Assignment3

Faizan Khalid Mohsin

Contents

1 Question 1

2 Introduction

In the world of Precision Medication, Genetics research and Big data in general, methods for clustering over large number of variables, sometimes over 10,000 features in genetics data, has become very important. With data becoming ever bigger, and being collected without any specific use in mind at the time of collection, unsupervised clustering methods can be very useful when trying to analyze the data for future purposes. Also, in many cases the outcome variable of interest is not know, or collected, hence again, unsupervised clustering methods can be very useful.

There have been several unsupervised clustering methods developed over the past years, however, we will be mainly using the traditional k-means clustering and hierarchical clustering methods. In this study we will be exploring genetic data of mice with over 77 variables using these two methods.

3 Methods

3.1 Data Set

We will be using some data of expression levels of proteins/protein modifications that produced detectable signals in the nuclear fraction of cortex. In the original data there are 77 proteins/protein modifications. There are 38 control mice and 34 trisomic mice (Down syndrome), for a total of 72 mice. In the experiments, 15 measurements were registered of each protein per sample/mouse. Therefore, for control mice, there are 38x15, or 570 measurements, and for trisomic mice, there are 34x15, or 510 measurements. The dataset contains a total of 1080 measurements per protein.

For our purposes, we will treat each measurement as an independent sample/mouse.

The eight classes of mice are described based on characteristics such as genotype, behavior and treatment. According to genotype, mice can be control or trisomic.

Feature information in columns

- 1 Mouse ID
- 2-78 Values of expression levels of 77 proteins (their names are given as names of the columns)
- 79 Genotype: control (c) or trisomy (t)
- 80 Treatment type: memantine (m) or saline (s)
- 81 Behavior: context-shock (CS) or shock-context (SC)
- 82 Class: c-CS-s, c-CS-m, c-SC-s, c-SC-m, t-CS-s, t-CS-m, t-SC-s, t-SC-m

Features 79 to 82 are all categorical variables and we will be collectively referring to them as the "characteristics" the their factors as "characteristics levels".

3.2 Missing Data

Now, for dealing with missing data, multiple imputation by chain equations with predictive mean matching is a very good and reasonable method for handling missing data. The fact that it is a semi-parametric method and provides some degree of robustness against the violations of the imputation model assumptions such as normality, homoscedasticity of residuals, and linearity (Morris, White, and Royston 2014), are advantageous. However, random forest imputation (RFImp) is a completely non-parametric method and can handle nonlinearities (Shah et al. 2014).

Therefore, we will be using random forest imputation (RFI) for the missing data. RFImp will be implemented using the "missForest" package in R. Further, for the RFImp the mouse ID variable was removed since it had

1080 factors and is not informative because we are treating each data entry (row) as independent. Moreover, mouse ID has 1080 factors and missForest cannot handle variables with more than 53 factors.

3.3 Data exploration

We perform some data exploration to get an idea of how our data looks like. We present the correlation matrix of the numerical variables as a heat map, with dark red representing a strong negative correlation, blue a strong positive correlation and white a correlation of zero. We also present pairwise scatter plots of some of the numerical variables that exhibit an interesting pattern. Finally, we present a heatmap of the data which allows one to visualize the entire dataset at a glance. The heatmap also shows the feature clustering as well as the observation clustering via a dendrogram.

3.4 Silhouette: Comparing Clustering Methods (K-Means, Hierarchical and PAM) and Number of Clusters.

We use the silhouette method to determine which of the three methods perform better clustering and what the optimal number of clusters is. We describe the silhouette method below.

Once clustering is performed, the silhouette method produces a silhouette width for every point. The values of the silhouette width range from +1 to -1, where values close to +1 indicate that the point has been clustered correctly. Values close to -1 indicate that it has been clustered incorrectly. Average silhouette width can be used for selecting the optimal number of clusters as well as comparing between clustering methods.

The closer the average silhouette width is to +1 for a clustering method, for the indicated number of clusters, the better the performance.

We calculated the average silhouette width for k-means, hierarchical and PAM clustering methods, for the number of clusters from 2 to 5.

We used the highest the average silhouette width value for determining the best clustering method and the optimal number of clusters.

3.5 Reduced Dimension PCA Plots

Visualizing a data set with over 77 variables is a challenging task.

Now once we performed the clustering, to visually see if the clusters are associated with any of the different genotype, behaviour and/or treatment, we need to do a dimension reduction. Ideally, the dimension reduction would preserve some structure of the data and reduce the dimension to two. This would allow us to easily plot and see the association between the clusters and the categorical variables, when we include those in the reduced dimension plots.

Hence, to do this, after the clustering is done, we perform PCA on the scaled data and then using the first two principal components to plot a graph with the data points. One these data points we also indicate the different levels the points belong for a categorical variable through different colours. Further, the points are plotted using different shapes (triangles, circles, etc.) denoting which clusters the they belong to. To create these visualizations we create a helper function which is mainly built on the clusplot() function in R. This helper function is defined in Appendix B.2.

This is a good method firstly, because it is a visual method, instead of presenting tables (concordance tables, which we also do for completeness), we can immediately see the any general patterns and which levels of a categorical variable are associated with which clusters.

Lastly, we call these reduced PCA (rdPCA) plots.

3.6 K-Means Clustering

We performed k mean clustering to find distinct clusters in the data. We first scale the data because we do not want the variable scales to effect the clustering. For example, we do not want the fact that we might have the height in meters versus centimeters to affect the the clustering. Therefore, we scale the data before performing k-means clustering. Further, we only include numerical variables in the k-means clustering because its methodology depends on calculating the distance of the data points. A concept which is ill defined for categorical variables using the traditional distance metrics.

We do exploratory k-means clustering for k from 2 to 5, using primarily visual techniques. We also present some concordance tables to see if any clusters are associated with any of the different characteristics (categorical variables). For k = 2, we do formal analysis of the clusters and also describe the two clusters using means of the proteins, and counts of the levels (see Table 2 in the K-means clustering Results section).

For performing k-means clustering, the nstrat value (the number of random starts the k-means clustering will do) we chose the value of 40 because its total.see plot was stable for all the k-means clustering we preformed.

Computationally, when we tried to create 8 clusters using k-means clustering, we found that, even nstart values of up to 500 did not produce stable plots for tot.sse. Further, we when tried running this for nstart 600, throughout the run, the message "did not converge in 10 iterations" was continuously being produced and took a very long time to run in R and ultimately crashed. Hence, running k-means clustering for higher values of k was computationally intensive and challenging.

Therefore, we limited performing k-means clustering to small values of k, and in particular did not perform k-means clustering for k = 8 because we did not find an appropriate nstart value for the tot.sse plot to be stable. Even though the Class variable had 8 levels.

3.7 Hierarchical Clustering

For the hierarchical clustering we first choose only the numerical variables and then scale the data, similarly as in k-means clustering due to the same reasons. Lastly, we create the distance matrix as the hierarchical clustering model only needs the distances and not the actual data for clustering.

We perform hierarchical clustering using the three linkages: complete, average and single, and use the linkage which creates the most balanced clusters. We find this to be complete linkage as it appears to do the most balanced clustering and it also looks the most reasonable.

Computationally, running hierarchical clustering was much easier compared to k-means clustering. Once, the hierarchical clustering model finished running (which was very quick), it was very easy to cluster the data, even for very high values of k, unlike for the k-means clustering.

Lastly, even if the average silhouette width for hierarchical clustering is less than that of PAM. We still use this method for doing exploratory clustering because of the great potential and easy of understanding and implementing this method. Further, all the main conclusions will be drawn from k-means clustering since it had the highest average silhouette width.

3.8 Comparing Means between two Clusters for Different Proteins.

Once we have clustered the data into two groups using k-means clustering, we will test which proteins have a statistically significant difference in mean for two clusters. We will be using the t-test with non-equal variance assumption. Since there are 77 proteins in the data set we will conduct 77 t-tests. Due to multiple hypothesis testing, we will do a bonferroni correction to adjust the significance level when determining which proteins have a statistically significant difference in mean for the two clusters. We will be using a bonferroni corrected significance value of $\alpha = 0.05/77$.

4 Results

4.1 Results Summary

4.1.1 Data Exploration

From the correlation plot (Figure 1.) we see that a lot of the variables are strongly and positively correlated with one another. This could be because we have multiple measurements coming from the same mice making the data points correlated.

4.1.2 Silhouette: Comparing Clustering Methods and the Optimal Number of Clusters

We found that for both, k-means clustering and hierarchical clustering, the optimal number of clusters was k = 2.

Further, we found that for k = 2, k-means clustering was a better clustering method compared to hierarchical clustering as it had a higher average silhouette width. The average silhouette width for k-means, PAM and hierarchical clustering were 0.144, 0.139, and 0.121, respectively. Relatively speaking, these values are low, and ideally we would like to have the average silhouette width of our clustering method to be closer to 1.

Now, even though PAM cluster better than hierarchical, the second method we will use for clustering, after k-means clustering, will be hierarchical clustering. We do this because hierarchical is a very different clustering method to both K-means and PAM. Further, it is very easy to implement and cluster for large values of k, and what it is doing for clustering, is captured in dendrograms, and can be understood easily.

4.1.3 K-Mean Clustering

Using the rdPCA plots we found the levels of "Behavior", C/S and S/C, to be associated with with the different clusters we created (see Figures 6, 7 and 8).

We found 58 proteins to have a statistically significant difference in mean between the two clusters after the boferroni correction. The list of these proteins are (see Table 1 to see their p-values in the K-Means Clustering section in Results).

Since, k = 2 is the optimal number of clusters we also described the two clusters using mean protein levels and counts for the categorical variables in Table 2 in the K-Means Clustering section in Results.

Looking at Table 2 we found that the class t-SC-m is mainly found in cluster 2 (113 out of 135 (83%) t-SC-m's were in cluster 2). Further, from the same table we can see that more S/C Behavior points are found in cluster 2 (366 out of 555 (70%) S/C's were in cluster 2).

For k = 3, we found some interesting clustering for the proteins DYRK1A_N and BDNF_N. In Figure 14 cluster 2 is on the lower right bottom of the graph. Cluster 1 is on the upper left side of the graph. The two clusters make a wedge and the two flank cluster 3, which is found in between the two. It can also be seen that almost all S/C points are below the value 0.5 for the DYRK1A_N protein (x-axis).

4.1.4 Hierarchical Clustering

From looking at the plots of the hierarchical clustering of the complete, average and single linkages, complete linkage produced the most balanced clusters and appeared the most reasonable. Hence, for performing hierarchical clustering, we used complete linkage.

For k = 2, it can be seen in the rdPCA plots, that the clustering is different compared to what we had for k-means clustering. For instance, looking at Figure 19 one can see that hierarchical clustering produces one is very large cluster and another much smaller cluster. Whereas, k-means had two clusters that were relatively of similar sizes.

We further went and calculated how many points were clustered similarly. From Table 3 it can be seen that about 24.1% (260 out of 1080 points) were clustered differently from k-means clustering. Hence, approximately 76% of the data points were clustered the same way by both clustering methods. See Table 3 on page 28, and Figure 6 and Figure 19 on pages 14 and 27, respectively (note that in the figures, k-clustering cluster 1 corresponds to cluster 2 of the hierarchical clustering - the labels were switched).

For k = 3, we notice that cluster three has very few points for all four characteristics. This pattern is present as we increase k. Most of the points are in few of the clusters and the remaining clusters have very points in general.

For k = 8, in the concordance table we do not see any distinct clusters representing any of the 8 levels of the class characteristic distinctly.

4.1.5 Characteristics

We found that for the proteins ITSN1_N and BDNF_N the means for two levels of the behavior characteristic C/S and S/C were statistically significantly different (see Figures 12 and 13).

4.1.6 Principal Component Analysis: Proteins that exhibit a distinct expression pattern (profile)

We see that the first 9 pc's explain 80.6% of the total variance.

Further, there is a distinct pattern in the biplot. All the proteins are pointing left of the y-axis.

any particular set of proteins that exhibit a distinct expression pattern (profile) in any or all of these clusters

In terms of any particular set of proteins that exhibit a distinct expression pattern (profile), we can see from Figure 24, almost all the proteins can be grouped into two clusters: either they are positively correlated to the second principal component or negatively correlated to the second principal.

4.2 Data exploration

```
##     Genotype       Treatment   Behavior      class
##  Control:570   Memantine:570   C/S:525   c-CS-m :150
##  Ts65Dn :510   Saline   :510   S/C:555   c-SC-m :150
##                                          c-CS-s :135
##                                          c-SC-s :135
##                                          t-CS-m :135
##                                          t-SC-m :135
##                                          (Other):240
```

From the summary we can see that all the categories are well balanced. This is expected given the information provided in section 3.1. of the methods section.

Below we graph the correlation plot to get a sense of how correlated our data is.

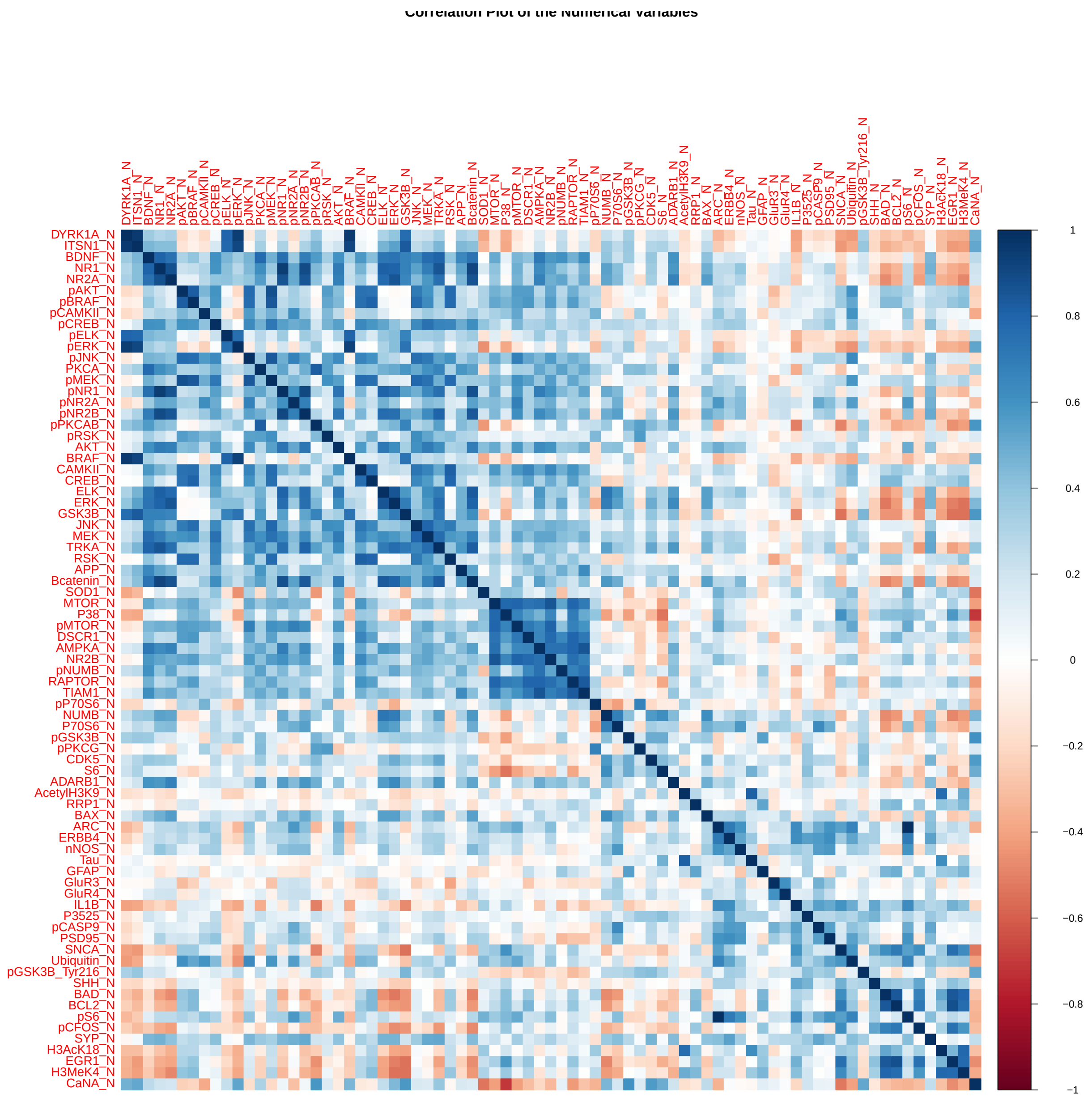
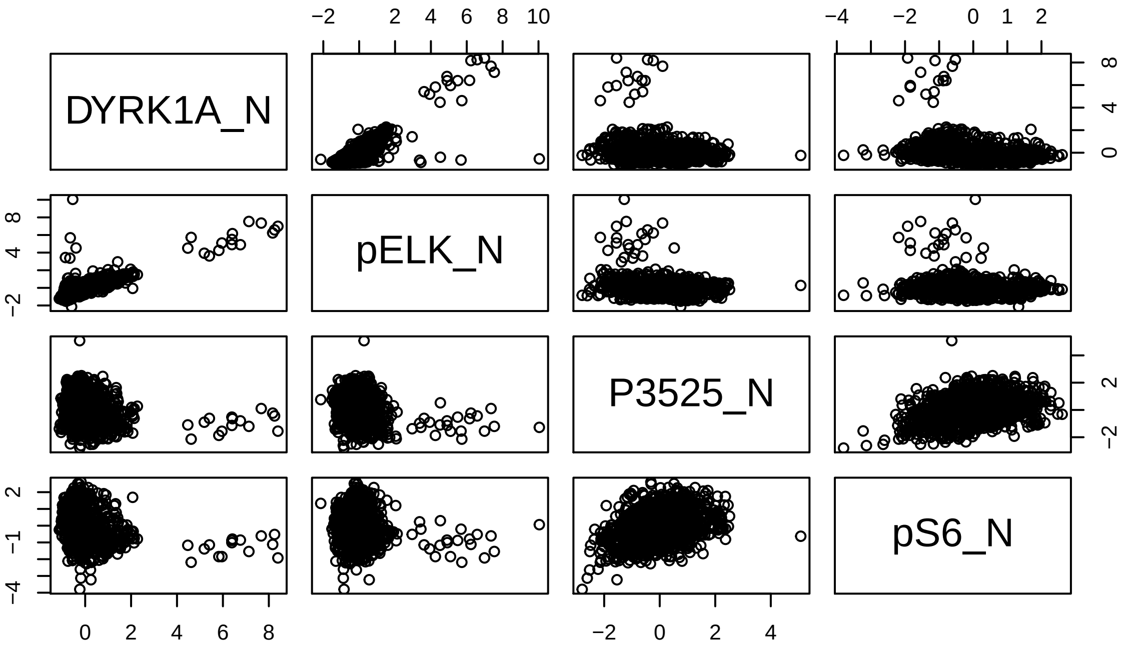
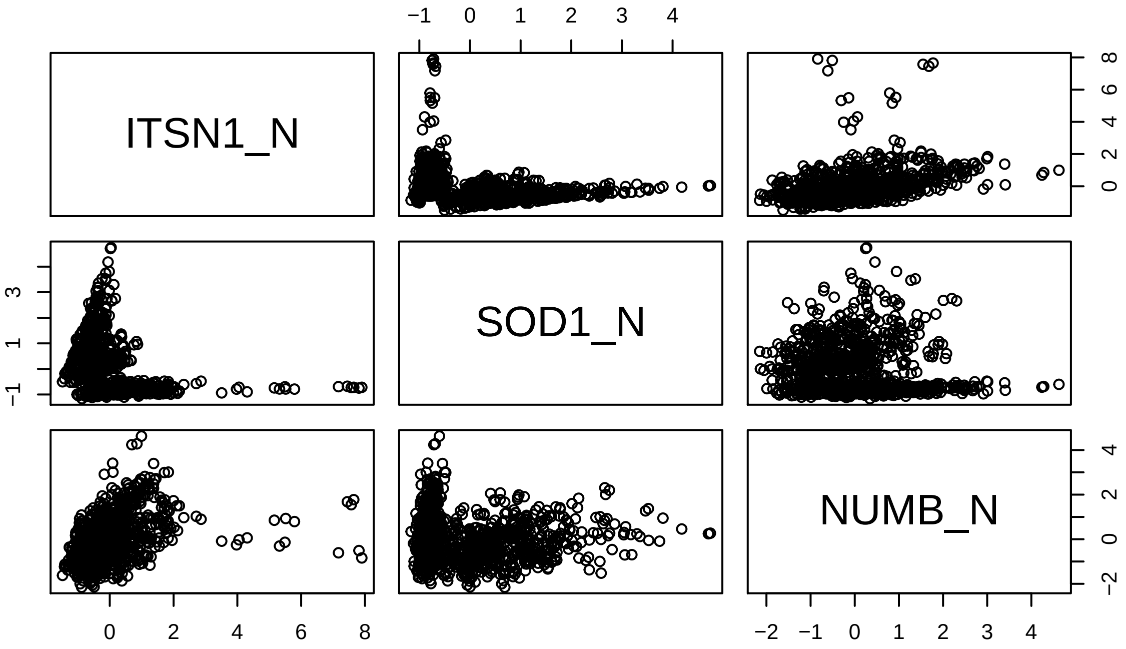


Figure 1.

From this graph we see that most of our variables are positively correlated with one another and that several of them are very strongly correlated. This most likely is because the data points are correlated as one mouse has several measurements taken from him.

Below we will produce the pairwise scatter plots of some of the proteins that appeared to have an interesting pattern.





In these scatter plots variables appeared to have interesting patterns such as some having wedge like shapes, some being strongly linear and for some having two distinct separate groups of clusters.

