the rest follow the same way of interpretation. For the Single linkage method used, the algorithm gather 36 observations in a grouped and named that 1 and 7 in the second group and named that 2. As for the ground truth, there is the exact division of the observation which is a good result to stick to. For the Ward.D link-method we can see that it divided the first group into two subgroups where it placed 23 observations in a group together and named this group 1 and the other 13 to a second group (namely group 2) The third group contains only 7 observations which are indicated inn group 2 as for the ground truth. So here we can see that the Hierarchical clustering method

decided to divide (ground truth) group 1 into two subgroups. The same case is for Ward.D2 but instead of splitting group one into 23 and 13 observations, it divided it in 28 observations to a grouped which is names 1 and 8 to a second group which is named 2.

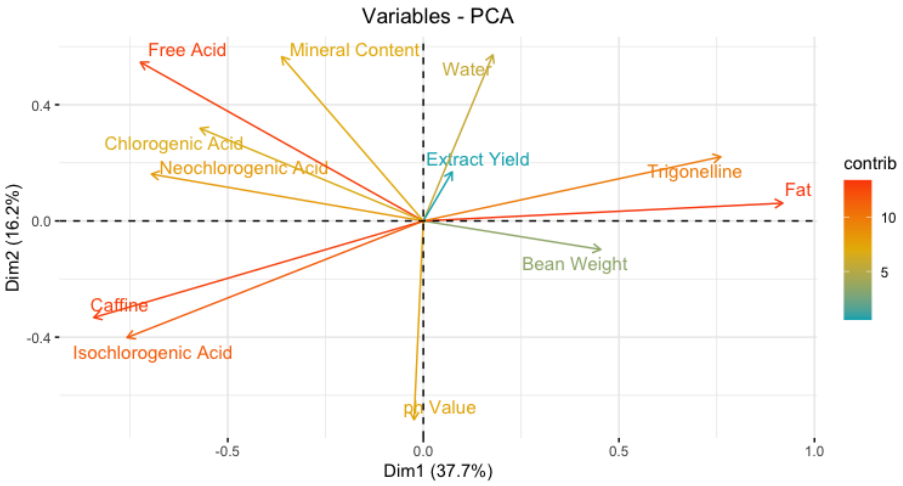
A final check is going to be implemented by using the Adjusted Rand Index (ARI) in order to get a score of the clustering chosen and make up our minds.

Adjusted Rand Index				
Distance	Single	Complete	Ward.D	Ward.D2
Euclidean	1	-	0.397	0.515
Manhattan	1	0.383	0.383	0.383
Minkowski	1	-	0.397	0.515

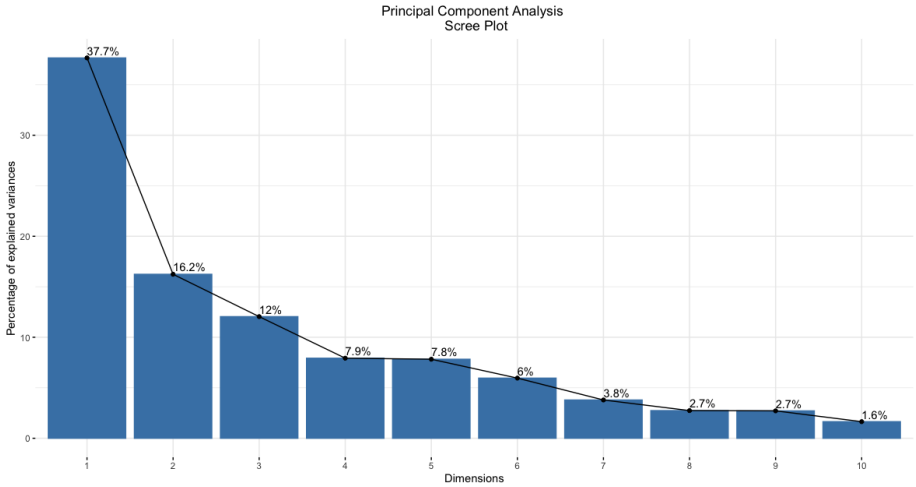
As expected, we can see that clustering that have as outcome 2 subgroups (single link-method), outperform the rest ones according to the results generated by using ARI.