4.3 Heatmap

We first look at the heatmap of our data before doing any clustering to get a general idea of our data. We create a heatmap using our numeric data below.

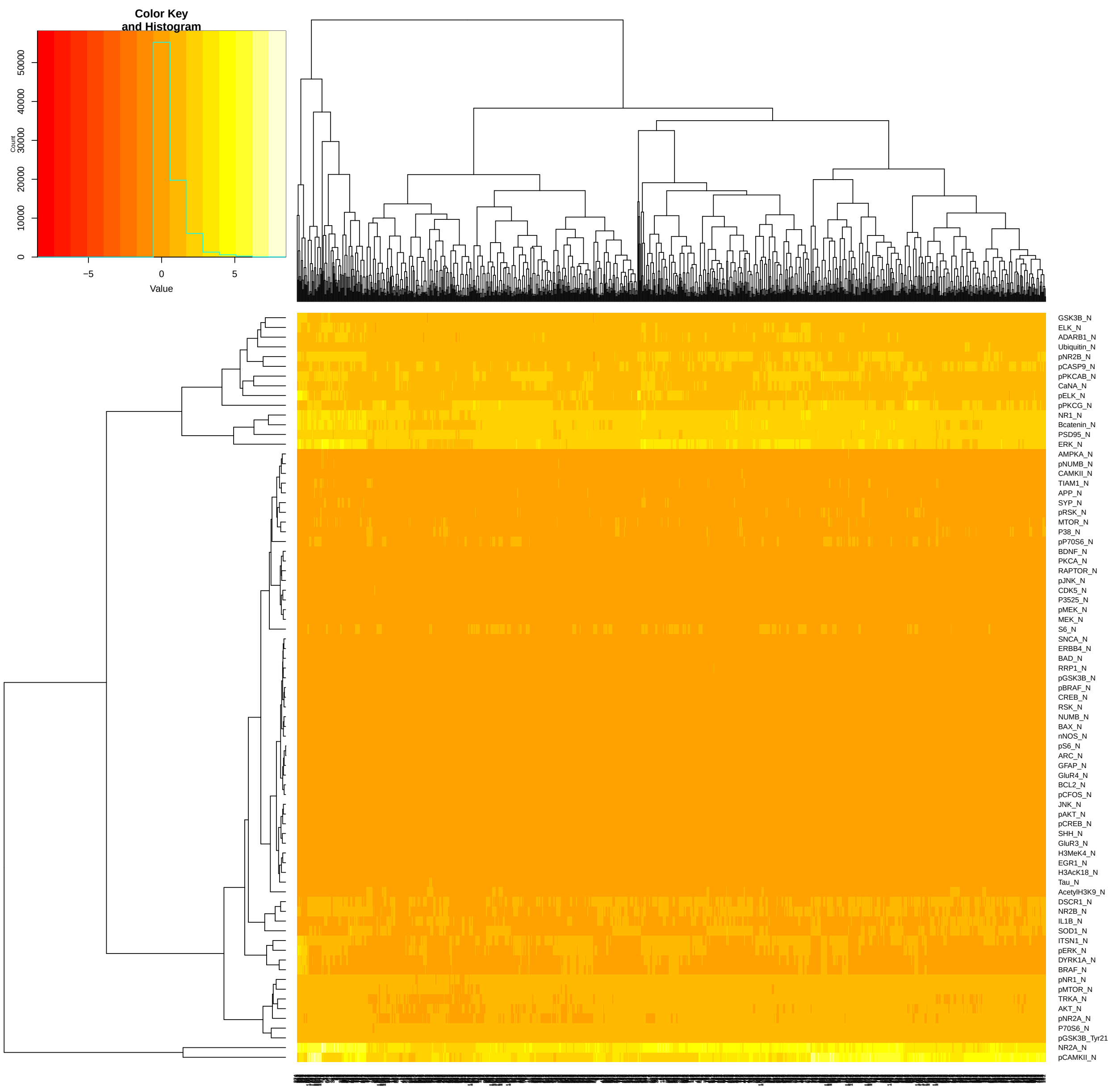


Figure 2.

From the heatmap above, by looking at the dendrogram on the left hand side that clusters the features, we see that our data has about two to three main clusters it naturally can be grouped into.

4.4 Silhouette: Comparing Clustering Methods and Number of Clusters

We first calculate the average silhouette width values for kmeans clustering for k from 2 to 5. The average silhouette width is plotted against the number of clusters in the graph below.

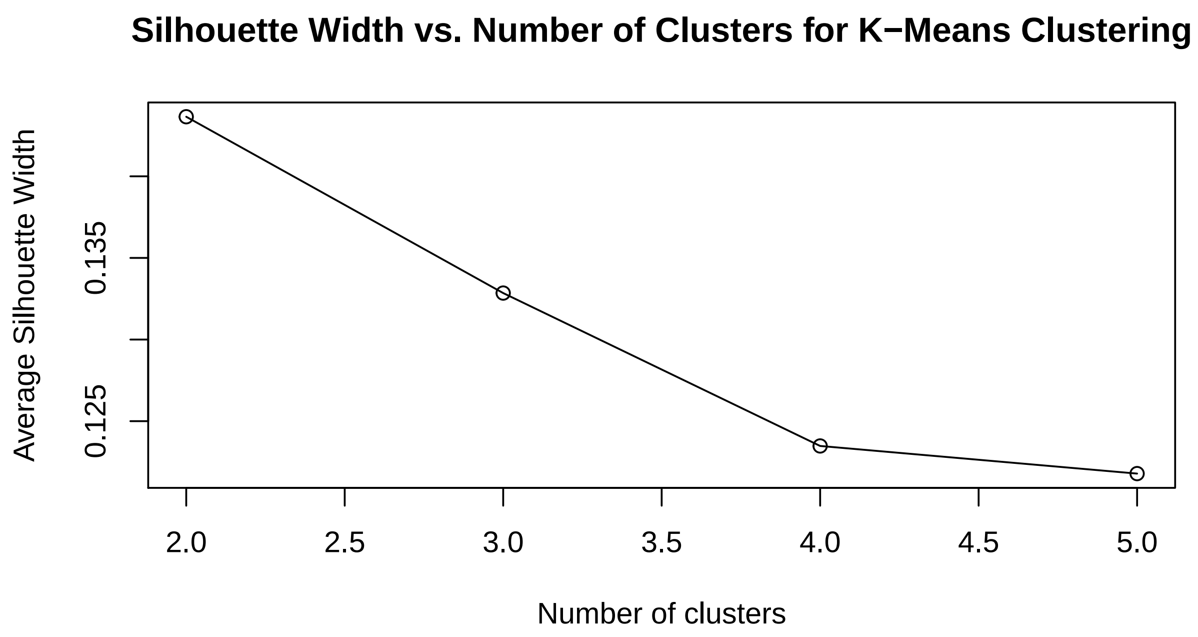


Figure 3.

From the plot above we can see that the k-mean clustering average silhouette width is highest for k = 2 clusters (0.1437) and it drops as k increases. Hence, this indicates that the optical number of clusters for k-mean clustering is 2.

We now plot the hierarchical clustering average silhouette width against the number of clusters in the graph below with k going from 2 to 5. For hierarchical clustering we use complete linkage.

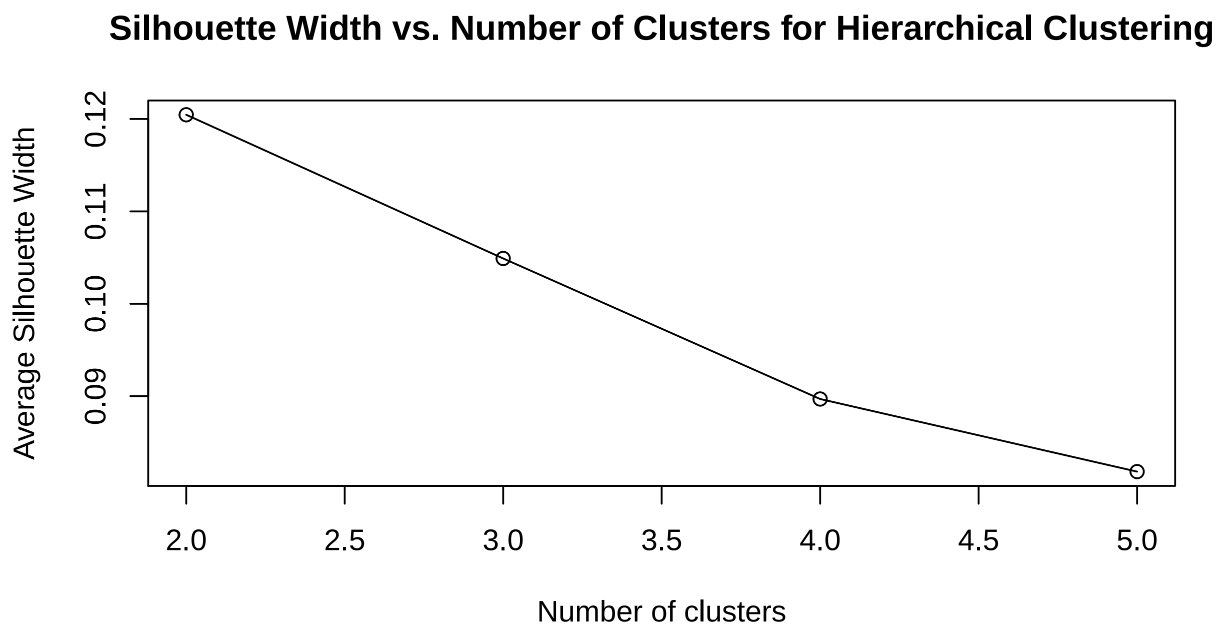


Figure 4.

From the above plot, we see that we have similar results as for k-means clustering. We can see that the hierarchical clustering average silhouette width is highest for k = 2 clusters (0.1206) and drops sequentially as k increases. Hence, the optical number of clusters for hierarchical clustering is 2.

We now plot the Partitioning Around Medoids clustering average silhouette width against the number of clusters in the graph below with k going from 2 to 5. For hierarchical clustering we use complete linkage.

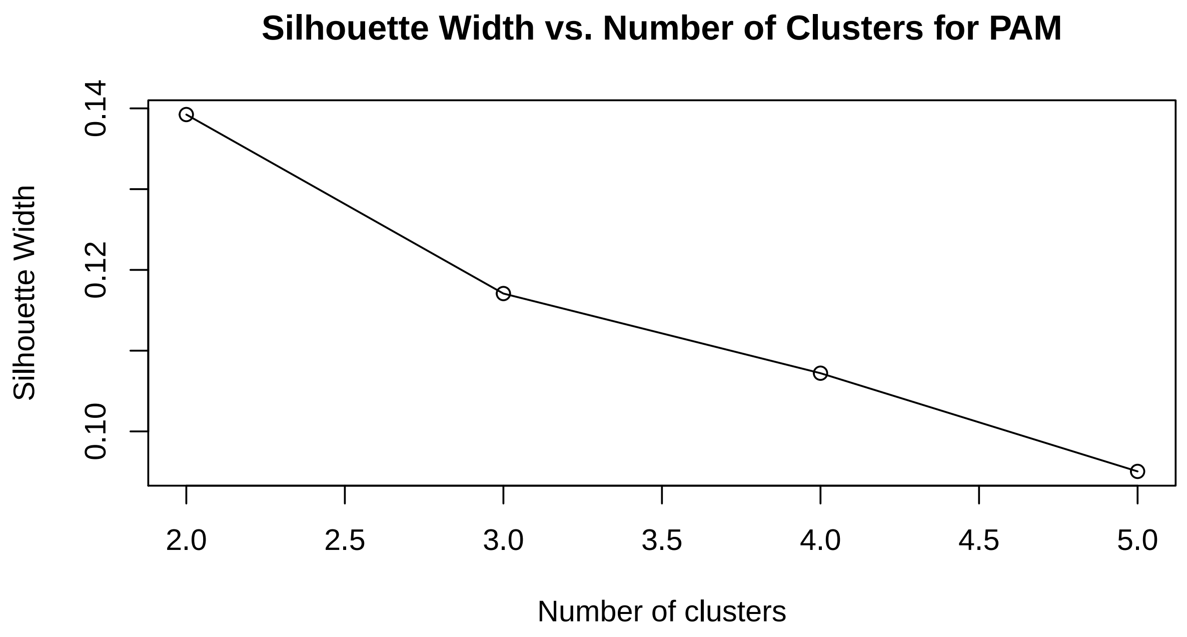


Figure 5.

Again, partitioning around medoids average silhouette width is highest for k = 2 clusters (0.1392) and drops as k increases.

Final Conclusion

Finally, we observe that for k = 2, k-means clustering has the higher average silhouette width (0.1437) compared to that of hierarchical clustering (0.1206) and PAM (0.1392). Hence, it appears k-means clustering is the best performing clustering method and PAM is the second best. Now, relatively speaking, the average silhouette width for all our clustering methods is low (≈ 0.14). We would like values as close to 1 as possible.

4.5 K-Means Clustering

Summary of what we will do for k-means clustering

As we have already determined that the optimal number of clusters is 2 we will perform a few analysis on the two clusters we create using k-means clustering (i.e. comparing means of the two clusters for different proteins). However, first we will do exploratory k-means clustering for different values of k from k = 2 to k = 8, and will plot the clusters using the PCA dimension reduction technique (explained in the methods section) plotted with different characteristics to see if any of the clusters represent or are associated with any of the classes or the levels of the characteristics.

For creating these reduced dimensions PCA (rdPCA) plots we create a helper functions called "plot_PCA_kmeans" that can be found in Appendix B.2 along with other helper functions created for plotting graphs.

Further, we graphed plots of the tot.sse for all the k values from 2 to 5 with nstart = 40, for the k-means clustering. All the plots were stable and can be found in Appendix A.1. The plots for k greater that 5 are unstable as they are not straight lines. .

Exploratory k-means clustering

We did k-means clustering for values of k from 2 to 5 and plotted the clusters using the rdPCA plots with different characteristics. We found the levels of the characteristic "Behavior" to have interesting overlays with the different clusters we created.

We also plotted the other characteristics using the rdPCA plots, however, we did not find any interesting clustering patterns or associations between the clusters and the different characteristic levels. These plots can be found in the Appendix A.2 section.

We present the behaviour rdPCA plots below.

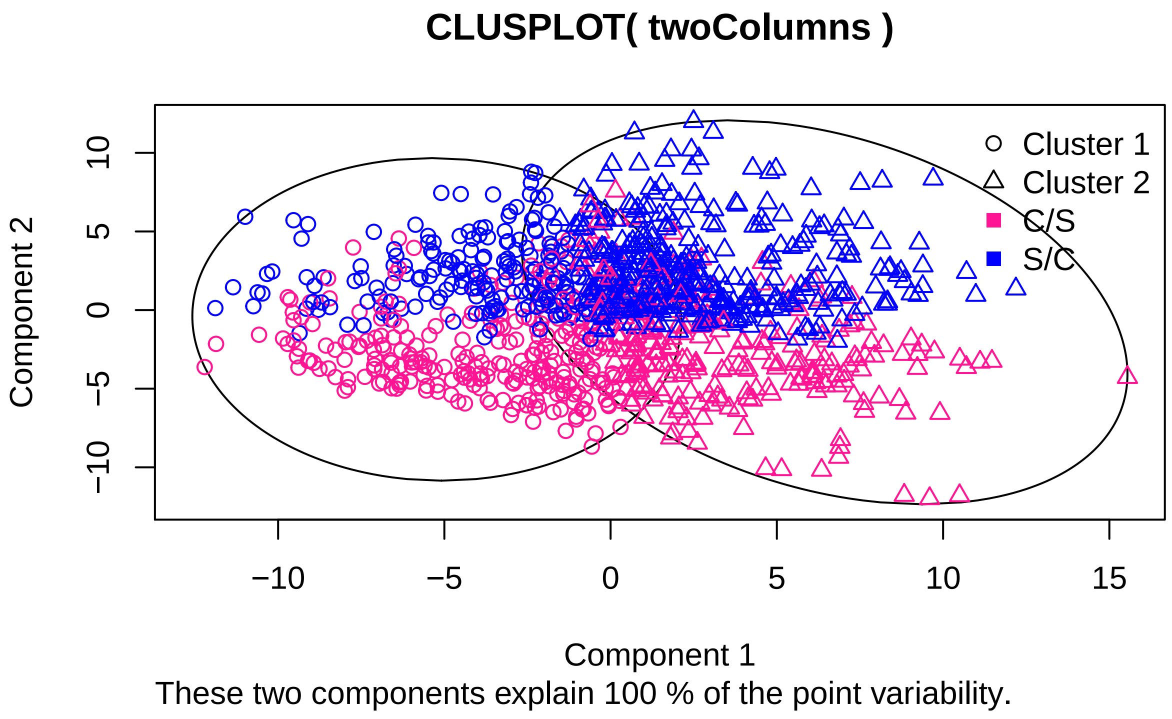


Figure 6.

This has interesting pattern. For the characteristic: behavior levels, the data seems to be well divided into upper and lower halves at zero, with all the S/C being above zero and majority of C/S being below zero.

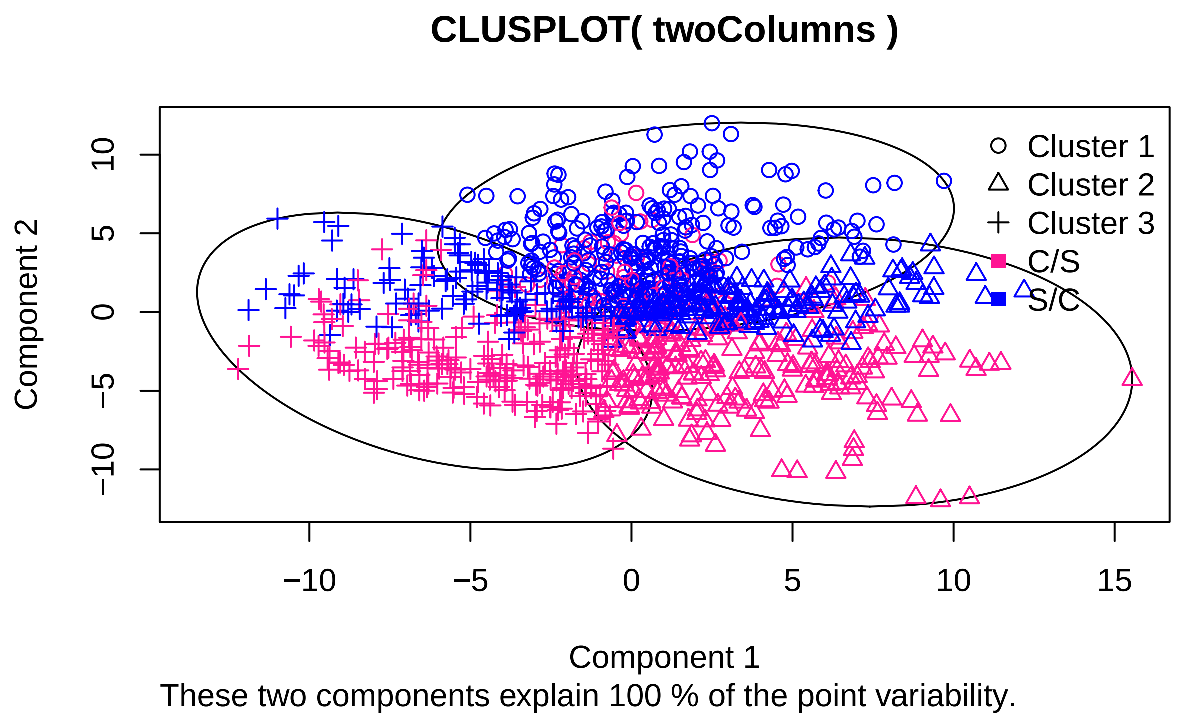


Figure 7.

From this plot it appears that cluster 1 (the circles), captures a lot of the S/C points. And that cluster 2 and 3 capture a lot of the C/S points. Further, all the points of cluster 1 (the circles) belong to S/C.

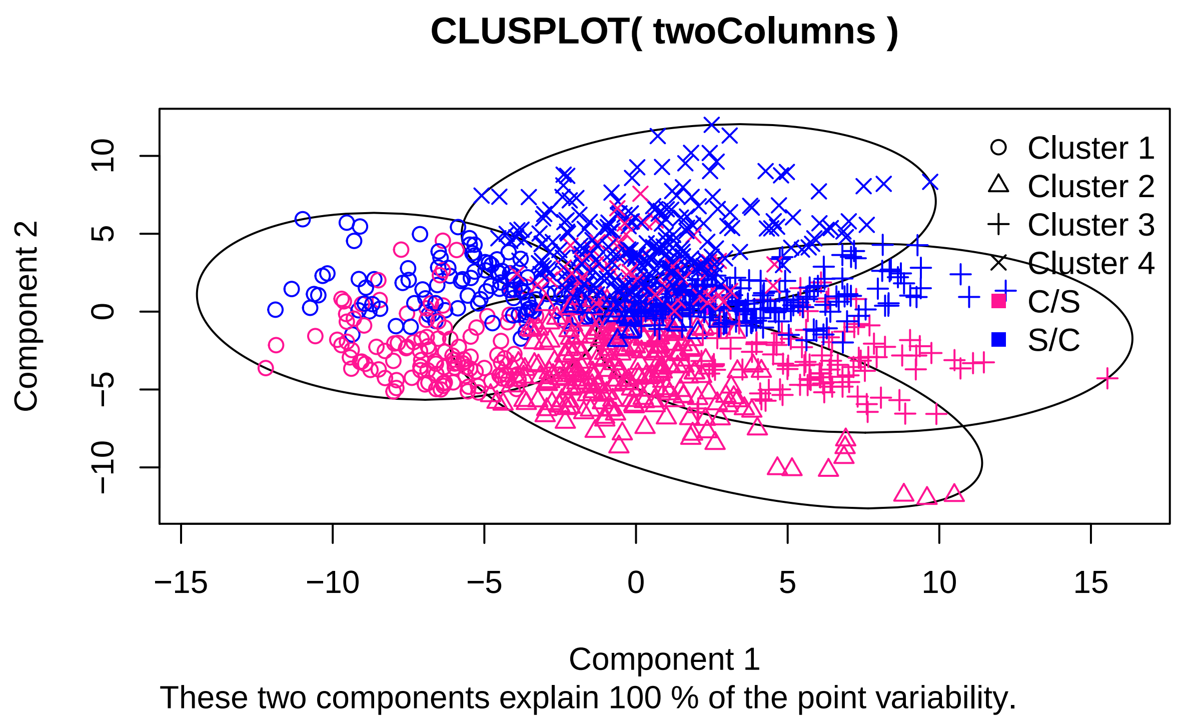


Figure 8.

We see an interesting clustering structure here. It appears that majority of the points in cluster 4 are C/S points (since there are few pink crosses), and the majority of points in cluster 2 are S/C points. Whereas, for clusters 1 and 3, they both have a well mixed number of C/S and S/C points. So clusters 2 and 4 appear to be homogeneous clusters with S/C and C/S, respectively and clusters 1 and 3 appear to be balanced heterogeneous clusters with both S/C and C/S points well represented in them.

For the other characteristics, they were equally distributed over the clusters for k equal to 2 and 5 and we could not really see any specific pattern or any of the clusters representing any levels of the other characteristics. We show a sample plot below. The other plots were very similar to this one. They can all be found in the section Appendix A.2.

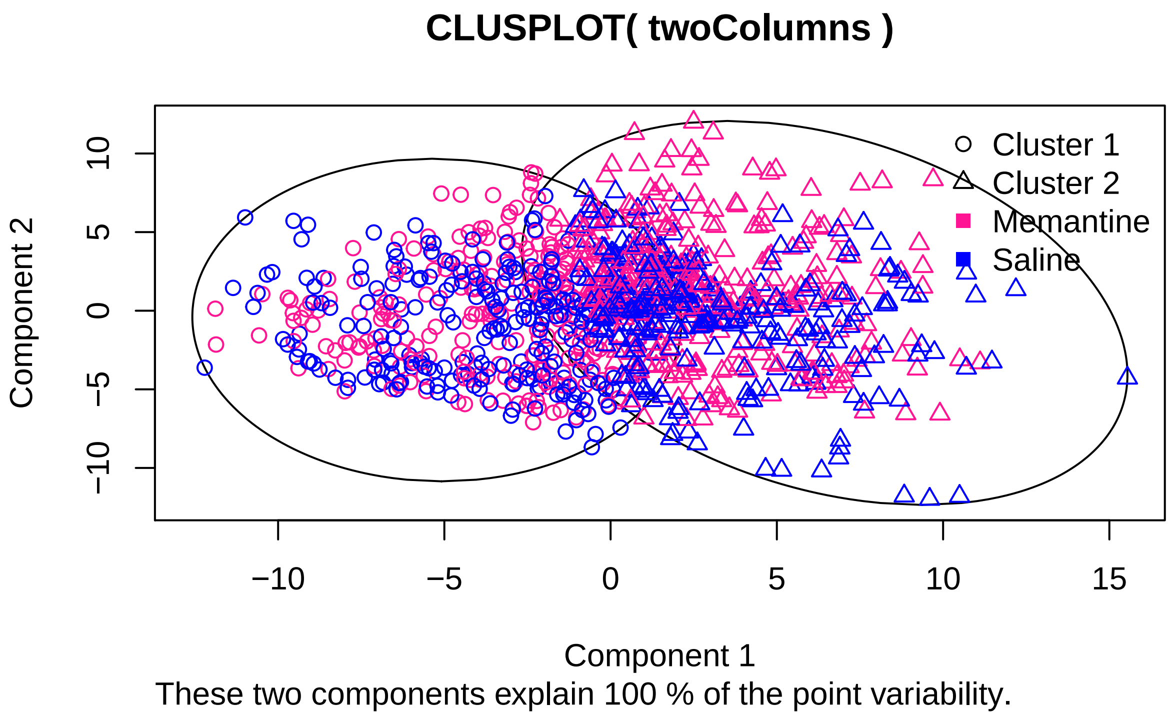


Figure 9.

As you can see, two levels, Memantine and Saline, are equally mixed in the two clusters. The two clusters do not appear to represent any of the two levels.

We will take a closer look at k = 2 clusters and k = 3 cluster.

For now we look more deeply into do k = 2 clusters first.

Now we will plot the two clusters and the classes against different variables.

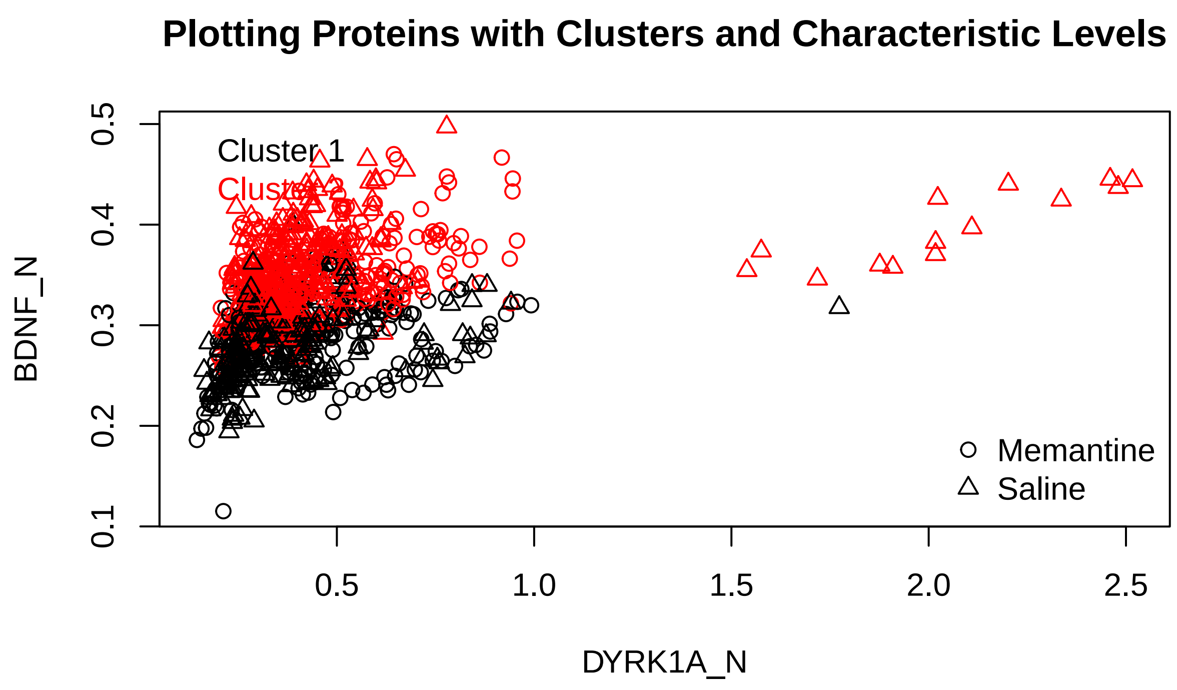


Figure 10.

From this graph we see that data points with values higher than 0.31 for protein BDNF_N are in cluster 2. We will further investigate this with plotting boxplot of the protein value for the 2 clusters and will also do a test to test if the difference in mean of the two clusters is statistically significant.

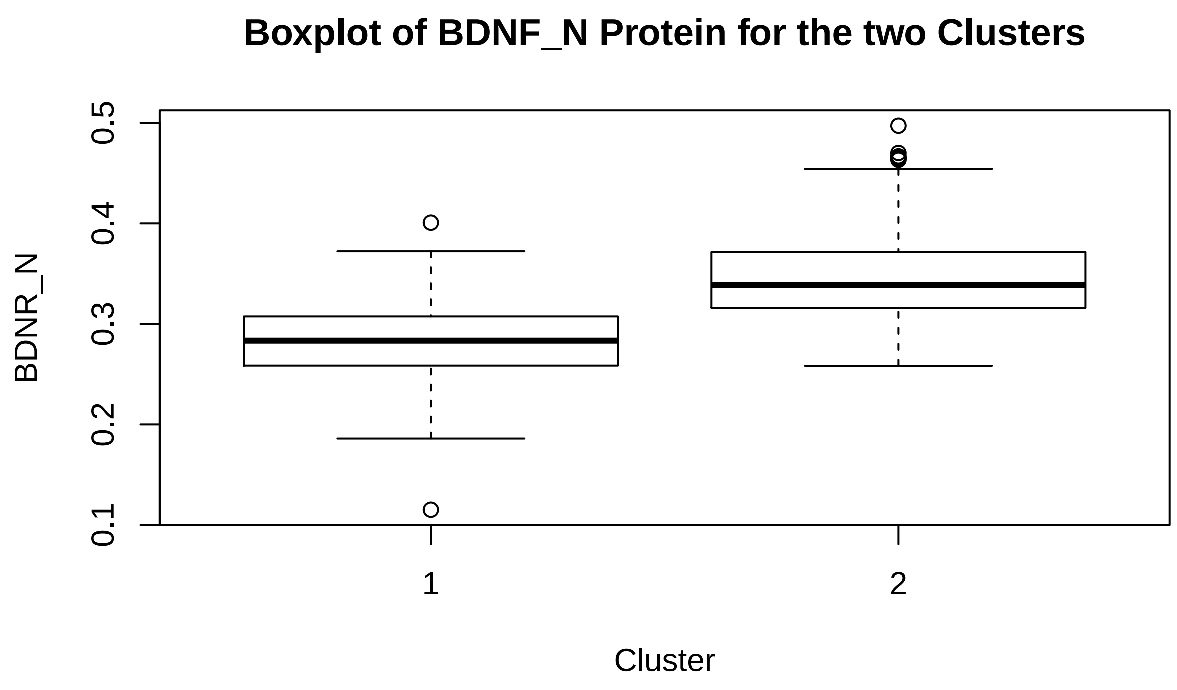


Figure 11.

From the boxplot it appears that the means of the two clusters are different and that cluster 2 has on general higher values.

We will now do the t test.

```
##
##  Welch Two Sample t-test
##
## data:  BDNF_N by cluster
## t = -27.047, df = 1048.5, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.06717986 -0.05809159
## sample estimates:
## mean in group 1 mean in group 2
##       0.2830985       0.3457343
```

From this test it is clear that the two clusters have different mean BDNF_N protein values as p-value <<< 0.001. The the difference in mean is 0.0626 units.

We will now do a t-test for all the 77 proteins comparing their means in the two clusters.

After conducting 77 t-tests we found the following 58 proteins to be statistically significant.

Here below is the table with the mean difference and the p-values.

Table 1: Proteins with Statisitically Significant mean difference between the two clusters.

	Difference.in.mean.of.the.two.clusters	p.value
ITSN1_N	0.0872878	0.0000000
BDNF_N	0.0626357	0.0000000
NR1_N	0.4584640	0.0000000
NR2A_N	1.0964168	0.0000000
pAKT_N	0.0418538	0.0000000
pBRAF_N	0.0285885	0.0000000
pCAMKII_N	1.1709929	0.0000000
pCREB_N	0.0382237	0.0000000
pELK_N	0.2247383	0.0000000
pJNK_N	0.0707897	0.0000000
PKCA_N	0.0515323	0.0000000
pMEK_N	0.0556359	0.0000000
pNR1_N	0.1549727	0.0000000
pNR2A_N	0.2158451	0.0000000
pNR2B_N	0.3662552	0.0000000
pPKCAB_N	0.2086036	0.0000000
pRSK_N	0.0484745	0.0000000
AKT_N	0.1611814	0.0000000
CAMKII_N	0.0615219	0.0000000
CREB_N	0.0229522	0.0000000
ELK_N	0.3420543	0.0000000
ERK_N	0.6334897	0.0000000
GSK3B_N	0.1678937	0.0000000
JNK_N	0.0414745	0.0000000
MEK_N	0.0577354	0.0000000

	Difference.in.mean.of.the.two.clusters	p.value
TRKA_N	0.1564933	0.0000000
RSK_N	0.0242870	0.0000000
APP_N	0.0515581	0.0000000
Bcatenin_N	0.5355866	0.0000000
SOD1_N	0.1465295	0.0000000
MTOR_N	0.0667464	0.0000000
P38_N	0.0502839	0.0000000
pMTOR_N	0.1460418	0.0000000
DSCR1_N	0.0871318	0.0000000
AMPKA_N	0.0662523	0.0000000
NR2B_N	0.1008577	0.0000000
pNUMB_N	0.0537189	0.0000000
RAPTOR_N	0.0462649	0.0000000
TIAM1_N	0.0618101	0.0000000
NUMB_N	0.0164835	0.0000000
P70S6_N	0.1112987	0.0000000
pGSK3B_N	0.0046220	0.0000887
CDK5_N	0.0218891	0.0000000
ADARB1_N	0.3072740	0.0000000
AcetylH3K9_N	-0.0445227	0.0002344
BAX_N	0.0123764	0.0000000
ARC_N	0.0113858	0.0000000
ERBB4_N	0.0106801	0.0000000
nNOS_N	0.0165018	0.0000000
GluR4_N	0.0093229	0.0000000
P3525_N	0.0115913	0.0000000
pCASP9_N	0.1402230	0.0000000
PSD95_N	0.1478144	0.0000000
Ubiquitin_N	0.1388115	0.0000000
pGSK3B_Tyr216_N	0.0230028	0.0000582
pS6_N	0.0113858	0.0000000
pCFOS_N	-0.0064697	0.0000061
SYP_N	0.0647348	0.0000000

All of these proteins were found to have statistically significant different means for the two clusters after the bonferroni correction. The table also provides the mean difference between the two clusters for the each protein. Note that NR2A_N has the greatest mean difference of 1.0964168 among all the proteins. The protein pCFOS_N has the smallest difference in mean of -0.0064697 and may not be clinically significant.

We will now describe the two clusters in the table below for all the variables of the data.

Table 2: Describing the two clusters.

	1	2	p	test
n	459	621		
Genotype = Ts65Dn (%)	234 (51.0)	276 (44.4)	0.039	
Treatment = Saline (%)	225 (49.0)	285 (45.9)	0.339	
Behavior = S/C (%)	189 (41.2)	366 (58.9)	<0.001	
class (%)			<0.001	
c-CS-m	53 (11.5)	97 (15.6)		
c-CS-s	62 (13.5)	73 (11.8)		
c-SC-m	70 (15.3)	80 (12.9)		
c-SC-s	40 (8.7)	95 (15.3)		
t-CS-m	89 (19.4)	46 (7.4)		
t-CS-s	66 (14.4)	39 (6.3)		
t-SC-m	22 (4.8)	113 (18.2)		
t-SC-s	57 (12.4)	78 (12.6)		
DYRK1A_N (mean (sd))	0.40 (0.18)	0.44 (0.29)	0.004	
ITSN1_N (mean (sd))	0.57 (0.19)	0.65 (0.28)	<0.001	
BDNF_N (mean (sd))	0.28 (0.04)	0.35 (0.04)	<0.001	
NR1_N (mean (sd))	2.03 (0.25)	2.49 (0.27)	<0.001	
NR2A_N (mean (sd))	3.21 (0.60)	4.31 (0.85)	<0.001	
pAKT_N (mean (sd))	0.21 (0.04)	0.25 (0.04)	<0.001	
pBRAF_N (mean (sd))	0.17 (0.02)	0.19 (0.02)	<0.001	
pCAMKII_N (mean (sd))	2.86 (1.00)	4.04 (1.26)	<0.001	
pCREB_N (mean (sd))	0.19 (0.03)	0.23 (0.03)	<0.001	
pELK_N (mean (sd))	1.30 (0.41)	1.52 (0.48)	<0.001	
pERK_N (mean (sd))	0.51 (0.26)	0.57 (0.39)	0.004	
pJNK_N (mean (sd))	0.27 (0.05)	0.34 (0.03)	<0.001	
PKCA_N (mean (sd))	0.29 (0.05)	0.34 (0.04)	<0.001	
pMEK_N (mean (sd))	0.24 (0.04)	0.30 (0.04)	<0.001	
pNR1_N (mean (sd))	0.74 (0.08)	0.89 (0.10)	<0.001	
pNR2A_N (mean (sd))	0.60 (0.15)	0.82 (0.16)	<0.001	
pNR2B_N (mean (sd))	1.35 (0.19)	1.72 (0.21)	<0.001	
pPKCAB_N (mean (sd))	1.41 (0.46)	1.61 (0.47)	<0.001	
pRSK_N (mean (sd))	0.42 (0.07)	0.46 (0.06)	<0.001	
AKT_N (mean (sd))	0.59 (0.10)	0.75 (0.10)	<0.001	
BRAF_N (mean (sd))	0.36 (0.16)	0.40 (0.25)	0.003	
CAMKII_N (mean (sd))	0.33 (0.04)	0.39 (0.04)	<0.001	
CREB_N (mean (sd))	0.17 (0.02)	0.19 (0.02)	<0.001	
ELK_N (mean (sd))	0.98 (0.19)	1.32 (0.34)	<0.001	
ERK_N (mean (sd))	2.11 (0.46)	2.74 (0.64)	<0.001	
GSK3B_N (mean (sd))	1.08 (0.22)	1.24 (0.24)	<0.001	
JNK_N (mean (sd))	0.22 (0.03)	0.26 (0.02)	<0.001	
MEK_N (mean (sd))	0.24 (0.03)	0.30 (0.03)	<0.001	
TRKA_N (mean (sd))	0.60 (0.10)	0.76 (0.09)	<0.001	
RSK_N (mean (sd))	0.15 (0.03)	0.18 (0.03)	<0.001	
APP_N (mean (sd))	0.38 (0.05)	0.43 (0.06)	<0.001	
Bcatenin_N (mean (sd))	1.84 (0.29)	2.38 (0.38)	<0.001	
SOD1_N (mean (sd))	0.46 (0.24)	0.61 (0.29)	<0.001	
MTOR_N (mean (sd))	0.41 (0.06)	0.48 (0.05)	<0.001	
P38_N (mean (sd))	0.39 (0.09)	0.44 (0.08)	<0.001	
pMTOR_N (mean (sd))	0.68 (0.12)	0.82 (0.08)	<0.001	

	1	2	p	test
DSCR1_N (mean (sd))	0.54 (0.10)	0.62 (0.09)	<0.001	
AMPKA_N (mean (sd))	0.33 (0.05)	0.40 (0.06)	<0.001	
NR2B_N (mean (sd))	0.51 (0.08)	0.61 (0.07)	<0.001	
pNUMB_N (mean (sd))	0.33 (0.05)	0.38 (0.06)	<0.001	
RAPTOR_N (mean (sd))	0.29 (0.05)	0.34 (0.05)	<0.001	
TIAM1_N (mean (sd))	0.38 (0.05)	0.44 (0.06)	<0.001	
pP70S6_N (mean (sd))	0.38 (0.15)	0.40 (0.16)	0.016	
NUMB_N (mean (sd))	0.17 (0.03)	0.19 (0.03)	<0.001	
P70S6_N (mean (sd))	0.88 (0.16)	0.99 (0.17)	<0.001	
pGSK3B_N (mean (sd))	0.16 (0.02)	0.16 (0.02)	<0.001	
pPKCG_N (mean (sd))	1.71 (0.61)	1.71 (0.55)	0.967	
CDK5_N (mean (sd))	0.28 (0.04)	0.30 (0.03)	<0.001	
S6_N (mean (sd))	0.43 (0.14)	0.43 (0.14)	0.253	
ADARB1_N (mean (sd))	1.02 (0.29)	1.33 (0.36)	<0.001	
AcetylH3K9_N (mean (sd))	0.24 (0.23)	0.20 (0.15)	<0.001	
RRP1_N (mean (sd))	0.17 (0.03)	0.17 (0.04)	0.285	
BAX_N (mean (sd))	0.17 (0.02)	0.18 (0.02)	<0.001	
ARC_N (mean (sd))	0.11 (0.01)	0.13 (0.01)	<0.001	
ERBB4_N (mean (sd))	0.15 (0.01)	0.16 (0.01)	<0.001	
nNOS_N (mean (sd))	0.17 (0.02)	0.19 (0.02)	<0.001	
Tau_N (mean (sd))	0.21 (0.08)	0.21 (0.06)	0.257	
GFAP_N (mean (sd))	0.12 (0.01)	0.12 (0.01)	0.964	
GluR3_N (mean (sd))	0.22 (0.03)	0.22 (0.04)	0.187	
GluR4_N (mean (sd))	0.12 (0.02)	0.13 (0.03)	<0.001	
IL1B_N (mean (sd))	0.52 (0.08)	0.53 (0.08)	0.210	
P3525_N (mean (sd))	0.28 (0.03)	0.30 (0.03)	<0.001	
pCASP9_N (mean (sd))	1.47 (0.23)	1.61 (0.24)	<0.001	
PSD95_N (mean (sd))	2.15 (0.25)	2.30 (0.24)	<0.001	
SNCA_N (mean (sd))	0.16 (0.02)	0.16 (0.02)	0.001	
Ubiquitin_N (mean (sd))	1.16 (0.15)	1.30 (0.16)	<0.001	
pGSK3B_Tyr216_N (mean (sd))	0.84 (0.09)	0.86 (0.10)	<0.001	
SHH_N (mean (sd))	0.23 (0.03)	0.23 (0.03)	0.859	
BAD_N (mean (sd))	0.16 (0.03)	0.16 (0.03)	0.600	
BCL2_N (mean (sd))	0.14 (0.03)	0.14 (0.03)	0.608	
pS6_N (mean (sd))	0.11 (0.01)	0.13 (0.01)	<0.001	
pCFOS_N (mean (sd))	0.13 (0.02)	0.13 (0.02)	<0.001	
SYP_N (mean (sd))	0.41 (0.06)	0.47 (0.06)	<0.001	
H3AcK18_N (mean (sd))	0.18 (0.07)	0.17 (0.05)	0.055	
EGR1_N (mean (sd))	0.19 (0.04)	0.19 (0.04)	0.722	
H3MeK4_N (mean (sd))	0.21 (0.06)	0.21 (0.05)	0.140	
CaNA_N (mean (sd))	1.35 (0.31)	1.33 (0.32)	0.463	

As you can imagine, this table can be very interesting to a researcher. We can see several interesting things from this. The class t-SC-m is mainly found in cluster 2 (113 out of 135 (83%) t-SC-m's were in cluster 2). Further, from the same table we can see that more S/C Behavior points are found in cluster 2 (366 out of 555 (70%) S/C's were in cluster 2).