As it is quite hard to visualize 12 variables we are going to use Principal Components method in order to be able, hopefully, to use a 2-d axis or at most 3-d axis, to visualize our clustering outcomes. Initially, we are going to use *princomp* function in order to check how many components' variability we need to check to explain/interprate our data.

```
    Importance of components:
                           Comp.1    Comp.2    Comp.3
Standard deviation     2.1008321 1.3795089 1.1881160
Proportion of Variance 0.3765482 0.1623629 0.1204358
Cumulative Proportion  0.3765482 0.5389112 0.6593470
                           Comp.4     Comp.5     Comp.6
Standard deviation     0.96404668 0.95777485 0.83552640
Proportion of Variance 0.07929285 0.07826449 0.05956049
Cumulative Proportion  0.73863983 0.81690433 0.87646482
                           Comp.7     Comp.8     Comp.9
Standard deviation     0.66667008 0.56683219 0.56418252
Proportion of Variance 0.03791926 0.02741239 0.02715671
Cumulative Proportion  0.91438408 0.94179647 0.96895318
                          Comp.10    Comp.11     Comp.12
Standard deviation     0.43922679 0.30014429 0.284413172
Proportion of Variance 0.01645946 0.00768596 0.006901402
Cumulative Proportion  0.98541264 0.99309860 1.000000000
```

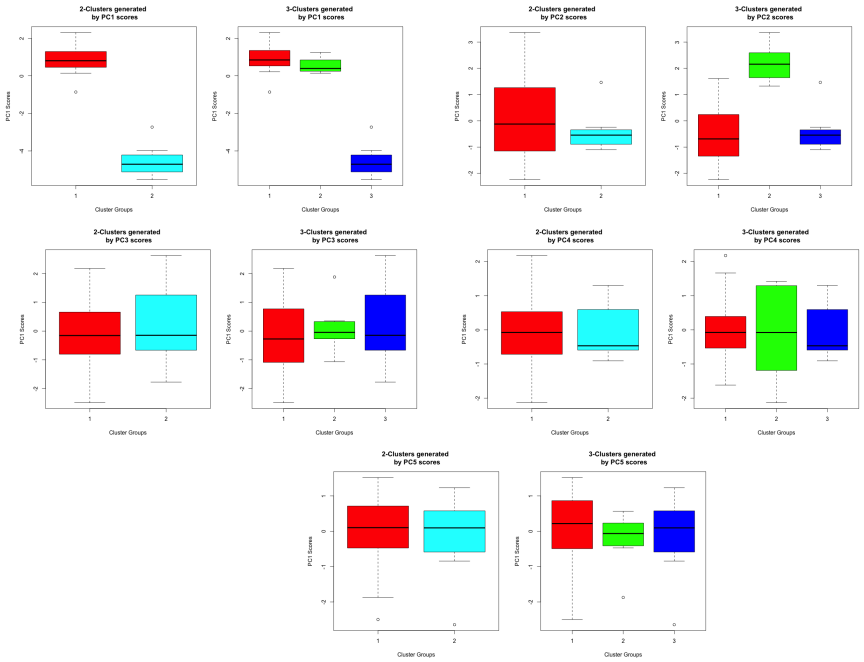


Note that : Here we can also see the correlations noted during the sort EDA part.

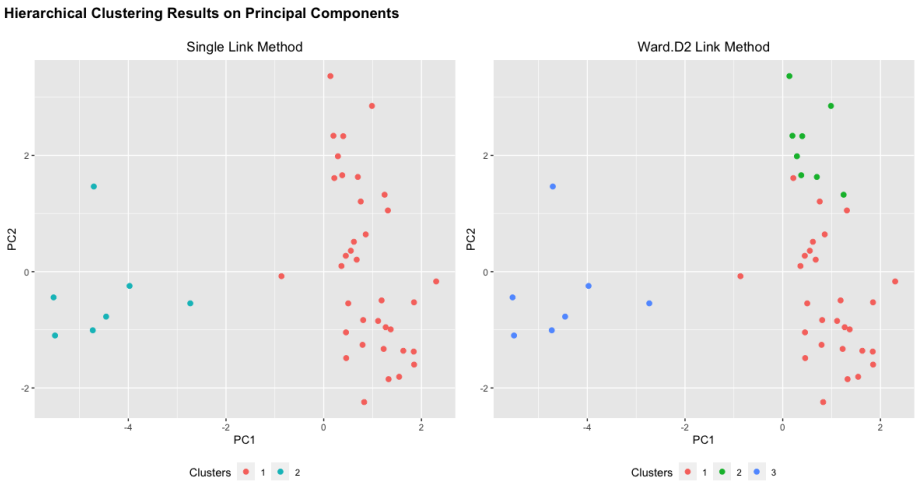


After acquiring the results presented above we can see that the first 5 components explain about 87.6% of the data and have standard deviation of almost almost 1 or greater. As a result, we decide that we need to consider Components 1 to 5. We are going to use the clusters generated by Euclidean Distance by using Single and Ward.D2 linkage methods. The reason is that Single link-method gives us the preferred number of groups and on the other hand Ward.D2 is the one with the highest ARI after that.