We looked at several more plots and the found the following two very interesting.

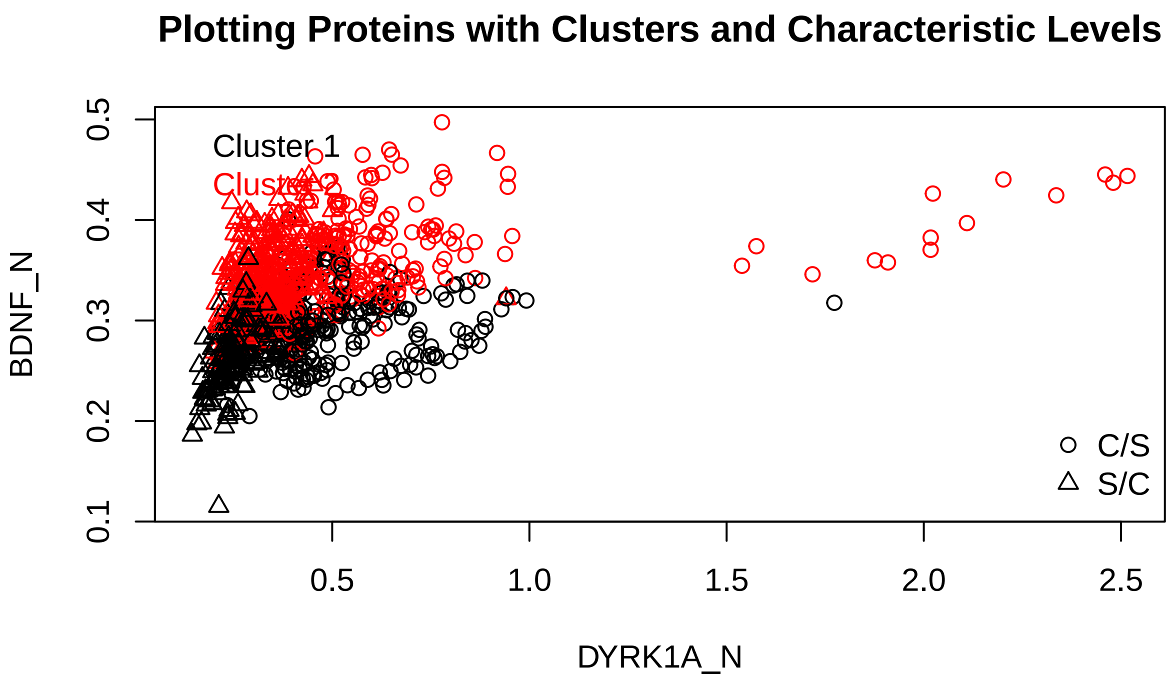


Figure 12.

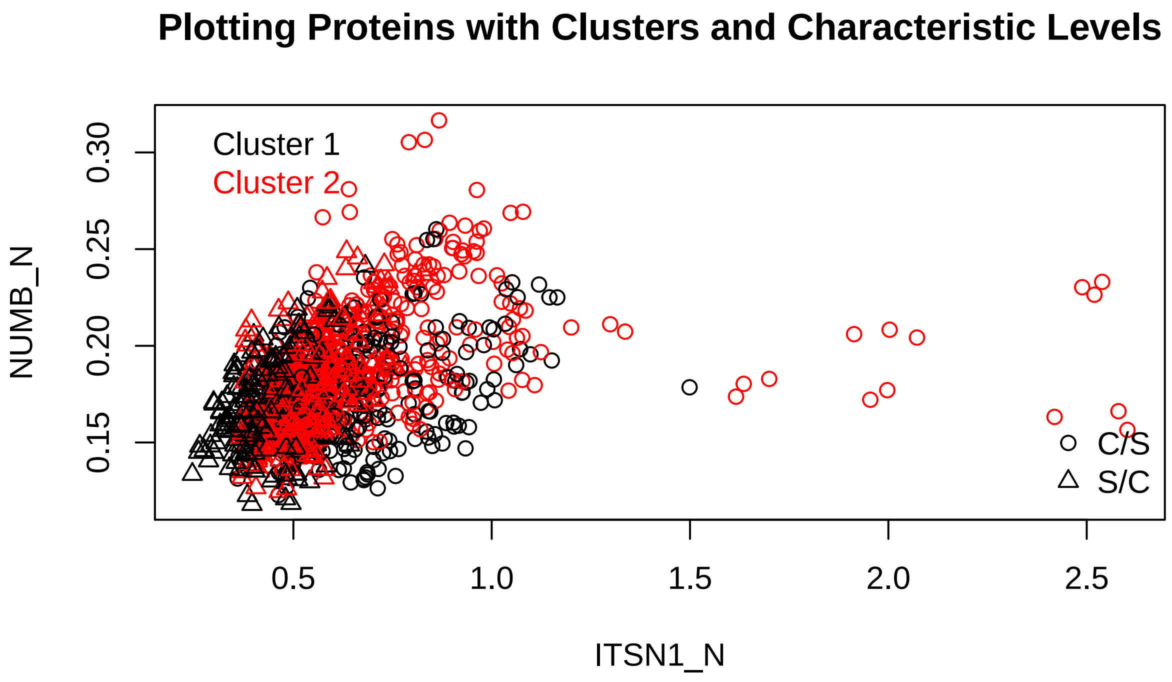


Figure 13.

From the two graphs above we notice that the behaviour level C/S has higher values for the ITSN1_N and

BDNF_N proteins.

We investigate this a bit further and perform a t-test to verify our finding.

```
##
##  Welch Two Sample t-test
##
## data:  ITSN1_N by Behavior
## t = 17.727, df = 640.5, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.2159409 0.2697402
## sample estimates:
## mean in group C/S mean in group S/C
##         0.7419253         0.4990847
```

We find the difference in mean between the two groups to be highly statistically significant.

We now do a t-test for the protein BDNF_N and compare the mean difference between the behavior groups of C/S and S/C.

```
##
##  Welch Two Sample t-test
##
## data:  BDNF_N by Behavior
## t = 4.7878, df = 1044.8, p-value = 1.929e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.008427233 0.020131762
## sample estimates:
## mean in group C/S mean in group S/C
##         0.3264522         0.3121727
```

Again, we find if to be statistically significant, but not as strongly as ITSN1_N

Now we do k-means clustering for 3 clusters.

We looked at many k = 3 clustering plots but only found the above two plots with proteins BDNF_N and DYRK1A_N and proteins pELK_N and P3525_N to have an interesting clustering pattern.

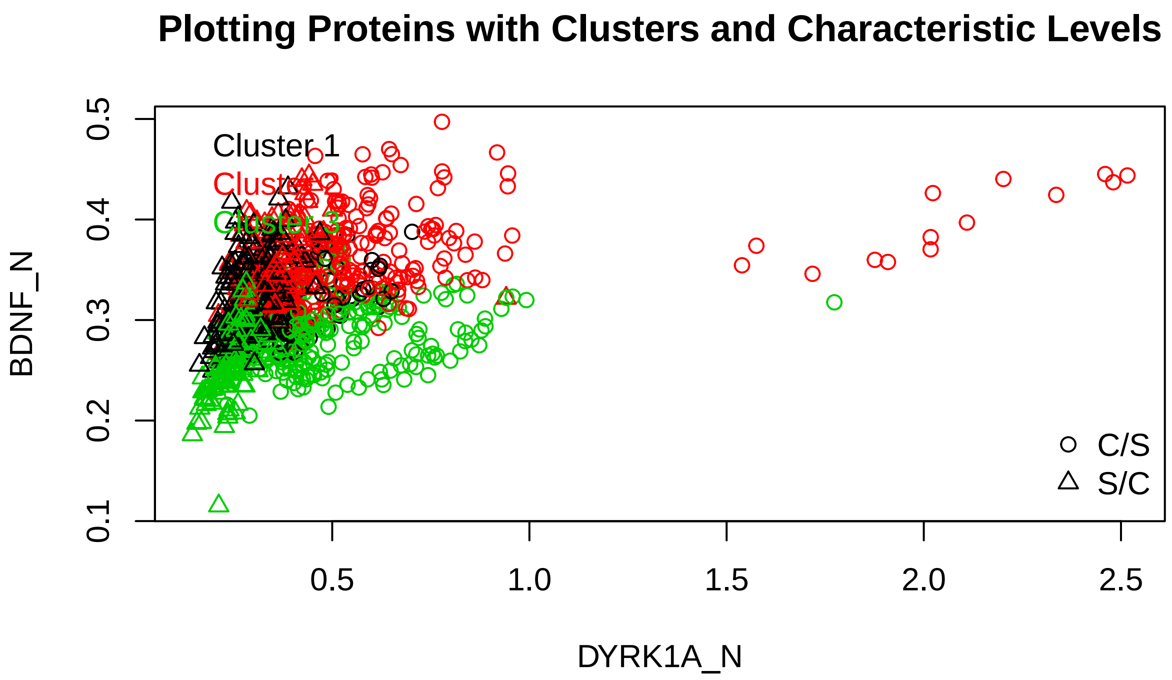


Figure 14.

In the above plot we can see that the clusters have an interesting pattern. Cluster 2 is on the lower right bottom of the values. Cluster 1 is on the upper left side of the values. The two clusters make a wedge and in between the two clusters, cluster 3 is found. It can also be seen that almost all S/C points are below the value 0.5 for the DYRK1A_N protein (x-axis).

```
##
##  Welch Two Sample t-test
##
## data:  DYRK1A_N by Behavior
## t = 18.762, df = 578.08, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.2253937 0.2781009
## sample estimates:
## mean in group C/S mean in group S/C
##         0.5551168         0.3033695
```

We now plot the clustering with the proteins pELK_N and P3525_N.

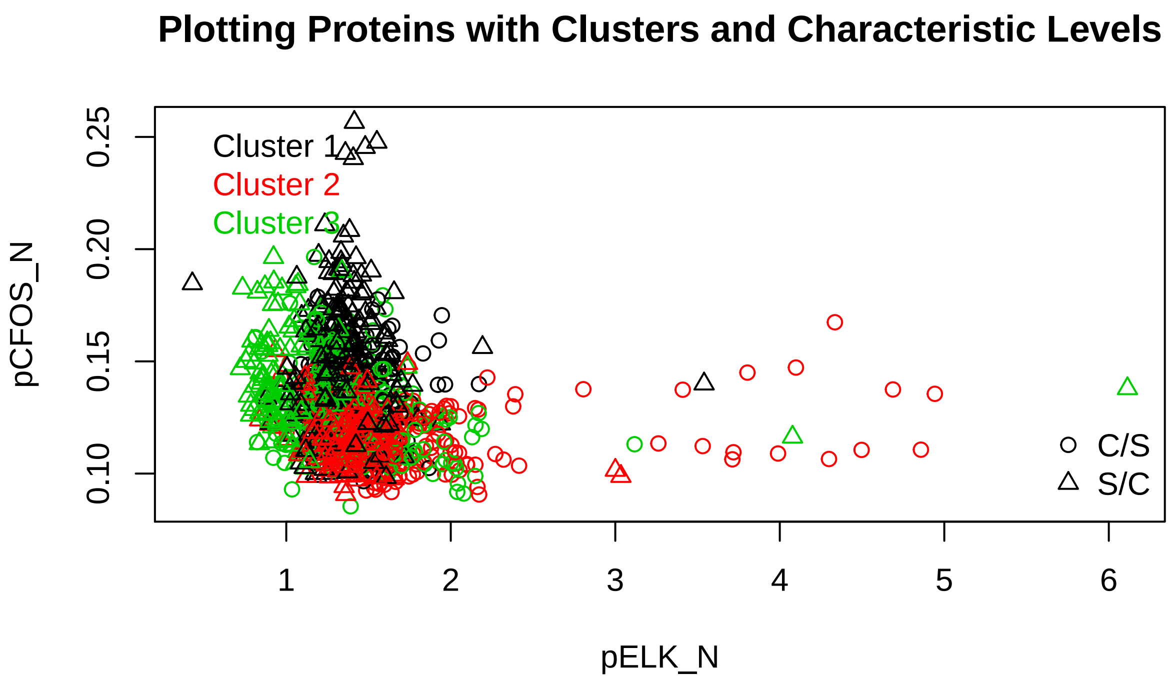


Figure 15.

From the above graph it can be seen that the clustering structure for these proteins is similar to in the previous graphs except that cluster 2 and cluster 3 create the wedge and cluster 1 is found in between them.

We present the concordance tables of the k = 3 k-means clustering.

```
##    labels_genotype
##     Control Ts65Dn
##   1     203    156
##   2     208    171
##   3     159    183
```

```
##    labels_behavior
##     C/S S/C
##   1  81 278
##   2 217 162
##   3 227 115
```

```
##    labels_treatment
##     Memantine Saline
##   1       226    133
##   2       183    196
##   3       161    181
```

```
##    labels_class
##     c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
##   1     33     20     96     54     13     15     84     44
##   2     76     62     23     47     41     38     43     49
##   3     41     53     31     34     81     52      8     42
```

From the above we see that fro Behavior cluster 1 has mostly C/S data points (278 out of 359).

4.6 Hierarchical clustering

We will now perform hierarchical clustering.

First, we will do clustering using the three linkages: complete, single, and average, and plot the clustering.

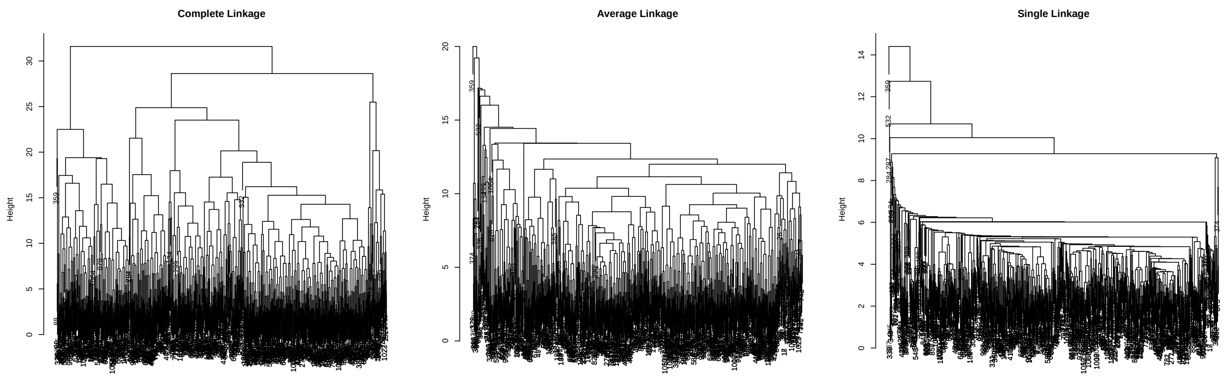


Figure 16.

From the above plot it is appears that the complete linkage creates the most balanced clustering and looks the most reasonable.

Further, for k = 2, we will plot the resulting clusters for the proteins BRAF_N and ERBB4_N below.

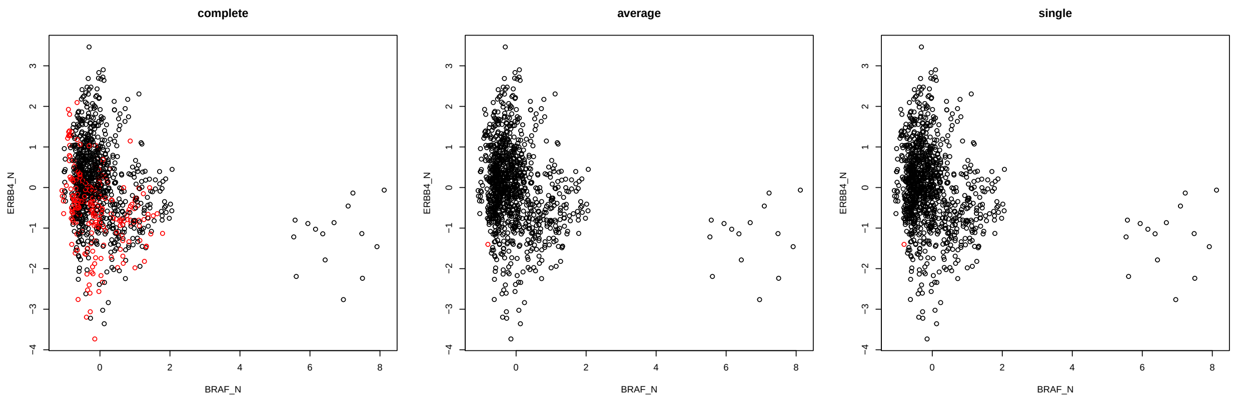


Figure 17.

Again, from the above graph complete linkage creates does most balanced clustering and looks the most reasonable. We can see from the above graph that the average and single linkages group majority of the points into one cluster. Hence, we will use complete linkage for hierarchical clustering.

Also, it is quite plain from the complete linkage plot that, at the high level, there are two main groups. This is consistent with what we found in our heatmap of the data as well as when we calculated the hierarchical clustering Silhouette plot.

Further, and then the second group has another distinct two groups. Hence, one would say that the data can be clustered also into three main groups, with the second cluster being divided into two further subgroups.

From the complete linkage plot it can be seen that if we like the data can in fact be grouped into five groups with Cluster 1 having two main subgroup, Cluster 2 also having two main subgroups, and Cluster 3 not having any real subgroup would just be the fifth group. Hence, we would have Cluster 1a, Cluster 1b, Cluster 2a, Cluster 2b, and Cluster 3 as our five clusters if we wanted to have 5 clusters instead of three.

Now, we will do a similar visualization of the clustering using the PCA dimension reduction trick.

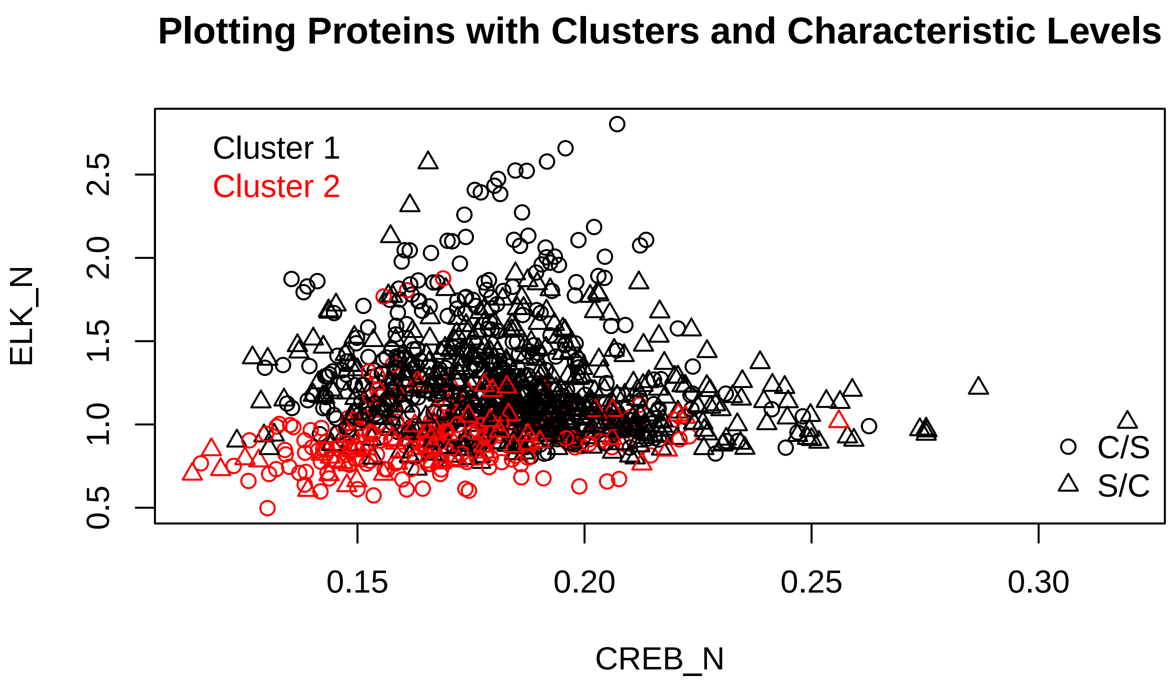


Figure 18.