The results of the Principal Components amongst the groups are the following:

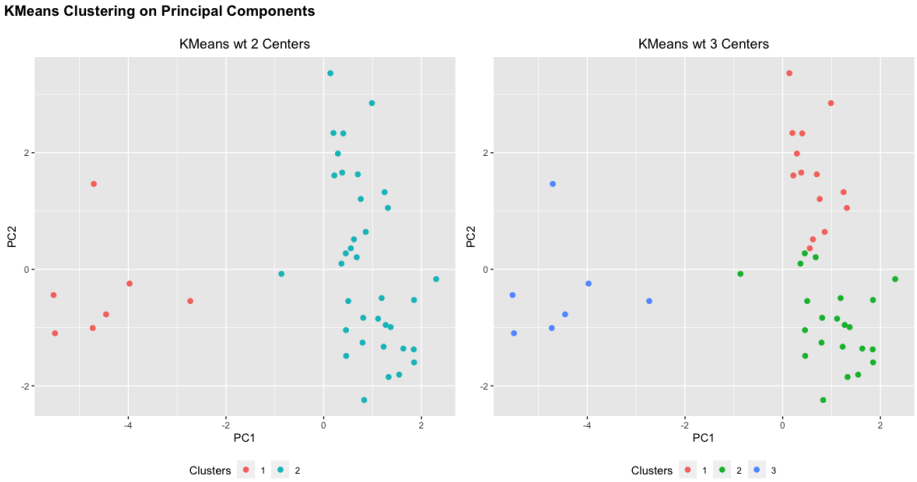


From the plots generated above for the first five principal components, we can see that there is great variability amongst the groups for the first two principals. The rest (PC3, PC4, PC5) do not seem to differ among the 3 groups generated from hierarchical clustering. As a result, we are going to proceed to our analysis (thankfully) by using Component 1, and Component 2. Now we can visualize the selected clustering results and try to visualize them by using the help of the selected components. In this point it is wise to remember that the selected components explain 53.8% of the observations' variability. As a resultwe have the following results:



We can now see, even if, at this point, we had no idea of ground truth, that the division of groups 1,2 (right plot) has little to no meaning.

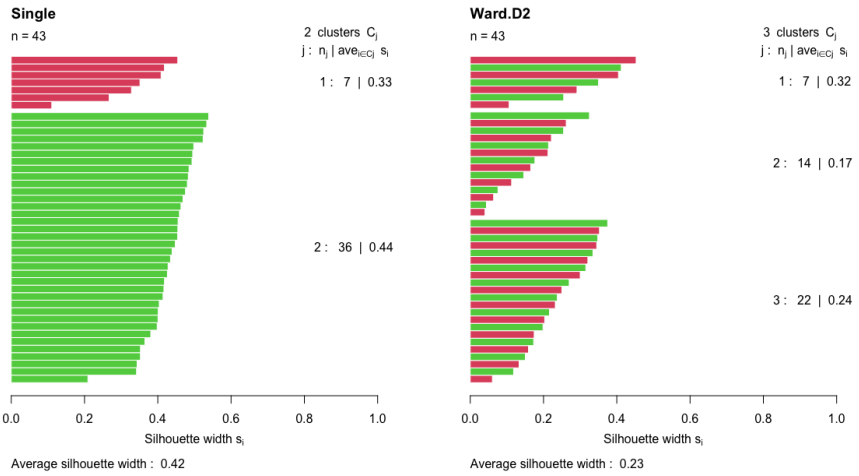
In this next part of our analysis, we are going to take advantage of Principal Components selected and use K-Means clustering method. In this part we are going to use function *kmeans*. The results acquired after settinng the method on run are



We can see that in the case that we have selected initially two centers, KMeans algorithm returns to us clusters well separated, whereas when we use three centers we can distinguish some overlapping on the clusters generated, which makes this case a bit shady and rises doubts. By checking the ARI for this case we can see that

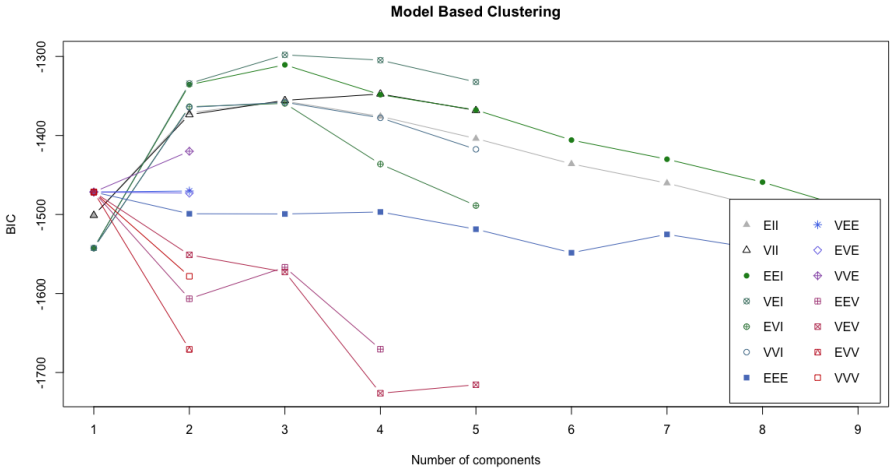
K-Means ARI	
Single	1
Ward.D2	0.383

The result acquired now is even less than 0.515 upon which this linkage method was selected. One other way to compare kmeans results is by using silhouette score.



We get an indicator here as well, when it comes to what number of groups should we go with. We can see here that Silhouet Score for 3 groups is 0.23 which is almost half the Silhouette score for 2 groups.

As a final part of our analysis, of the coffee dataset, we are going to proceed with model based approach with the help of *mlclust* package. We can take a look in our results, and check what how out initiative did.



In order to take as much as possible from the plot provided above, we need to give to the reader :

EII spherical, equal volume

VII spherical, unequal volume

EEI diagonal, equal volume and shape

VEI diagonal, varying volume, equal shape

EVI diagonal, equal volume, varying shape

VVI diagonal, varying volume and shape

EEE ellipsoidal, equal volume, shape, and orientation

EVE ellipsoidal, equal volume and orientation (*)

VEE ellipsoidal, equal shape and orientation (*)

VVE ellipsoidal, equal orientation (*)

EEV ellipsoidal, equal volume and equal shape

VEV ellipsoidal, equal shape

EVV ellipsoidal, equal volume (*)

VVV ellipsoidal, varying volume, shape, and orientation

As for our case we can see that the best models are provided for with 3 clusters which at this point we know that is not the best case. The models with the best BIC score are **VEI** and **EEI** models. On the other hand, even though the best models with two clusters are the same ones, are much better than all the other cases.

Based on the results and methods used above we can see that hierarchical clustering return to us results similar to ground truth. In order to get a further confirmation we used Adjusted Rand Index to compare and decide on the best model given the data provided. Of course there are many more methods one can choose to work with in order to get similar and even more confident results.

3 Exercise 3

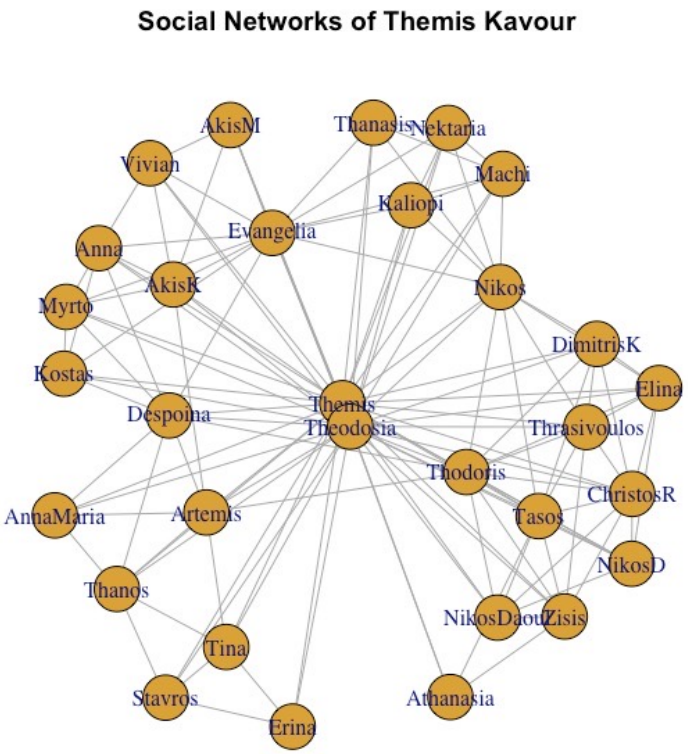
(Social Network). Consider a sample(minimum: 20 persons) from your personal social environment (e.g. colleagues, family friends, college friends, etc. . .). Construct the corresponding friendship network (that is: the adjacency matrix table the value 1 whenever two persons are frienδs and 0 otherwise). Visualize, analyze and cluster the data using appropriate method(s) and discuss the results.