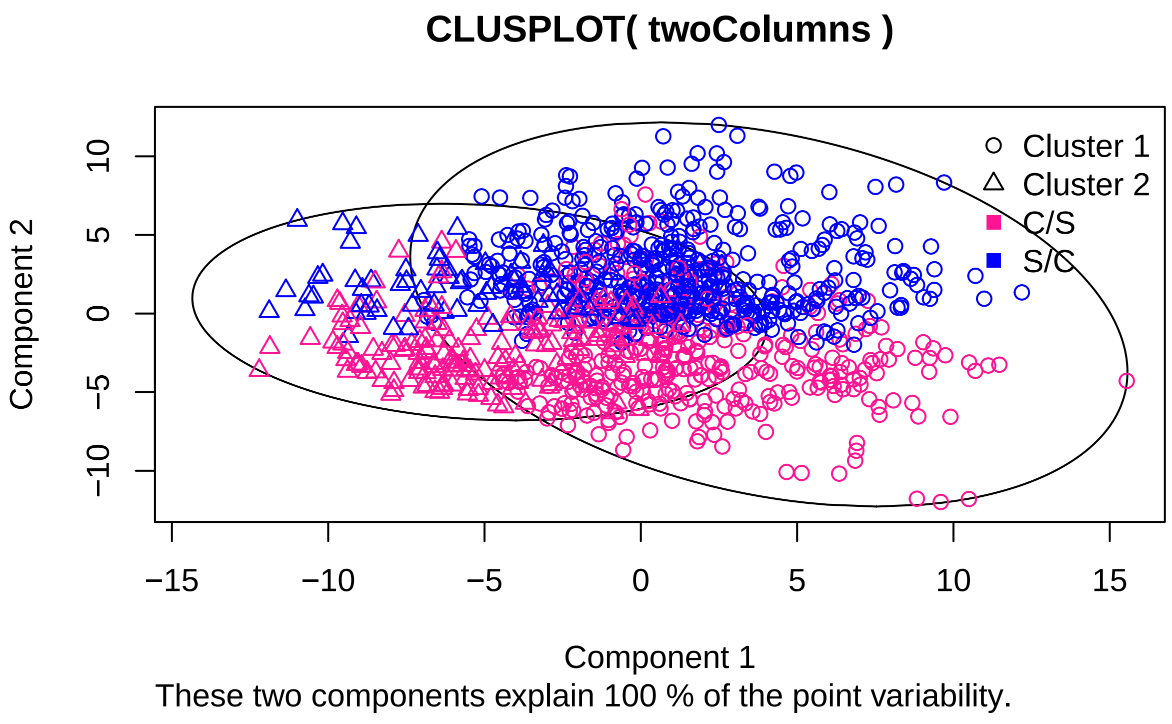


Figure 19.

We notice from the above graph, for k = 2, that the clustering is different compared to what we had for k-means clustering. The two clusters for k-means were of similar sizes. Here we can clearly see that hierarchical clustering produces clusters, where one is very large and the other is much smaller. Although the shapes appear very similar to that of k-means clustering.

We also notice that the coding is inverse to that of k-means (cluster 1, the circles were on the right had side).

We present the concordance table below, after correcting for this inversion.

```
##             labels_behavior
## cut.tree.com C/S S/C
##            1 167  70
##            2 358 485
```

```
##             labels_treatment
## cut.tree.com Memantine Saline
##            1       100    137
##            2       470    373
```

```
##             labels_genotype
## cut.tree.com Control Ts65Dn
##            1      96    141
##            2     474    369
```

```
##             labels_class
## cut.tree.com c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
##            1     35     36      5     20     60     36      0     45
##            2    115     99    145    115     75     69    135     90
```

We compare the results of the k=2 hierarchical clustering with the k-means clustering.

Table 3: Comparing clustering of hierarchical clustering (rows) with k-means clustering for k = 2.

	Kmeans Cluster 1	Kmeans Cluster 2
1	218	19
2	241	602

We see that approximately 24% of points were clustered differently.

We now do some clustering for k = 3.

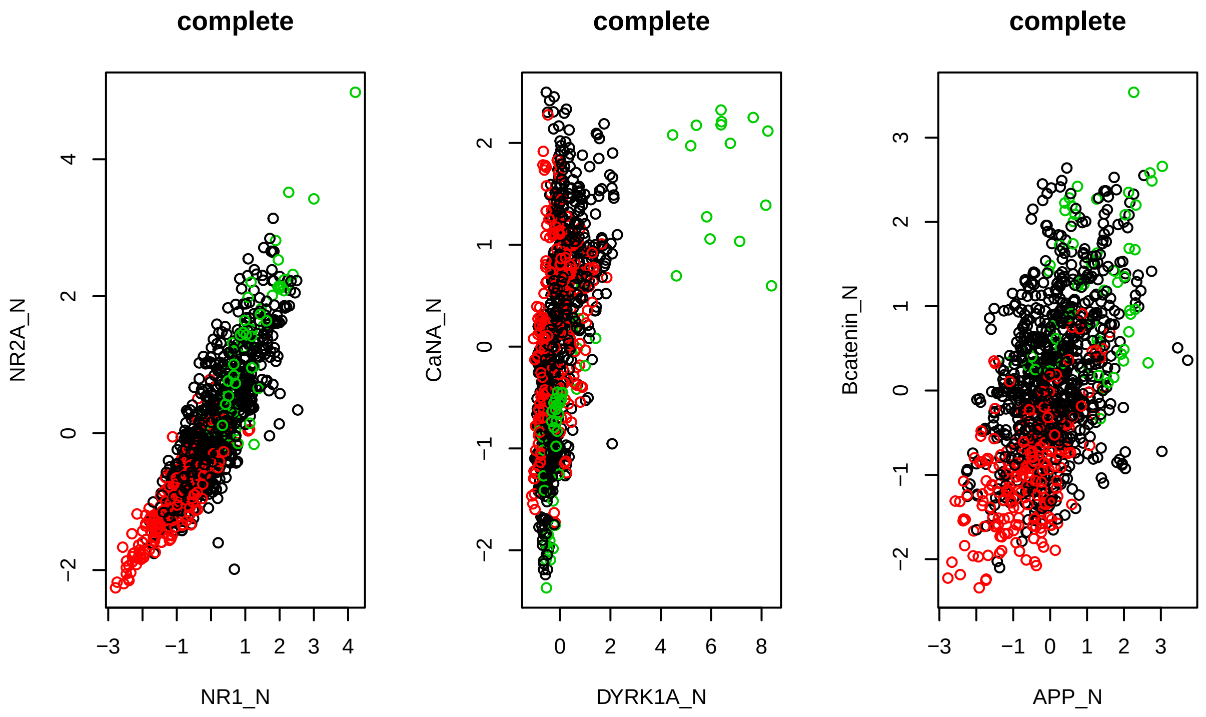


Figure 20.

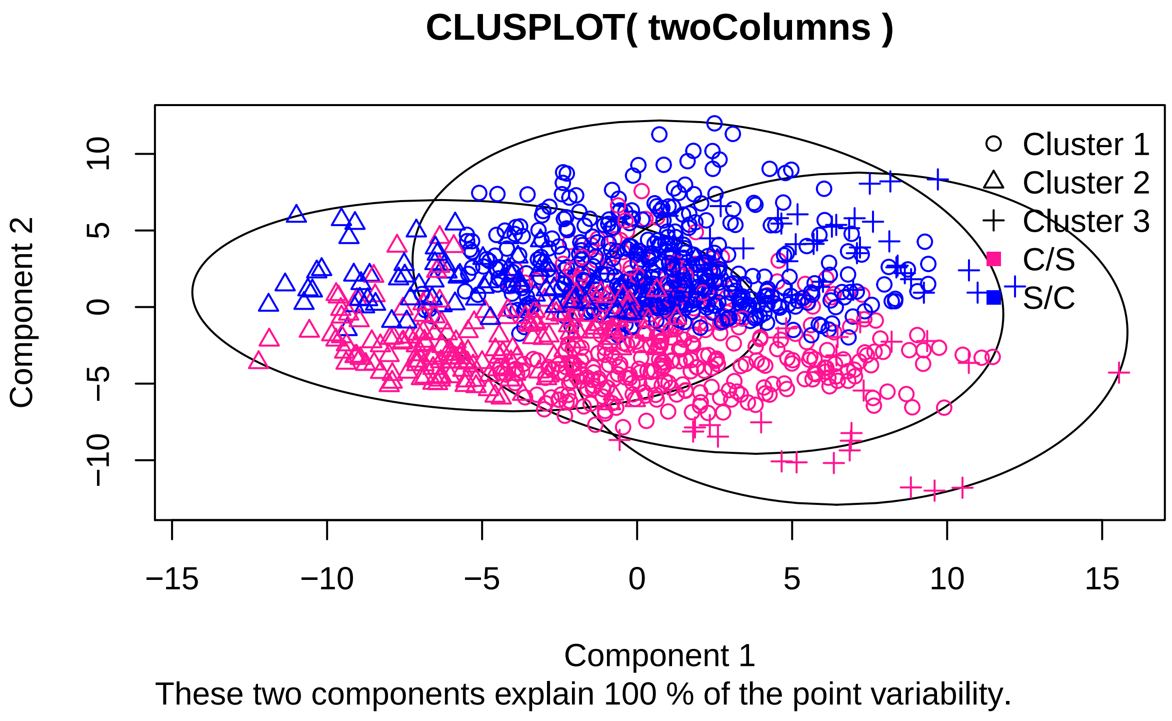


Figure 21.

```
##              labels_behavior
## cut.tree.com C/S S/C
##            1 332 449
##            2 167  70
##            3  26  36
```

```
##             labels_treatment
## cut.tree.com Memantine Saline
##            1       455    326
##            2       100    137
##            3        15     47
```

```
##             labels_genotype
## cut.tree.com Control Ts65Dn
##            1     447    334
##            2      96    141
##            3      27     35
```

```
##             labels_class
## cut.tree.com c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
##            1    115     73    144    115     75     69    121     69
##            2     35     36      5     20     60     36      0     45
##            3      0     26      1      0      0      0     14     21
```

We notice that cluster three has very few points for all four characteristics.

One interesting thing we noticed on how the clusters produced from hierarchical clusters is better when making mutually exclusive clustering, at least for k = 3, for our data. From the concordance table of the class, there is less of an overlap.

We now do k = 8 and see the concordance table for the class characteristic, and see if we get 8 individual clusters.

```
##             labels_class
## cut.tree.com c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
##            1     39     15      0     12     17     12     18     11
##            2     54     58    108     89     58     54     58     58
##            3     32     35      5     20     60     36      0     45
##            4     22      0     34     14      0      3     45      0
##            5      3      1      0      0      0      0      0      0
##            6      0      0      2      0      0      0      0      0
##            7      0     11      1      0      0      0     14     21
##            8      0     15      0      0      0      0      0      0
```

For k = 8, we do not see any distinct clusters representing any of the 8 levels of the class distinctly.

Lastly, we try k = 12 to see if we get any clusters that represent the 8 levels of the class distinctly.

```
##             labels_class
## cut.tree.com c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
##           1      39     15      0     12     17     12     18     11
##           2      44     30     75     76     49     42     51     43
##           3      20      6      0      0     24     24      0     20
##           4      10     28     33     13      9     12      7     15
##           5      22      0     34     14      0      3     45      0
##           6      12     29      5      5     36     12      0     18
##           7       3      1      0      0      0      0      0      0
##           8       0      0      2      0      0      0      0      0
##           9       0      6      1      0      0      0     14     21
##           10      0     15      0      0      0      0      0      0
##          11    0    5    0     0    0    0    0    0
##          12    0    0    0    15    0    0    0    7
```

Again, we do not. Mostly, all the points are in cluster 1 to 6 and the other clusters have very few points in them.

4.7 Principal Component Analysis: Proteins that exhibit a distinct expression pattern (profile)

Do we see any particular set of proteins that exhibit a distinct expression pattern (profile) in any or all of these clusters? (You may want to consider PCA, loadings and biplots for that question)

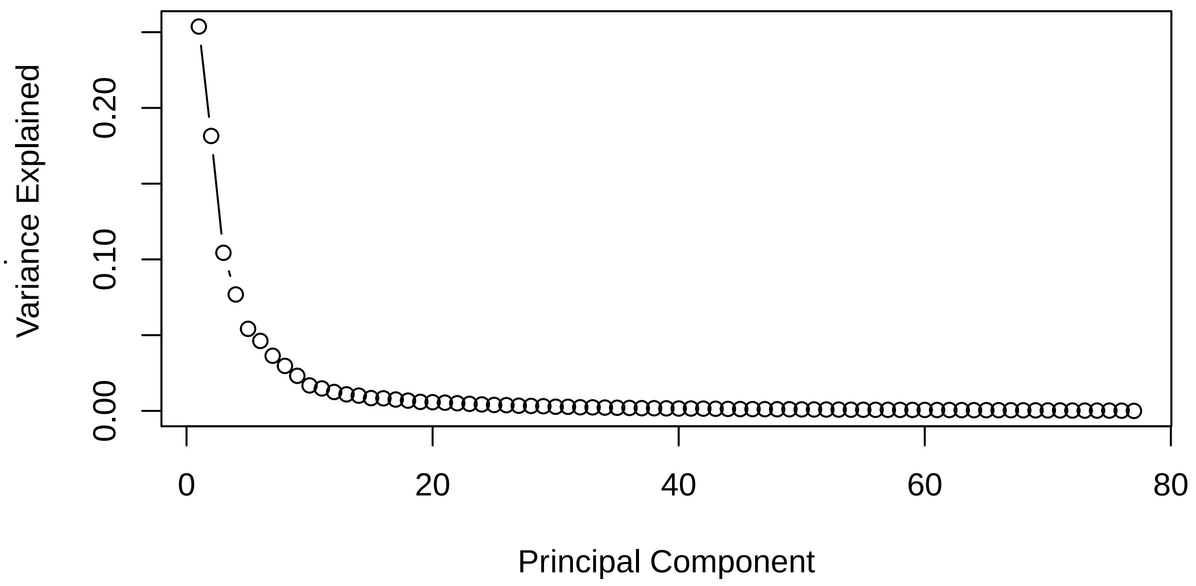


Figure 22.

There is a drop after about 5 pc's.

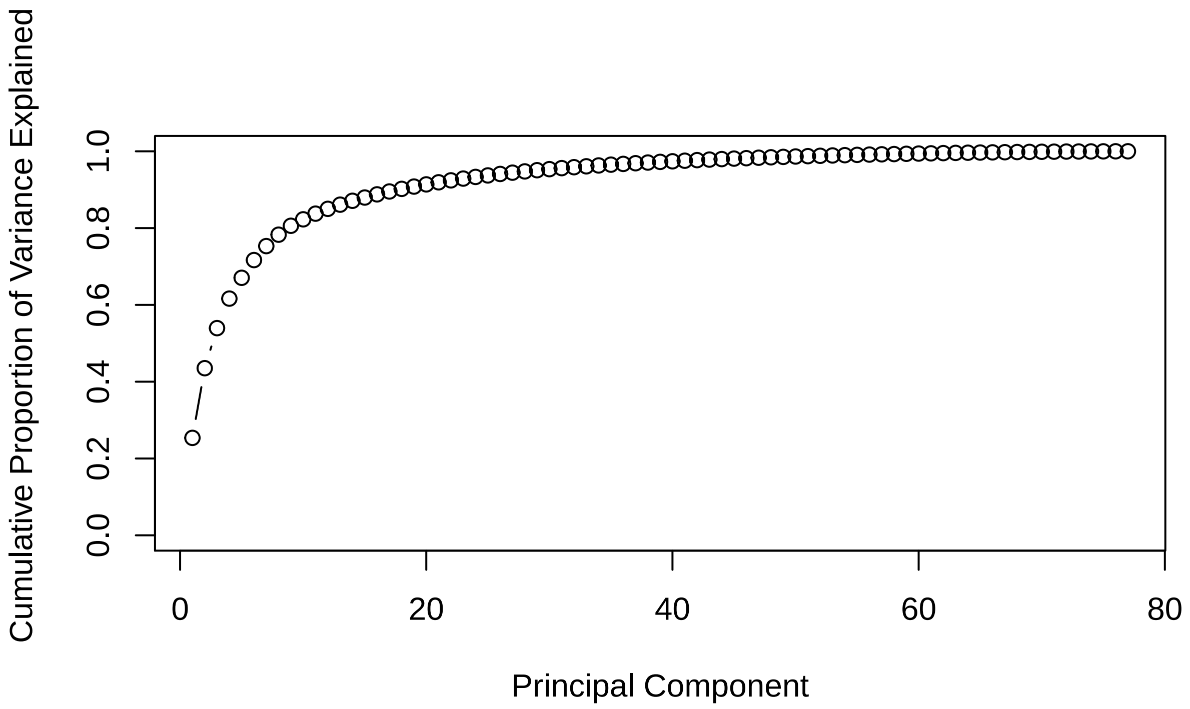


Figure 23.

```
##       PC1       PC2       PC3       PC4       PC5       PC6       PC7
## 0.2536976 0.4351741 0.5395393 0.6163944 0.6705579 0.7167573 0.7531531
##       PC8       PC9
## 0.7828619 0.8060153
```

We see that the first 9 pc's explain 80.6% of the total variance.

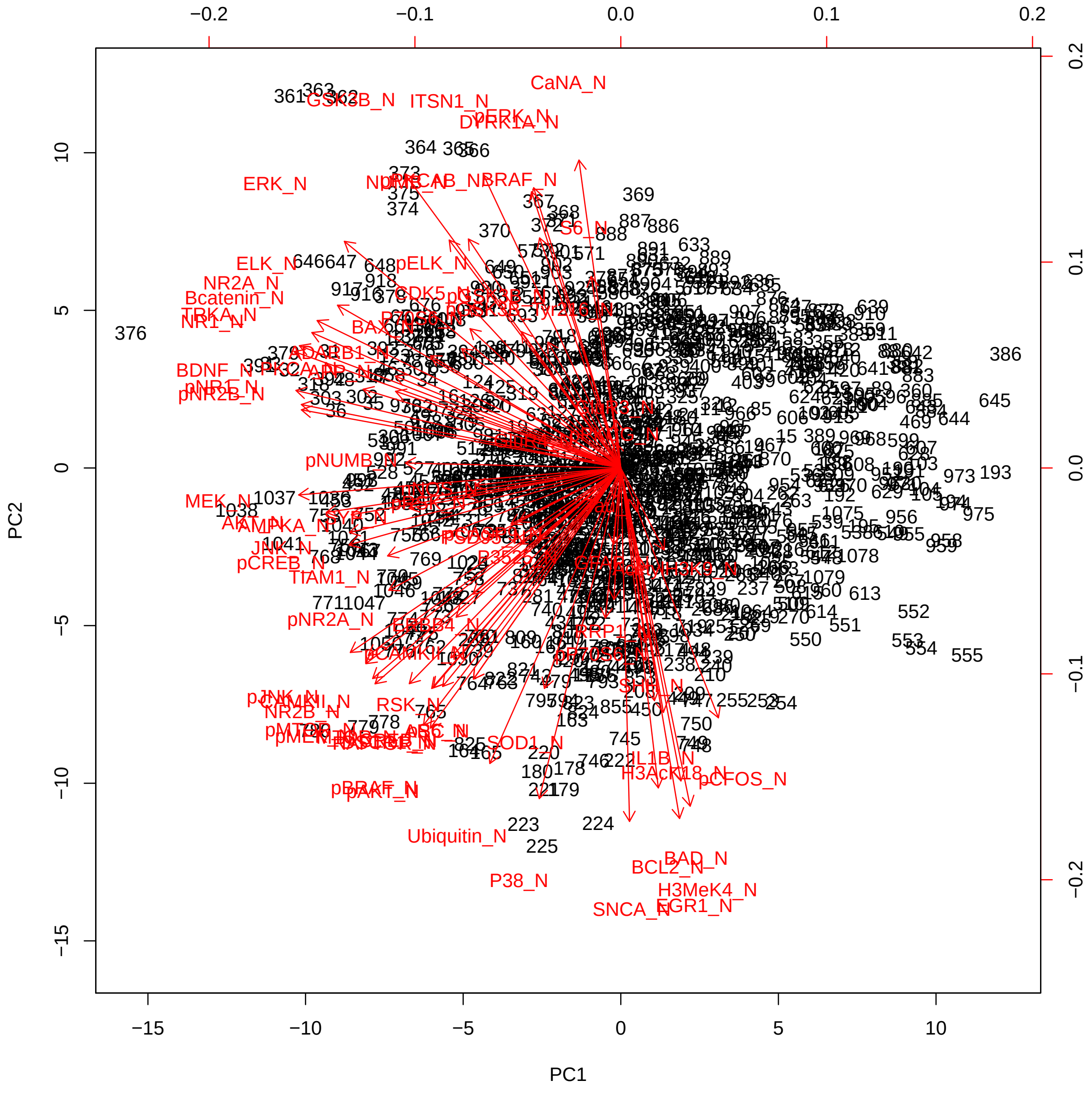


Figure 24.

There is a distinct pattern in the biplot. All the proteins are pointing left of the y-axis.

We identify which variables have the largest effect on each principal component. Loadings close to -1 or 1 indicate that the variable strongly influences the component. Interestingly, we see in the plot that none of the variables are positively correlated with the first principal component. Further, almost all the variables can be grouped, as either being positively or negatively correlated to the second principal component, or being negatively correlated to the first pc.

Hence, almost all the proteins can be grouped into two clusters: either they are positively correlated to the second principal component or negatively correlated to the second principal.

5 Discussion

Using any of the methods and techniques you learned under unsupervised learning (clustering, PCA, dissimilarity metrics, heatmap plots, biplots etc.) investigate any patterns in the data in order to answer questions like the following:

Q: Are there any distinct clusters in the measurements?

Yes, there are several which we discuss in the results section.

Q: Are these clusters associated with any of the different genotype, behaviour and/or treatment? E.g. you may find that some of these clusters are more representatives of some of the classes described above.

Yes, this is discussed in the Results Summary and Results sections.

Q: Do we see any particular set of proteins that exhibit a distinct expression pattern (profile) in any or all of these clusters? (You may want to consider PCA, loadings and biplots for that question)

Discussed, above.