Solution:

In order to complete this exercise we are going to use the following packages:

- igraph
- network
- intergraph
- ergm
- latentnet

As I have lived in multiple cities and developed social networks there, I have decided to include those, according to the geographical place established. The places that I am going to include are Alexandroupolis, Thessaloniki, Kozani and Athens. Initially, we need to define the relations within each network. We need to keep in mind that by the method used initially, the relationships are read by **R** in pairs of twos as a result, we need to specify every relationship that exists otherwise we will end up with 4 straight lines that extend from me, as those are 4 social networks that have me in common. So by working careful, I ended up in the following network:



Before moving forward to the analytical part of the newtwork, I shall add a new item in the social network list and name it gender. We can add any element in a new item in the network's list like where the initial introduction took place for example, but for now let us keep it simple and add just the gender of everyone.

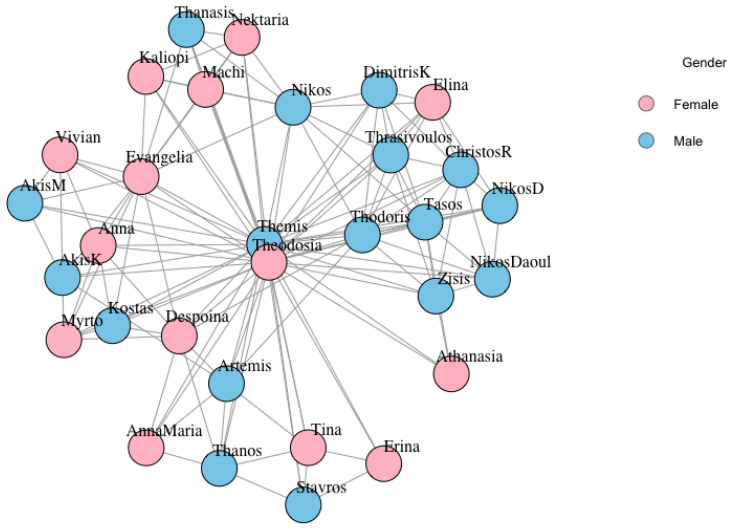


Figure 6: Social Network of Themis Kavour coloured according to the given gender.

As we explore the social network built earlier we can see that (*vcount*) we have 31 vertices in it that are connected with the help of (*ecount*) 141 edges. This number of edges is 30.3% of all possible edges one could find in this particular social network. Apart from those one can distinguish (*transitivity*) that there are generated 40% of the possible triangles. Lastly we can see that the longest route one can take is by following at most 2 edges. As generated in with the help of R:

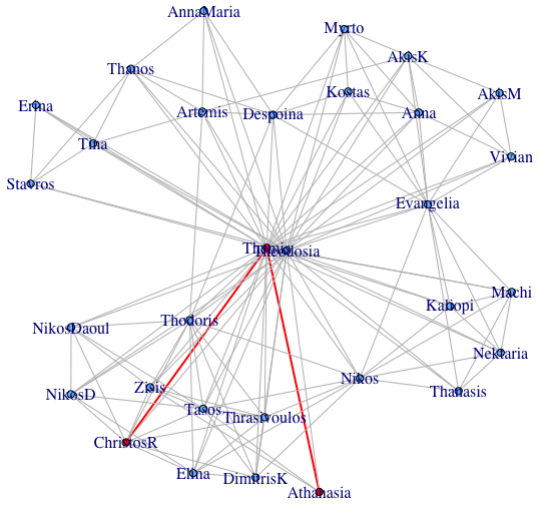


Figure 7: Longest route generated with the help of R.

Let us now see the most popular vertices (a.k.a. the ones that are connected the most.

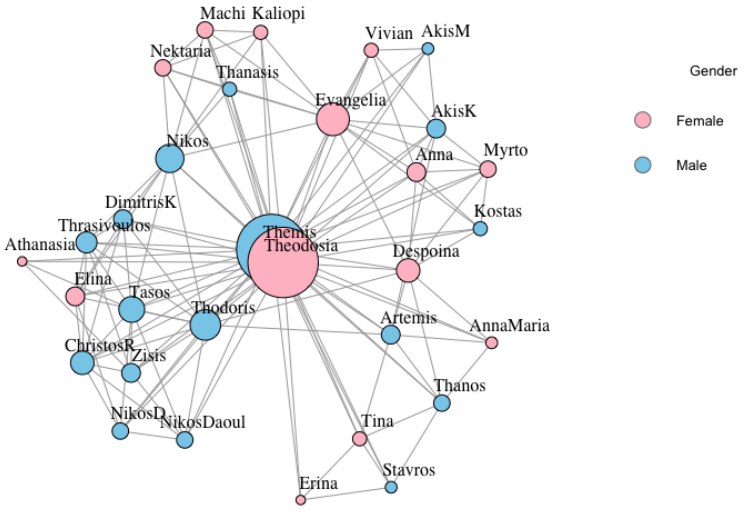


Figure 8: Social Network of Themis Kavour coloured by gender, sized by popularity.

Now that we have fully understood our social network, with the assistance of *igraph* package, we can try and detect communities. This can be easily done

in r by using *cluster_fast_greedy* function. This is a hierarchical algorithm for finding community structures. The result is the following graph which shows all the communities detected by the method.

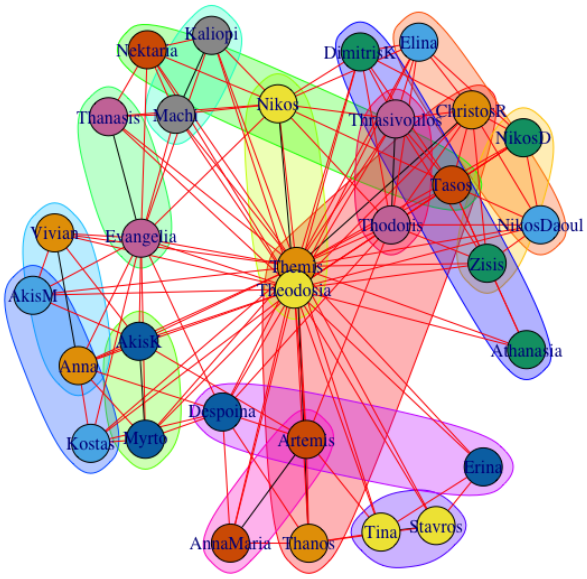


Figure 9: Communities Detected

Now it is possible that this makes no sense to you but, I can assure you that though there are some errors of the communities built given the ground truth, most of the situations found here do exist in real life as well. For example, Artemis and AnnaMaria are a couple, Vivian and Anna are really good friends as well as Thodoris and Trhasivoulos, Thanasis and Evangelia are siblings etc.

Moving forward we can produce a histogram to see what is the distribution of the vetrtices' connections:

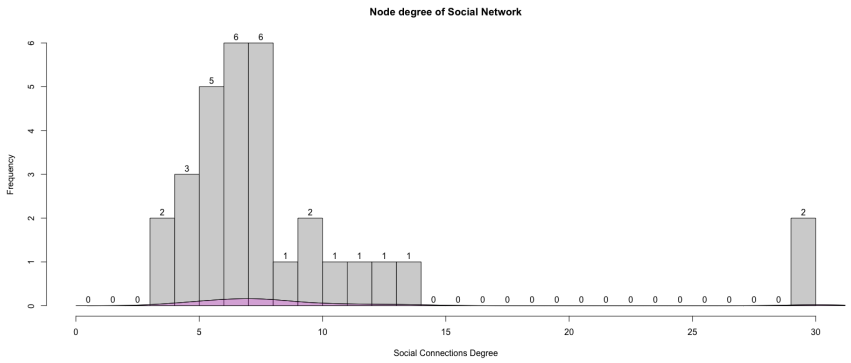


Figure 10: Number of connections

We can see here that there are two cases with the most connections. This is apparently true as my girlfriend do know everyone. So this is the case where, we are the ones with the most connections in my social network.

Now we are going to recalculate the community networks using some other. In order to do that, we first need to convert the (*igraph* object that we have been using all along, to a Network object. This can be done with the help of the function *asNetwork*. After that is completed, we are going to fit a new model, by using **Monte Carlo maximum Likelihood estunation** (**MCMLE**). This step will be implemented with the help of *ergm* package. After this step is completed, we can see that :

```
Call:
ergm(formula = net.m1 ~ edges)

Maximum Likelihood Results:

Estimate Std. Error MCMC % z value Pr(>|z|)
edges   -0.8320     0.1009      0  -8.247   <1e-04 ***
---
Signif.codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Null Deviance: 644.6  on 465  degrees of freedom
Residual Deviance: 570.6  on 464  degrees of freedom

AIC: 572.6  BIC: 576.8  (Smaller is better. MC Std. Err. = 0)
```

We can see (as expected) that the model in order to generate results, used edges' information and it is statistical significant variable. By the **Null Deviance** we get how well the model would predict the response variable by a model with nothing but the intercept. On the other hand, by **Residual Deviance** we can see how well the model predict the response variable with the a full model. In our case this is one variable difference (edges) which can also been seen by the degrees of freedom difference. Finally it is worth noting that edges has negative impact (*-0.8320* We can see (as expected) that the model in

order to generate results, used edges' information and it is statistical significant variable. By the **Null Deviance** we get how well the model would predict the response variable by a model with nothing but the intercept. On the other hand, by **Residual Deviance** we can see how well the model predict the response variable with the a full model. In our case this is one variable difference (edges) which can also been seen by the degrees of freedom difference. Finally it is worth noting that the information about AIC and BIC are presented here. Since AIC and BIC are mentioned, which are use to compare models and decide on which is better. We have decided to move forward to Goodness-of-fit check for the model already built.