Q: Bonus question:In the data we consider the samples as independent. The reality is that most likely they are not since multiple measurements are coming from the same mouse. Are there any implications about this? Discuss possible ways if any that you could improve the overall analysis plan by taking into consideration this potential cor-relatedness in the data.

Clearly, the samples are not independent as multiple measurements are coming from the same mouse. This is most likely the reason so many variables were correlated strongly, and mostly positively, to one another in the data (see Figure 1). The implications of this will be that some relationships will appear to be strong between variables when this in fact is simply due to the fact that multiple measurements are coming from the same mouse. If a mouse has a high value for a protein, then the other proteins may also have high values simply because they are coming form the same sample.

One way to take into account the fact that the data points are not independent, is to first create two columns; the first indicating the unique code of the mouse and the second indicating the measurement number indicator which would run from 1 to 15. Just by simply having these two columns themselves in the random forest imputation method would improve the imputed values for the missing data.

Secondly, we would segment the data into the number of mice and run the analysis on each mice data and then see the over all pattern of the results and draw our conclusions from that.

A third way, would be to create dummy variables in the data set that would serve as indicator variables for each mouse. So, since there are a total of 72 mice, we would create 71 dummy variables. These variables could then be used in the subsequent analysis, and would in a way account for having points that are correlated to one another since they are coming from the same mouse.

Finally, some form of repeated measures model should be used when conducting the analysis.

6 Question 2

Think of ways that you could use some of the unsupervised learning methods in problems you are or have recently faced in your research or work. Explain how and why they can be helpful in investigating the problems and answering the questions you are dealing with.

In one of my research work I had to find out what were the characteristics of the successful and sustainable charities in Canada using the tax return data that was provided by CRA of all 23,000 Canadian charities.

There was a very large number of variables, so the first thing we did was dimension reduction by performing PCA. Using the first three PCA's we then used those as input for k-mean clustering, where we used different values of k. We found that the best clustering was for when k was 3 and there were three main groups of charities, each had varying degree of sustainability/success. We measured sustainability/success of each group, each cluster, by measuring which cluster had the least number of charities that went bankrupt.

Now, we found that the three types of charities that the k-mean clustering were as follows: 1) The charities whose income source was primarily based on donations. 2) The charities whose income source was primarily based on self created income, such as investment or providing some service. 3) The charities whose income source was primarily based on funding from the government.

The charities with government funding were the most sustainable, followed by charities based on self created income and the least sustainable were donation based ones. This, makes very much sense especially, since what we also found that charities based on income and donations could had a very high chance of going bankrupt if a recession came. The charities who were government funded were sort of "immune" to a recession.

Unsupervised learning methods were very important here. It was very important to reduce the number of features, as there were over 100 variables in the dataset and it was very difficult to run.

Further, the research question was to find sustainable/successful charities, however, neither success nor sustainability were defined in the question. Only, after we had done the PCA and the k-mean clustering that it occurred to us that the number of charities that went bankrupt were significantly different in each of the 3 clusters.

Lastly, without the clustering and feature reduction techniques of unsupervised clustering, it would not have really been possible to answer this research question.

7 References

Morris, Tim P, Ian R White, and Patrick Royston. 2014. "Tuning Multiple Imputation by Predictive Mean Matching and Local Residual Draws." *BMC Medical Research Methodology* 14 (1). BioMed Central: 75.

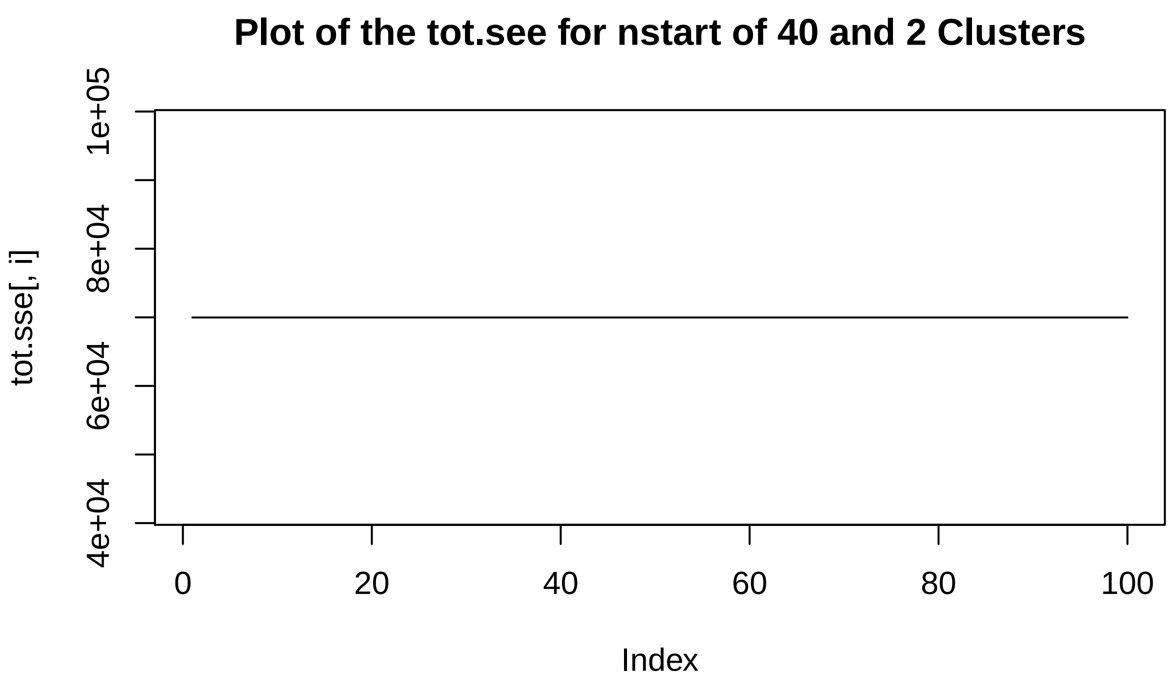
Shah, Anoop D, Jonathan W Bartlett, James Carpenter, Owen Nicholas, and Harry Hemingway. 2014. "Comparison of Random Forest and Parametric Imputation Models for Imputing Missing Data Using Mice: A Caliber Study." *American Journal of Epidemiology* 179 (6). Oxford University Press: 764–74.

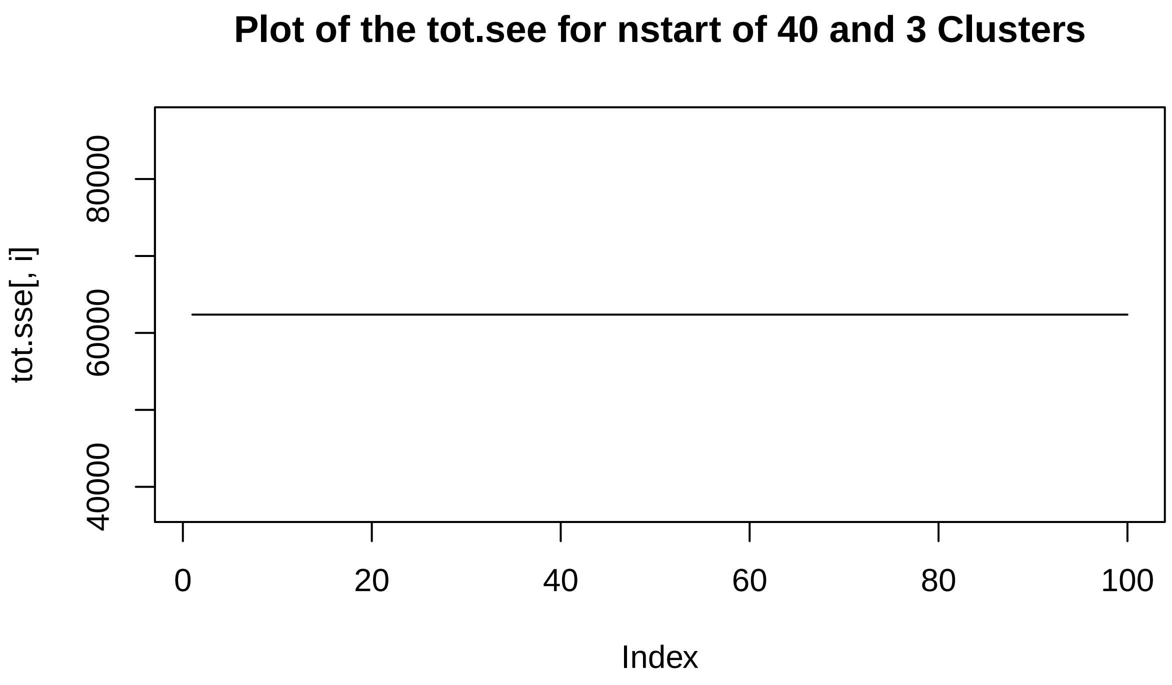
8 Appendix

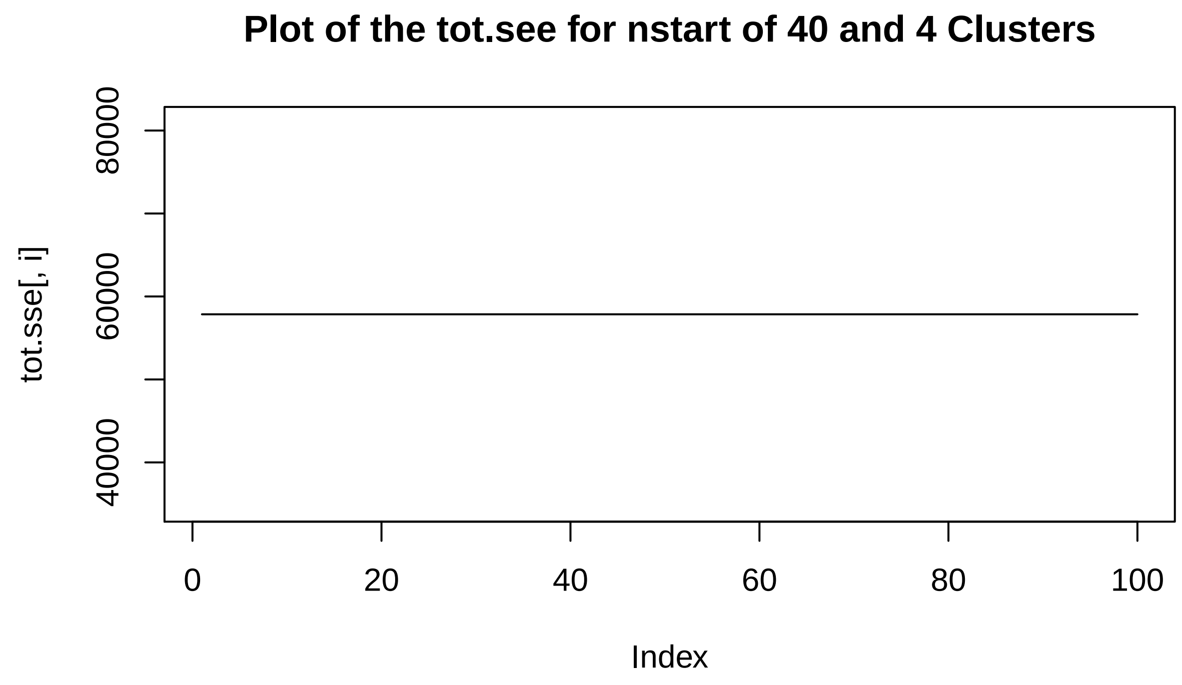
8.1 Appendix A.1. Plots of tot.sse for different values of k in kmeans clustering.

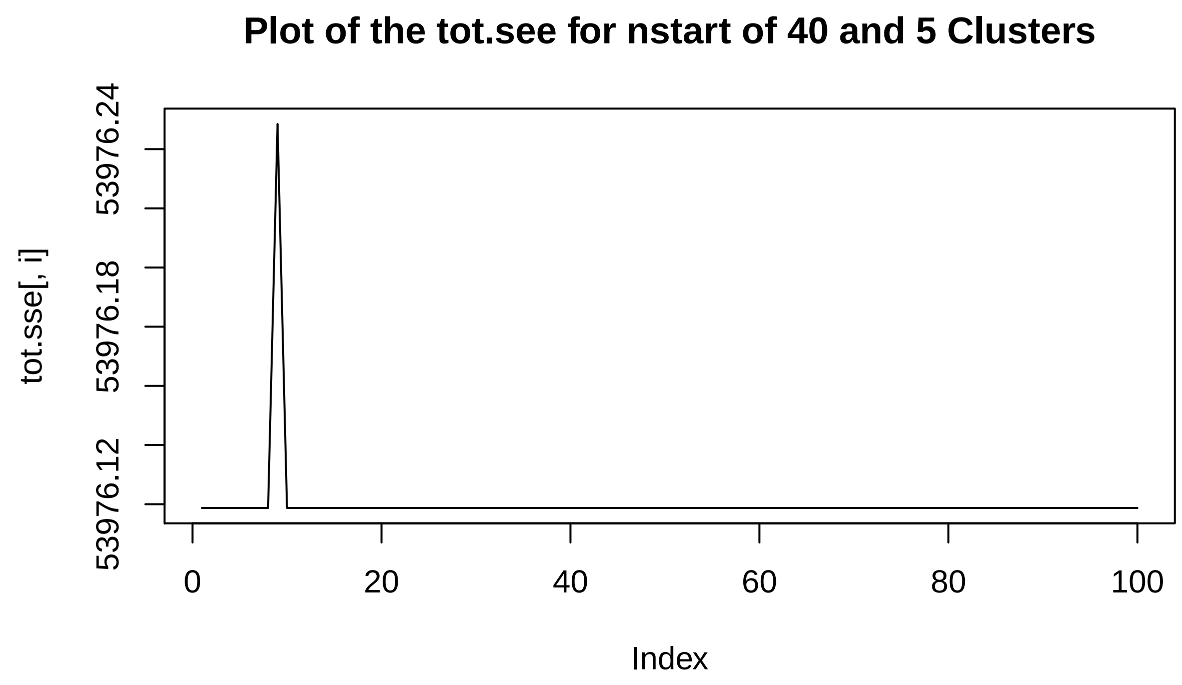
Now to choose the value of nstart for the kmeans clustering we will plot the graph of the tot.sse for kmeans clustering for k from 2 to 8. With nstart equal to 40.

```
tot.sse <- matrix(NA, nrow = 100, ncol = 4 )
for(k in 2:5){

  j = k - 1
  for (i in 1:100){
    km.out <- kmeans(kdata.scale,k,nstart=40) ##If this is not a straight line increase nstart.
    tot.sse[i, j] <- km.out$tot.withinss
  }

  colnames(tot.sse) = paste0("k", (2:(ncol(tot.sse)+1)))
}

for ( i in 1:4){
  plot(tot.sse[,i],type="l", main = paste("Plot of the tot.see for nstart of 40 and", i+1 , "Clusters")
# Clearly the plot of tot.sse is stable
}
```





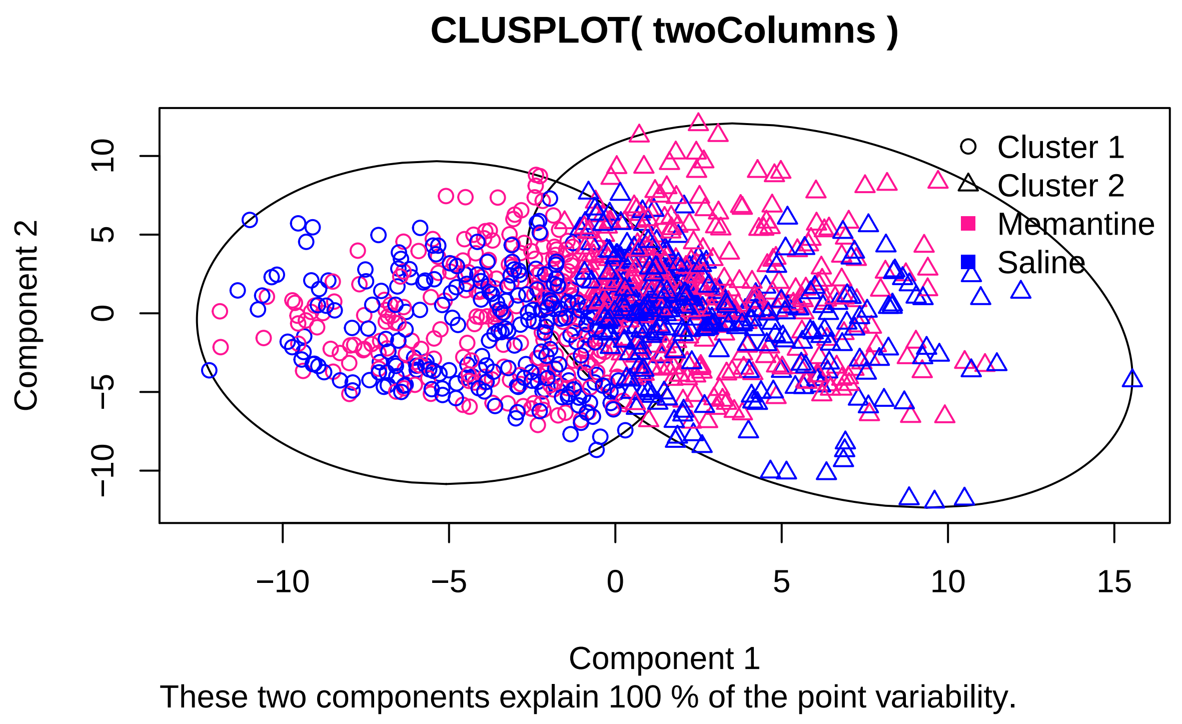




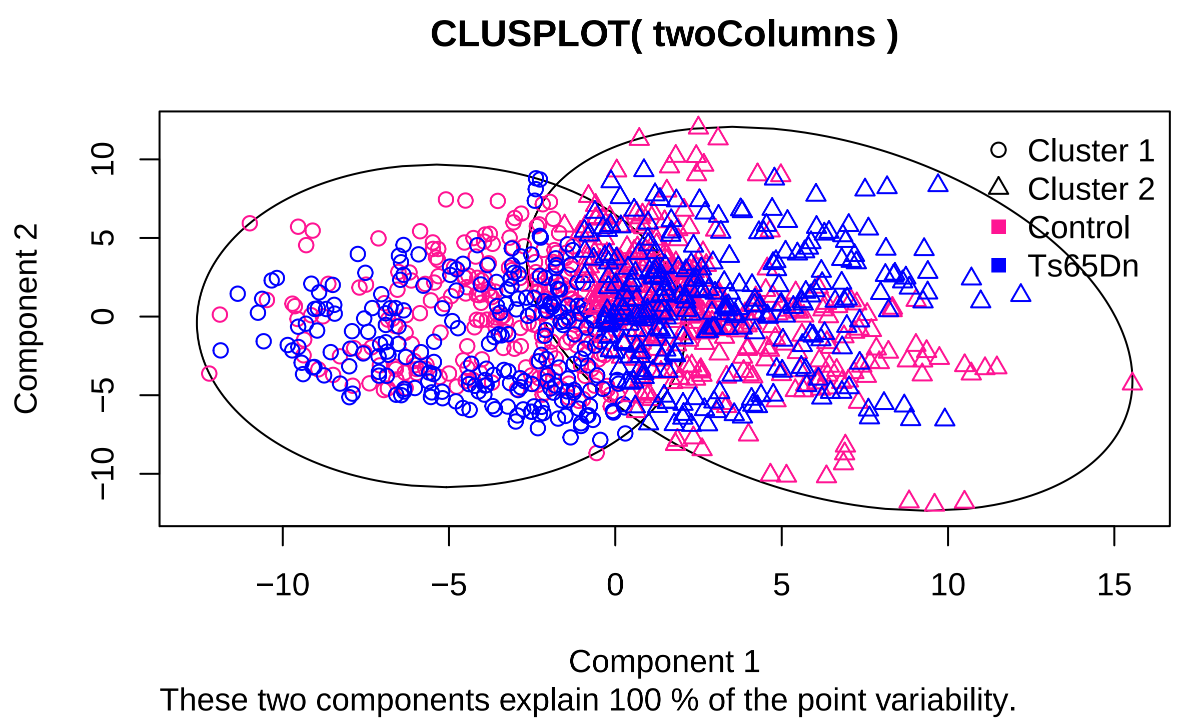
```
# # Individually plotted.
#
# tot.sse = c()
# k=5
# nstart8 = 40
# # jpeg(paste0('tot_sse_Plot_k', k, '_nstart', nstart8, '.jpg'))
# for (i in 1:100){
#     km.out <- kmeans(kdata.scale,k,nstart=nstart8) ##If this is not a straight line increase nstart.
#     tot.sse[i] <- km.out$tot.withinss
# }
#
# plot(tot.sse,type="l", main = paste("Plot of the tot.see for nstart of", nstart8, "and", k , "Cluster
# # dev.off()
# rm(i)
```