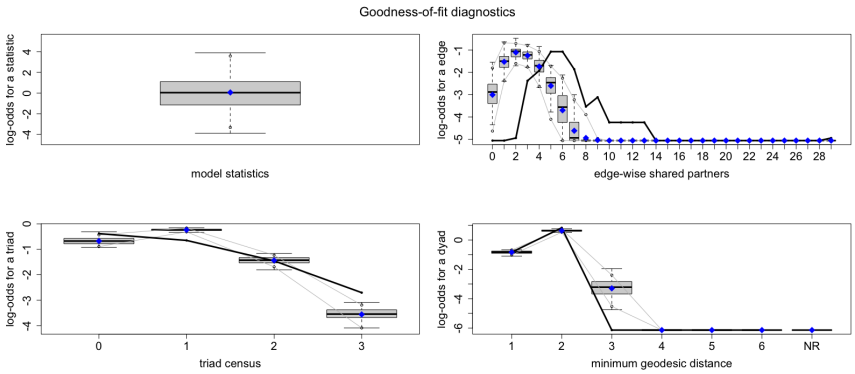


Figure 11: GOF Check

We can clearly see that the the fit is not good in our case. This can be easily be understood if we compare the black line to the given box-plot's limits. Lastly, we are going to take advantage of the package *latentnet* and build some final models. Here the algorithm works as follows. Each node in a d-Dimensional Euclidean space (in our case $d = 2$) is give a probability the belongs to a cluster. We are going to build models with $G = 2, \ldots, 5$ clusters. We have gone through this exercise with the city of initial introduction, and the number of cities is 4. So let us see if we have this as the best answer to our model buildings.

- Initially, we have a model for $d = 2$ and $G = 2$:

```
==========================
Summary of model fit
==========================

Formula:   net.m1 ~ euclidean(d = 2, G = 2)
Attribute: edges
Model:     Bernoulli
MCMC sample of size 4000, draws are 10 iterations apart, after
                                  burnin of 10000 iterations.
Covariate coefficients posterior means:
            Estimate    2.5%   97.5% 2*min(Pr(>0),Pr(<0))
(Intercept)   3.2271 2.4094 4.1015            < 2.2e-16 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Overall BIC:         588.0609
Likelihood BIC:      329.2961
Latent space/clustering BIC:      258.7648

Covariate coefficients MKL:
             Estimate
(Intercept) 2.001644
```

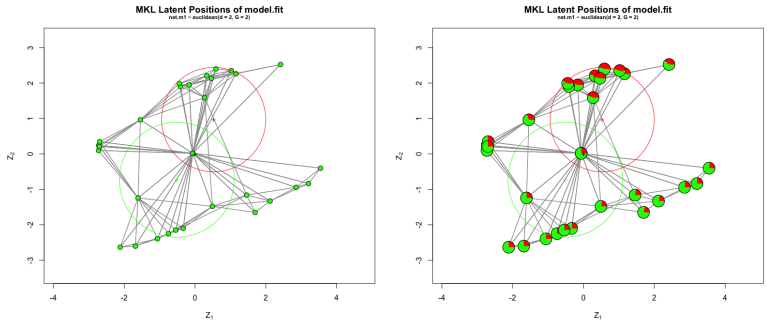


Figure 12: Cluster with $d = 2$ and $G = 2$

- The next model in line is the one for $d = 2$ and $G = 3$:

```
==========================
Summary of model fit
==========================

Formula:   net.m1 ~ euclidean(d = 2, G = 3)
Attribute: edges
Model:     Bernoulli
MCMC sample of size 4000, draws are 10 iterations apart, after
                                  burnin of 10000 iterations.
Covariate coefficients posterior means:
             Estimate   2.5%  97.5% 2*min(Pr(>0),Pr(<0))
(Intercept)   3.1772 2.4075 4.0085             < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Overall BIC:         578.3392
Likelihood BIC:      330.686
Latent space/clustering BIC:      247.6532

Covariate coefficients MKL:
             Estimate
(Intercept) 2.005083
```