8.2 Appendix A.2. Additional rdPCA Plots and some Clustering Plots

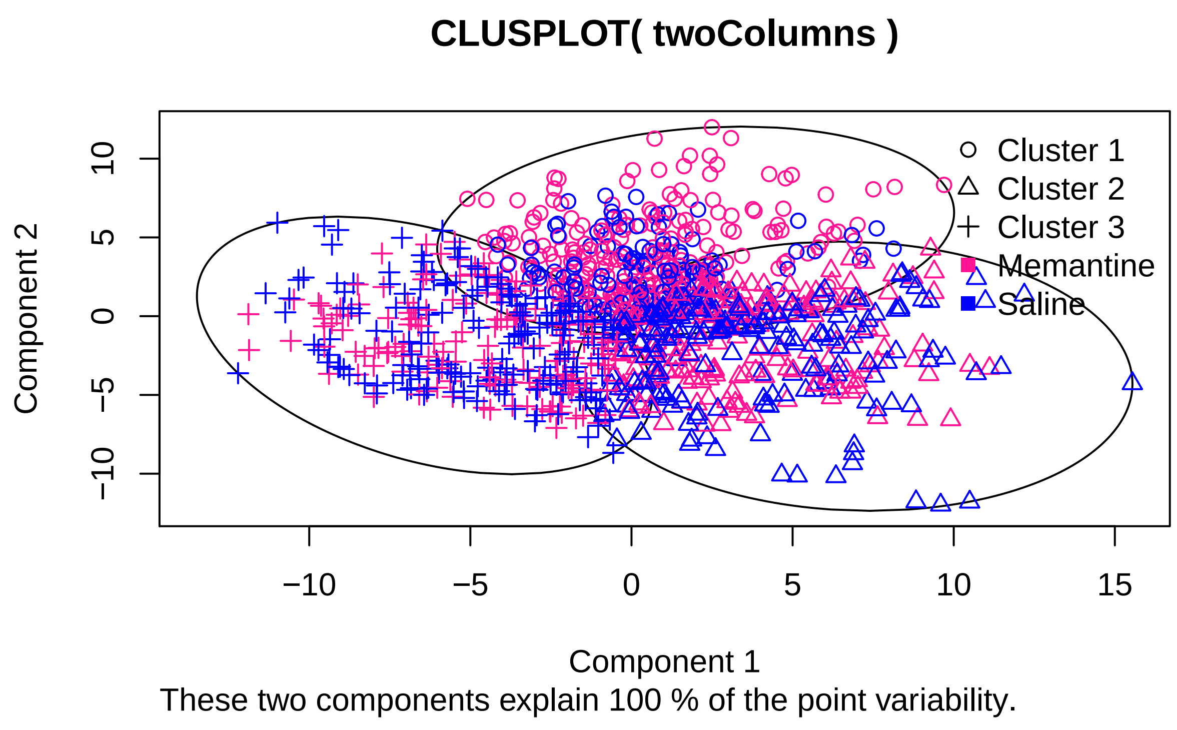
```
par(mfrow=c(1,1))


plot_PCA_kmeans(km2.mouse.out, labels_treatment)
```



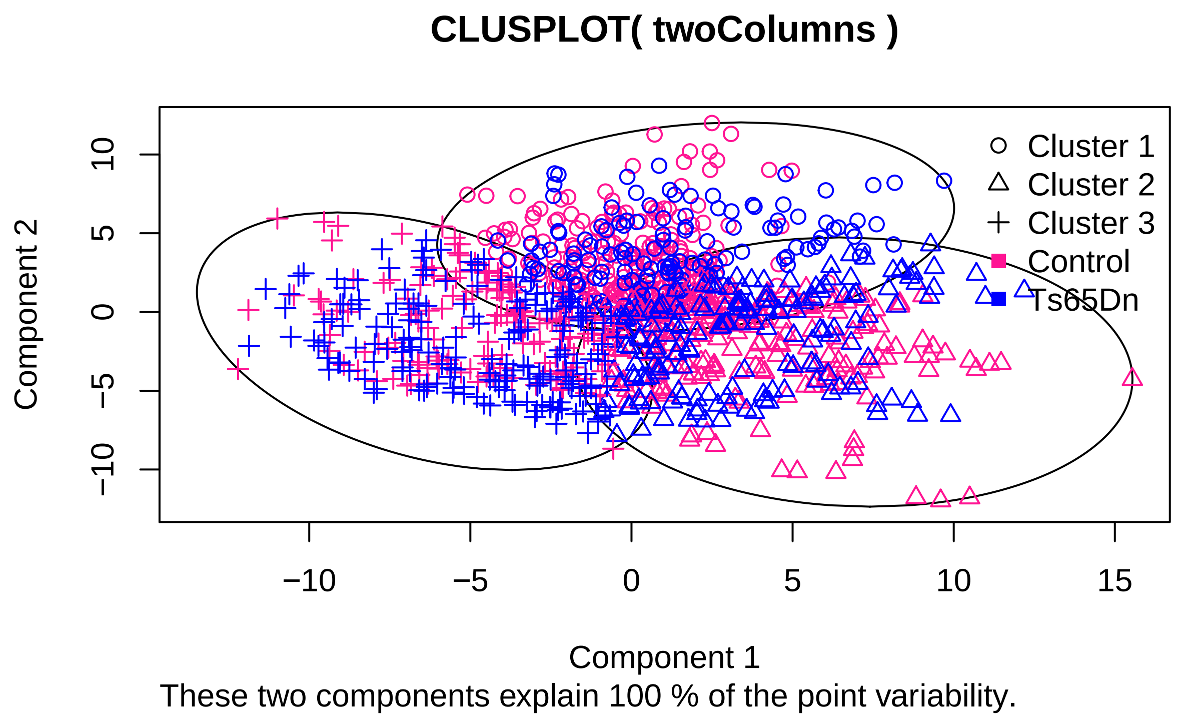
```
plot_PCA_kmeans(km2.mouse.out, labels_genotype)
```



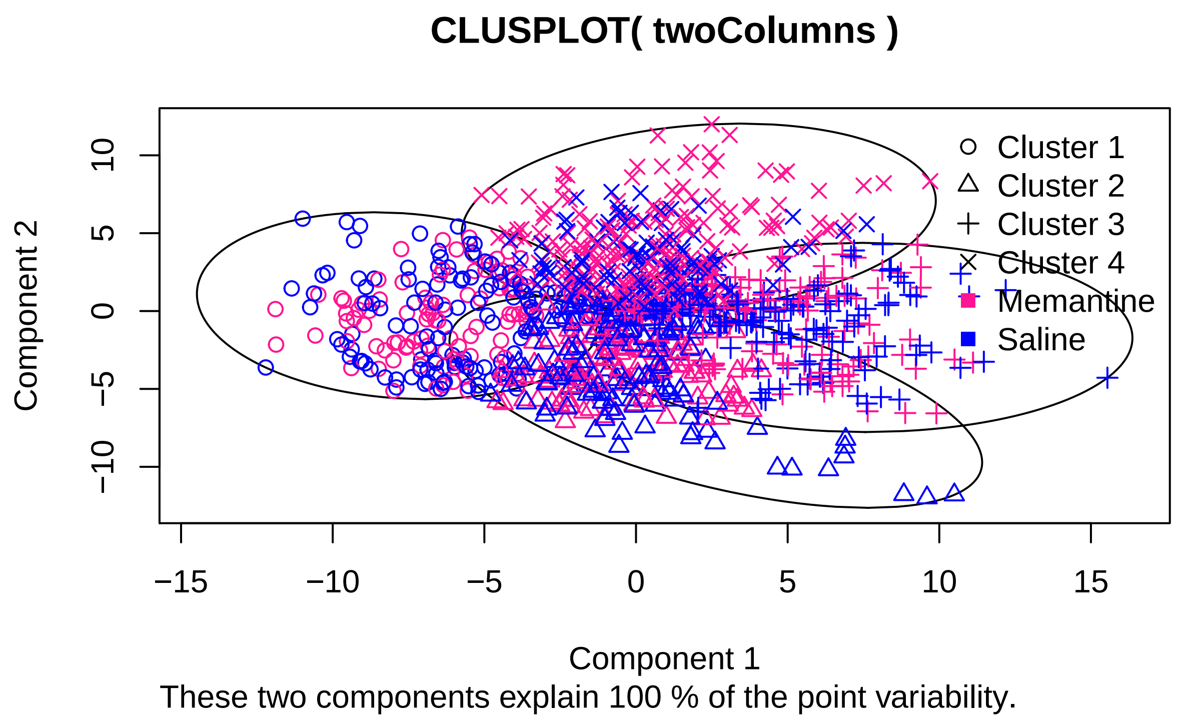
```
plot_PCA_kmeans(km3.mouse.out, labels_treatment)
```



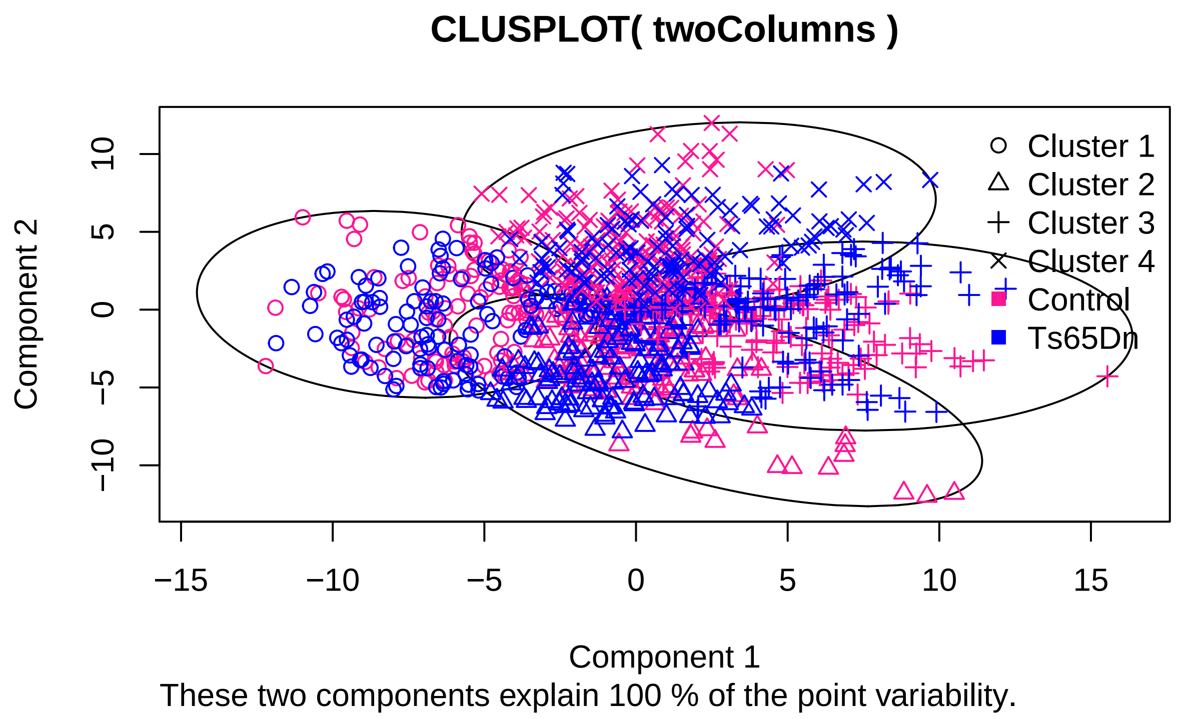
```
plot_PCA_kmeans(km3.mouse.out, labels_genotype)
```



```
plot_PCA_kmeans(km4.mouse.out, labels_treatment)
```



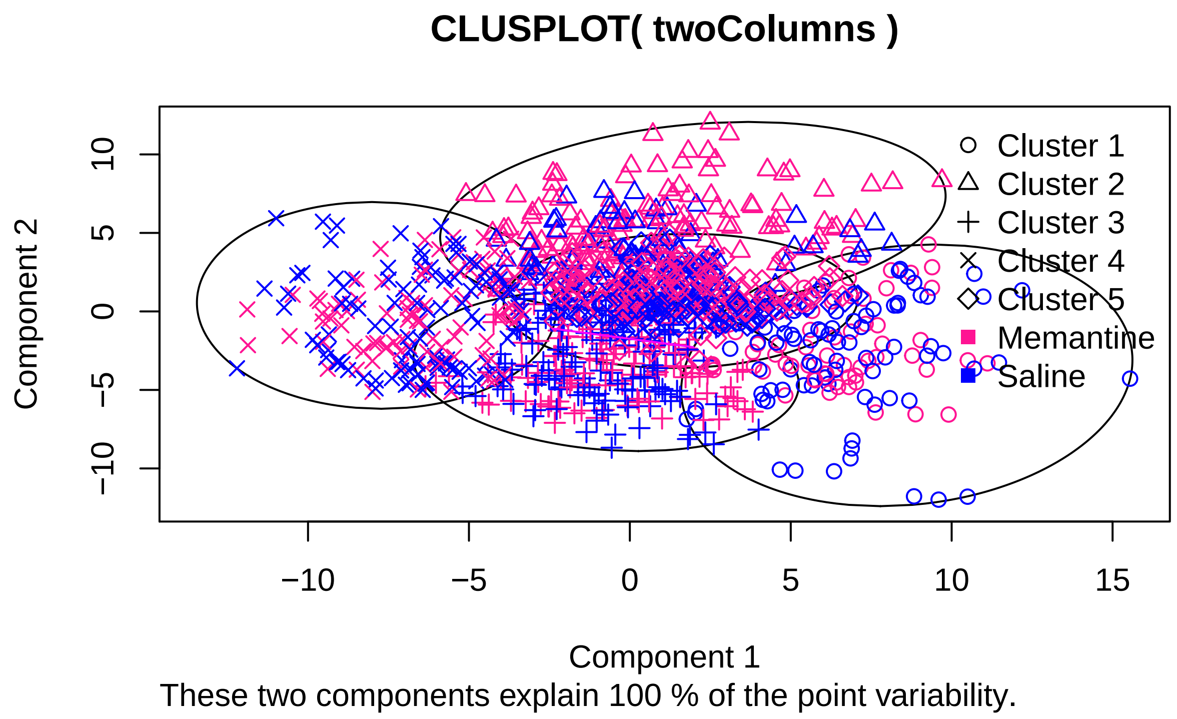
```
plot_PCA_kmeans(km4.mouse.out, labels_genotype)
```



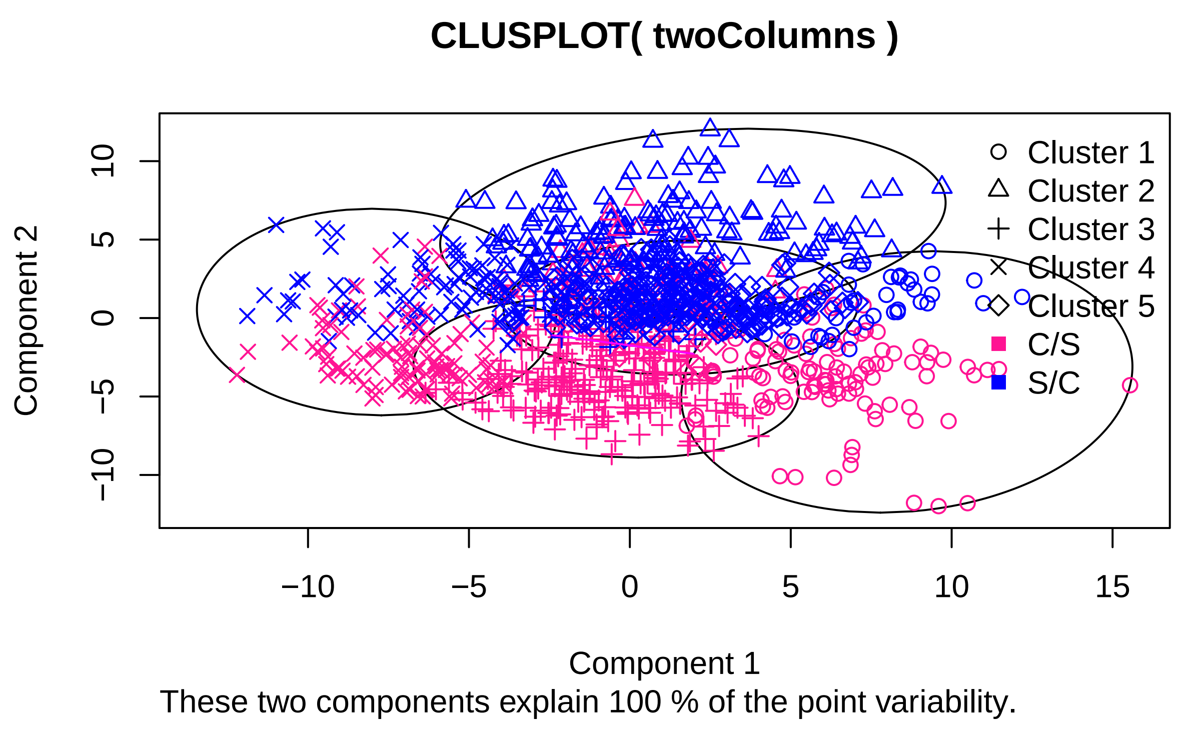
```
km5.mouse.out = kmeans(kdata.scale,5,nstart=40)
```

```
## Warning: did not converge in 10 iterations
```

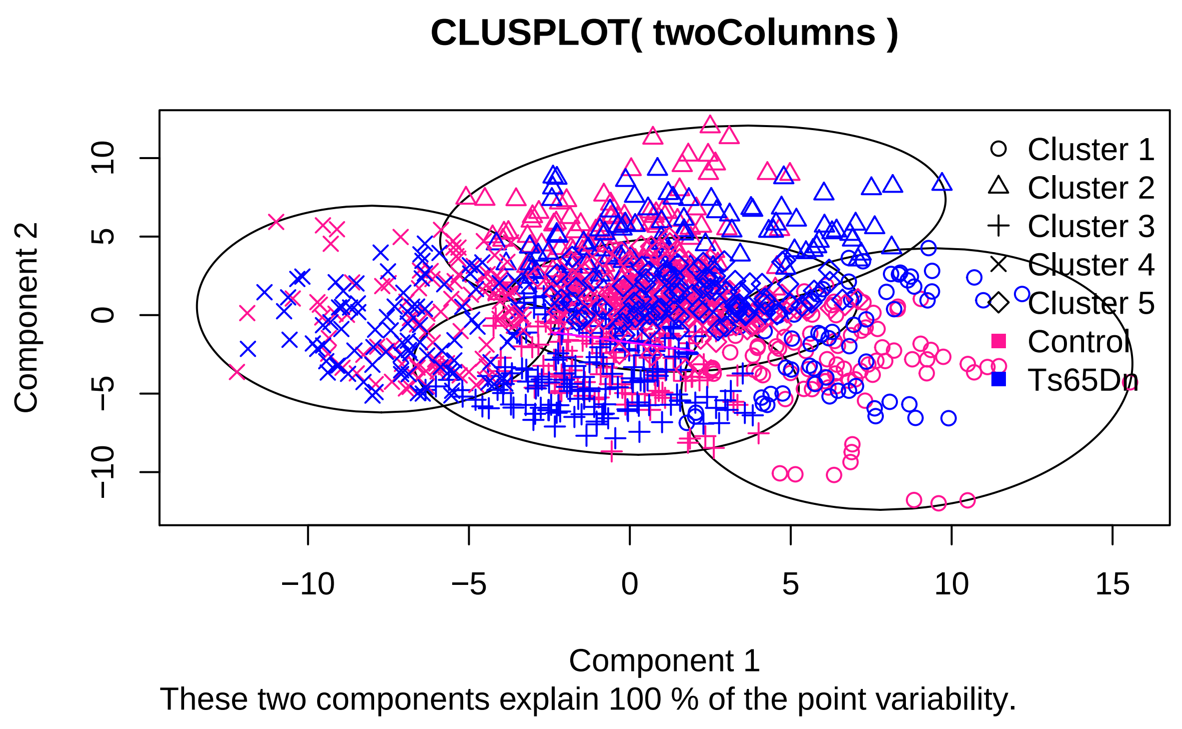
```
plot_PCA_kmeans(km5.mouse.out, labels_treatment)
```



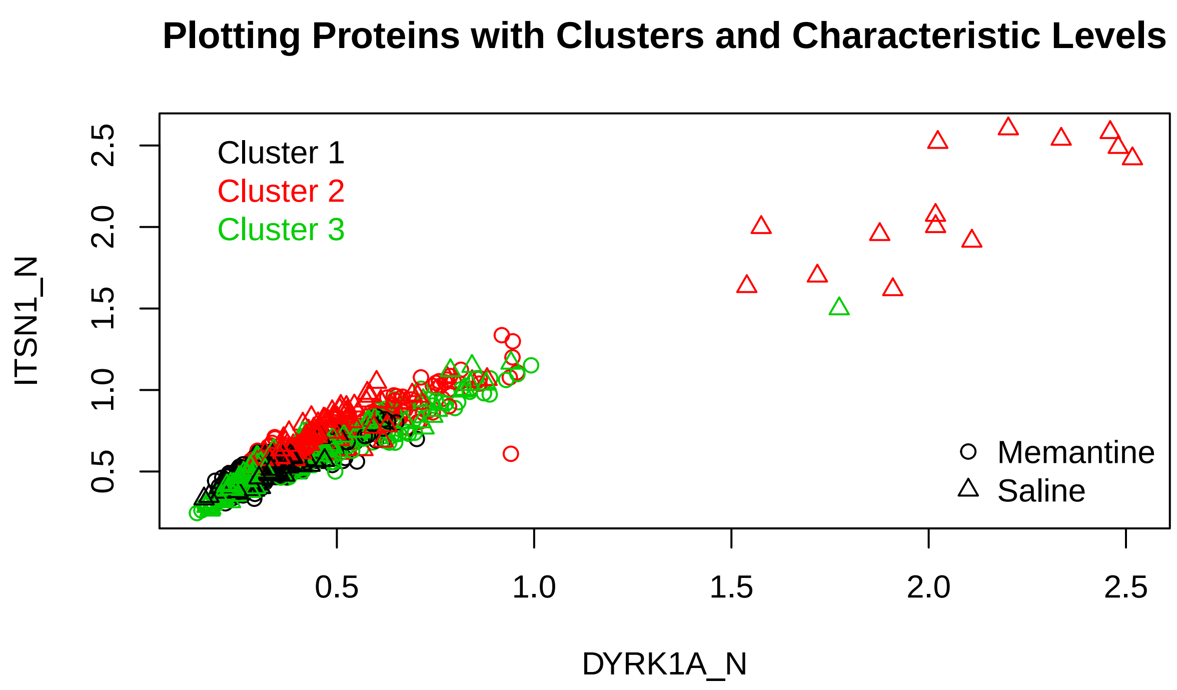
```
plot_PCA_kmeans(km5.mouse.out, labels_behavior)
```