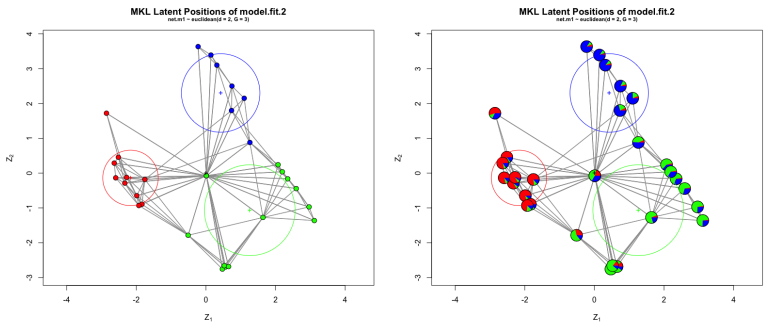


Figure 13: Cluster with $d = 2$ and $G = 3$

- Next model with $d = 2$ and $G = 4$:

```
============================
Summary of model fit
============================

Formula:   net.m1 ~ euclidean(d = 2, G = 4)
Attribute: edges
Model:     Bernoulli
MCMC sample of size 4000, draws are 10 iterations apart, after
                                    burnin of 10000 iterations.
Covariate coefficients posterior means:
            Estimate   2.5%  97.5% 2*min(Pr(>0),Pr(<0))
(Intercept)   3.1843 2.4135 4.0073             < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Overall BIC:         551.7635
Likelihood BIC:      330.3137
Latent space/clustering BIC:     221.4498

Covariate coefficients MKL:
            Estimate
(Intercept) 2.046161
```

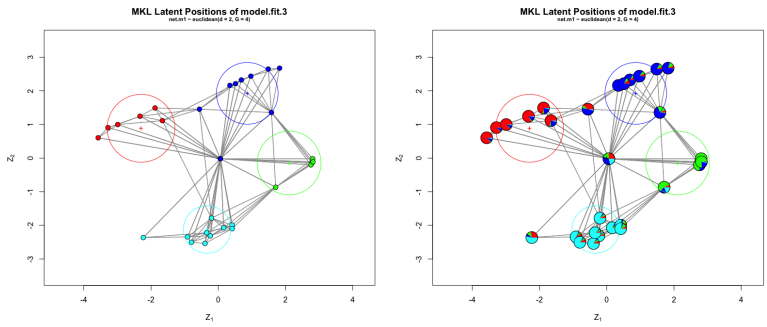


Figure 14: Cluster with $d = 2$ and $G = 4$

- Finally a model with $d = 2$ and $G = 5$:

```
==========================
Summary of model fit
==========================

Formula:   net.m1 ~ euclidean(d = 2, G = 5)
Attribute: edges
Model:     Bernoulli
MCMC sample of size 4000, draws are 10 iterations apart, after
                                    burnin of 10000 iterations.
Covariate coefficients posterior means:
            Estimate   2.5%  97.5% 2*min(Pr(>0),Pr(<0))
(Intercept)   3.1889 2.4274 3.9968             < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Overall BIC:         578.295
Likelihood BIC:      330.0624
Latent space/clustering BIC:     248.2327

Covariate coefficients MKL:
            Estimate
(Intercept) 2.070653
```

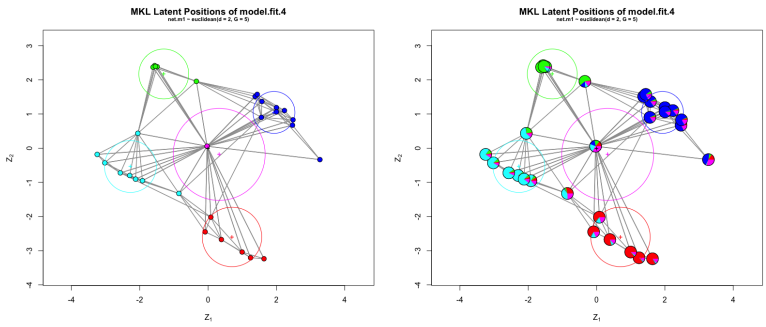


Figure 15: Cluster with $d = 2$ and $G = 5$

Below we have collected BIC values, Likelihood BIC values (**L-BIC**) as well as $\frac{\text{Latent space}}{\text{Clustering BIC}}$ values (**LSpace/ClBIC**) in order to make the comparison easier for the reader:

Models Comparison			
Model	BIC	L-BIC	LSpace/ClBIC
G = 2	588.06	329.3	258.76
G = 3	578.34	330.69	247.65
G = 4	551.76	330.31	221.45
G = 5	578.3	330.06	248.23

We can see that this method indicated that 4 clusters is the optimal number of them. One may continue and divide them into even more number of clusters, though since we know the what the optimal number should be investigating the limits of this method is out of this exercise's scope.