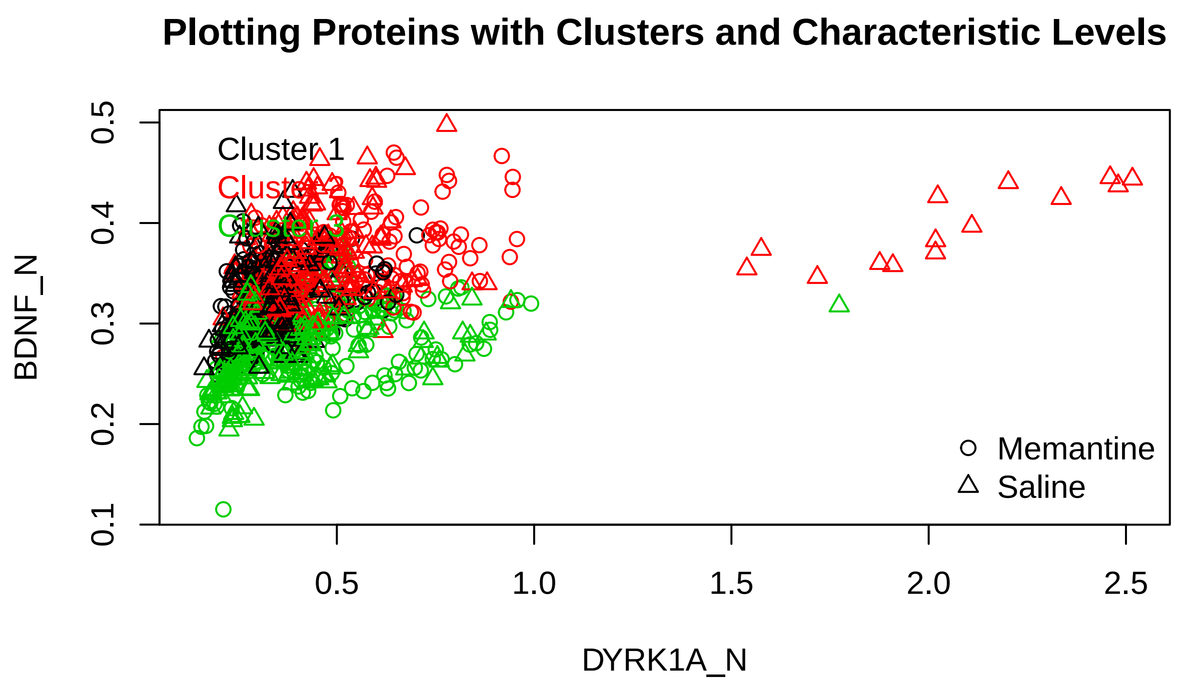
```
plot_PCA_kmeans(km5.mouse.out, labels_genotype)
```



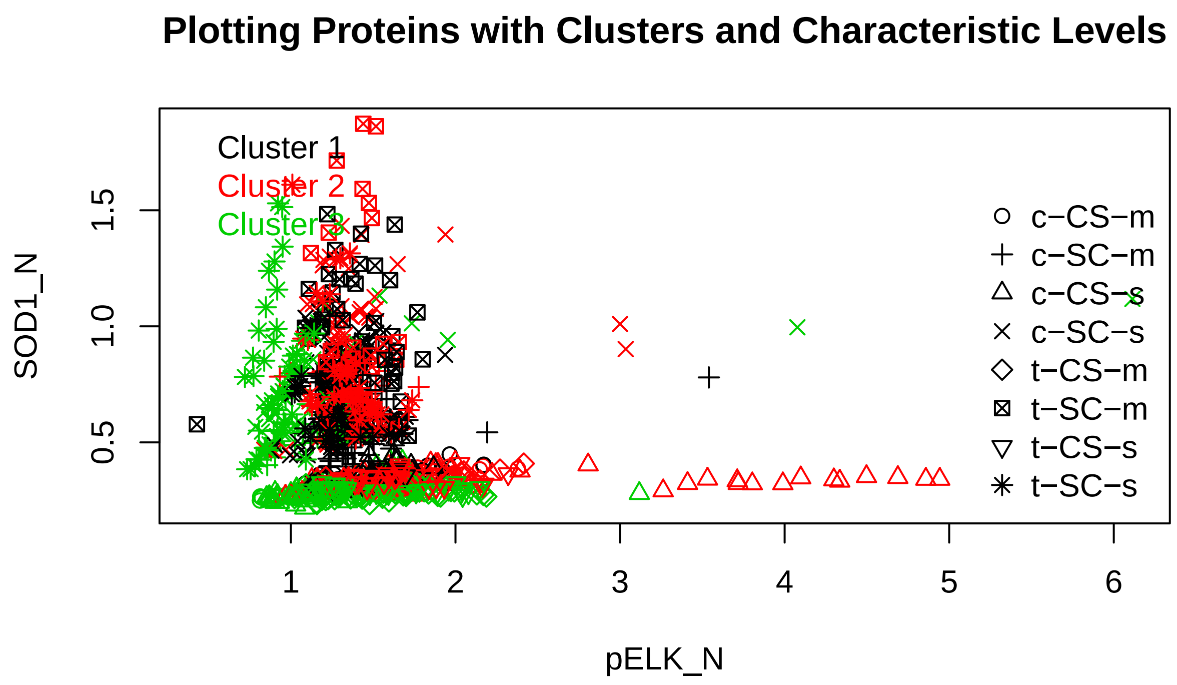
```r
plot_kmeans_variables(km3.mouse.out, 1, 2, labels_treatment)
```



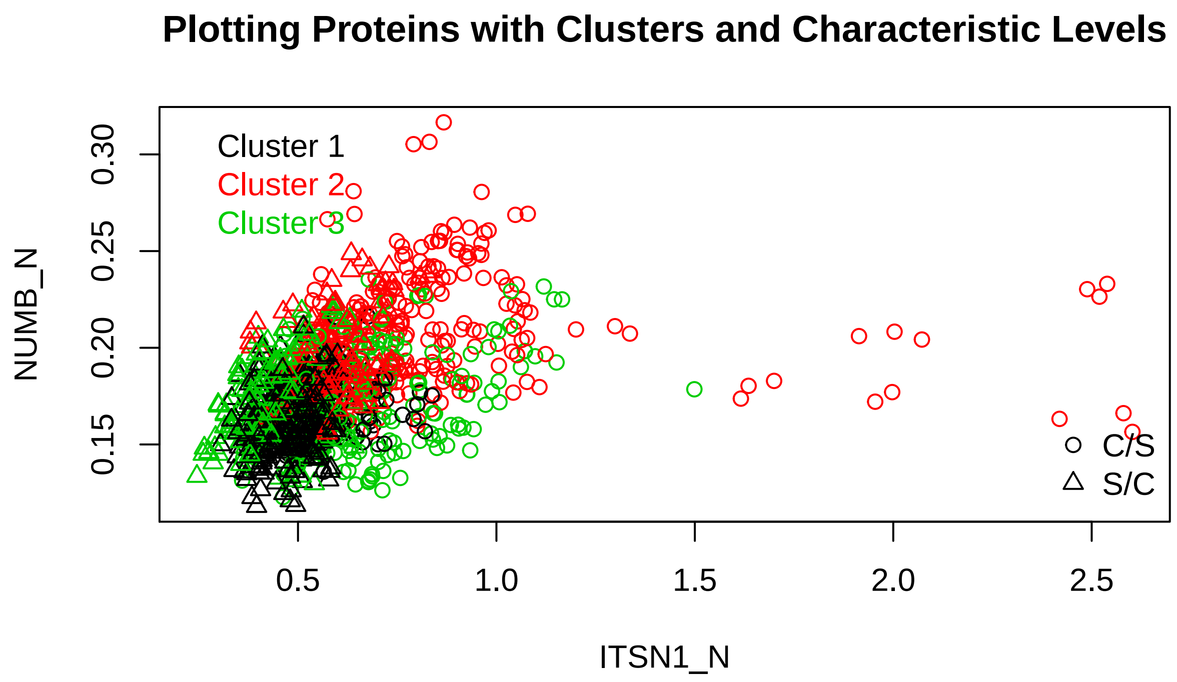
```r
plot_kmeans_variables(km3.mouse.out, 1, 3, labels_treatment)
```



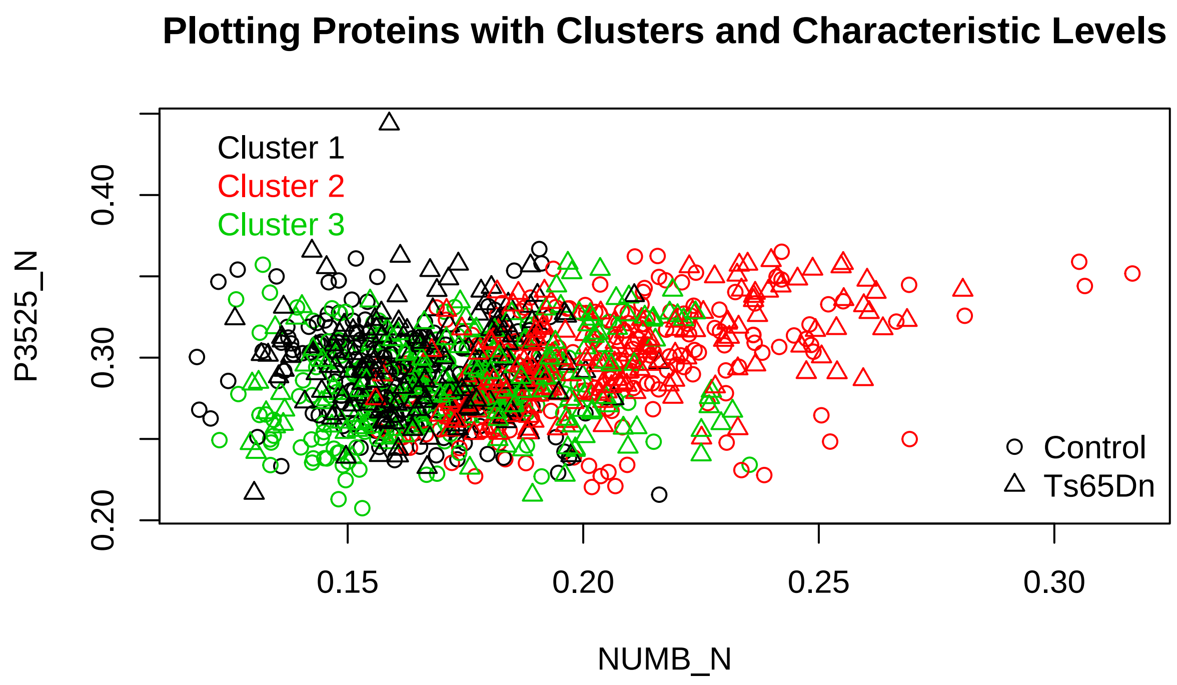
```
plot_kmeans_variables(km3.mouse.out, 10, 33, labels_class)
```



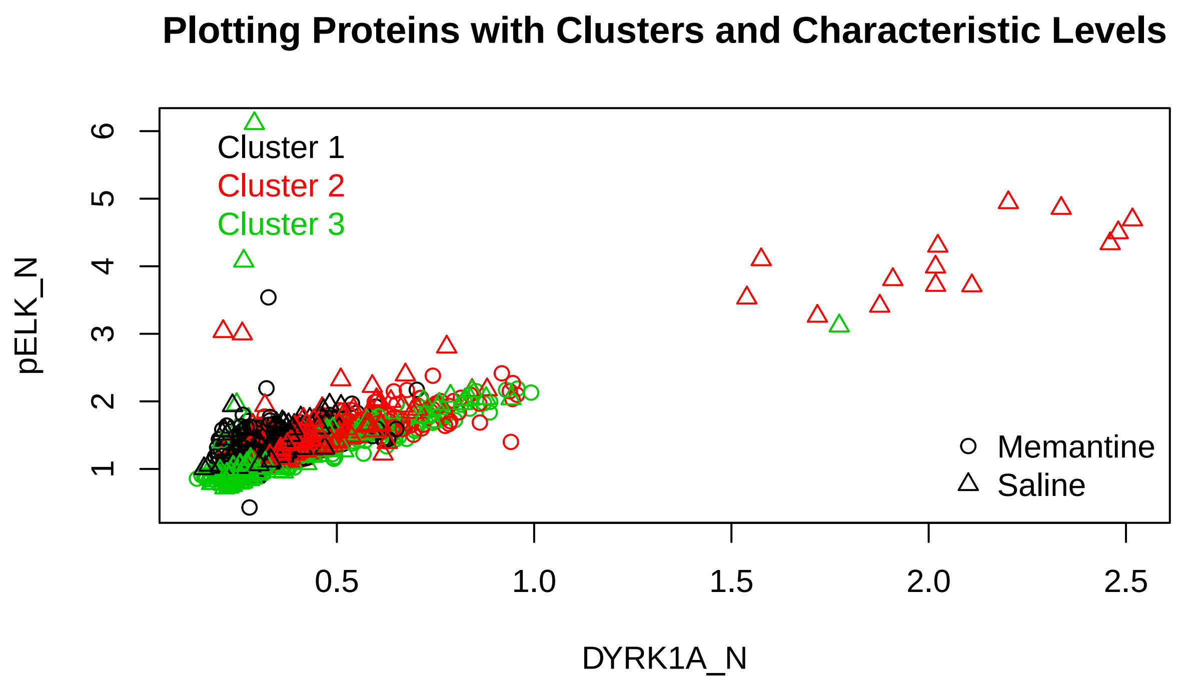
```
plot_kmeans_variables(km3.mouse.out, 2, 44, labels_behavior)
```



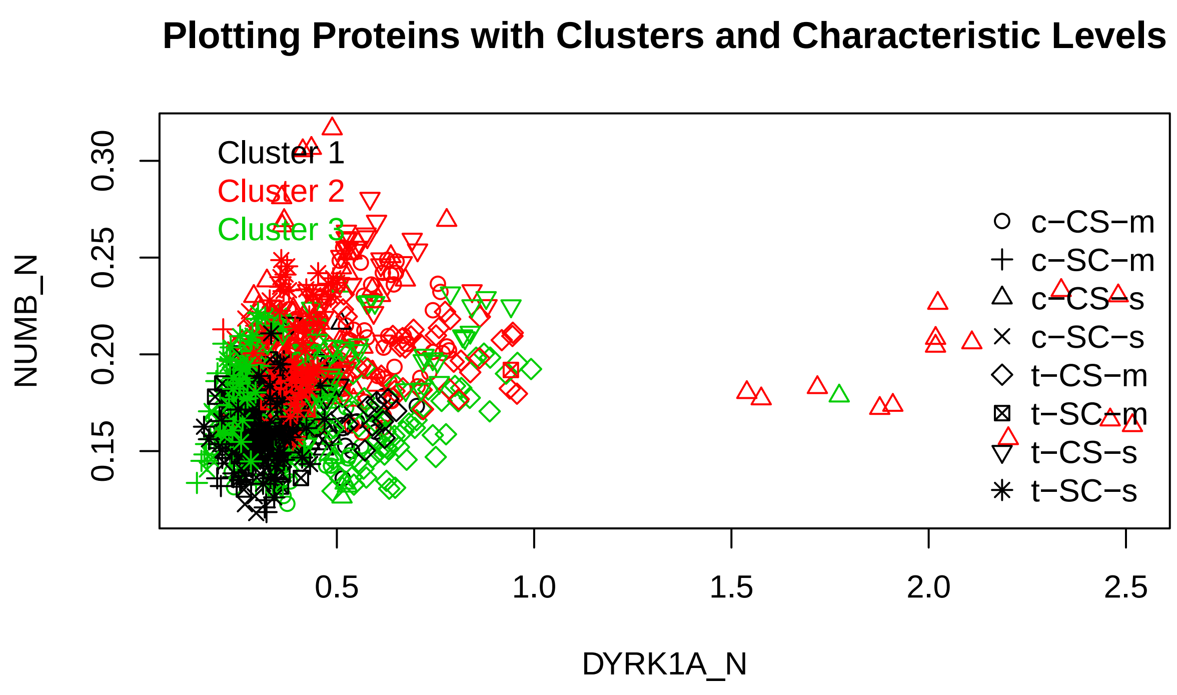
```r
plot_kmeans_variables(km3.mouse.out, 44, 62, labels_genotype)
```



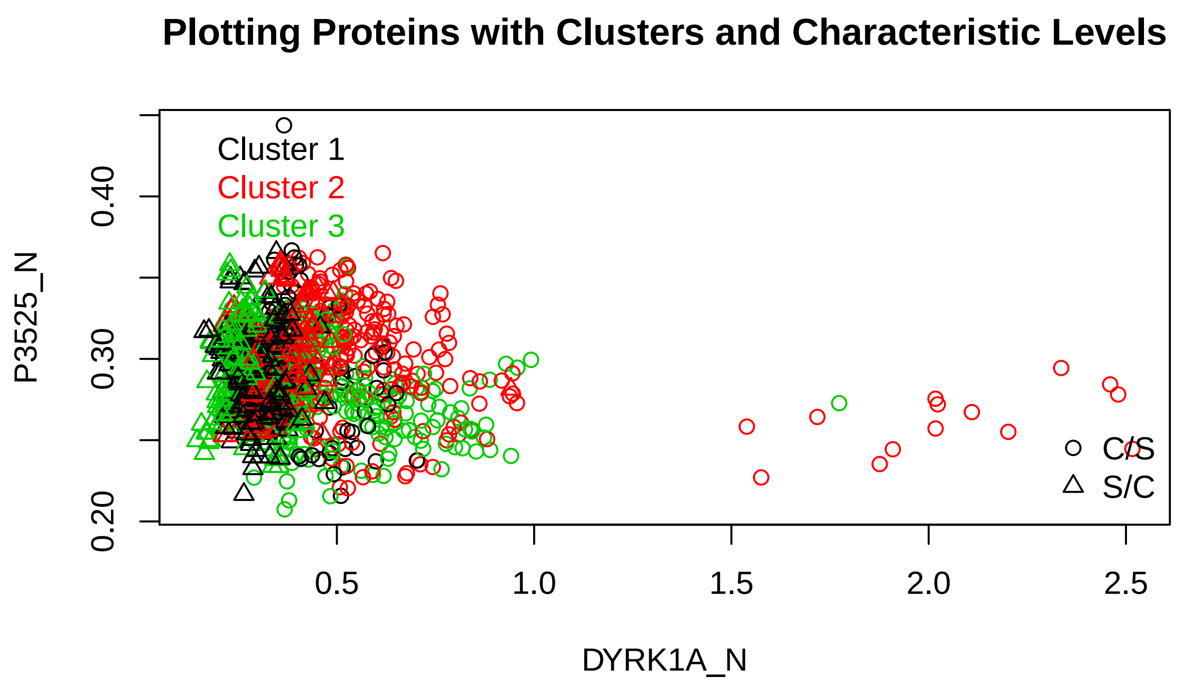
```r
plot_kmeans_variables(km3.mouse.out, 1, 10, labels_treatment)
```



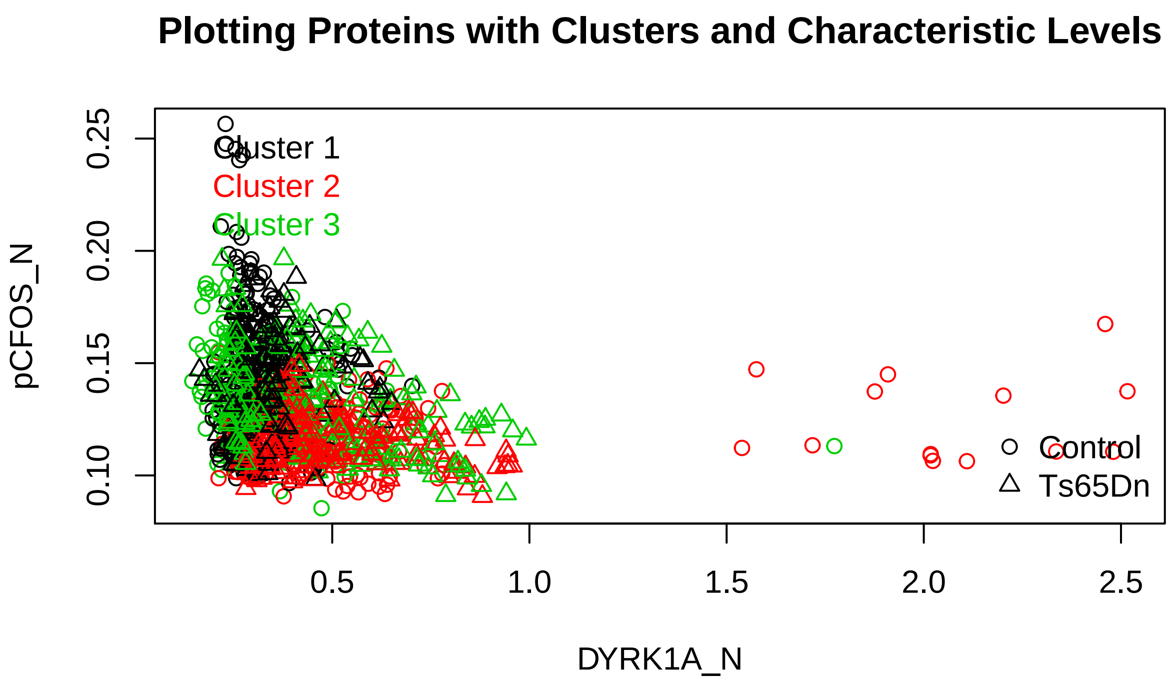
```
plot_kmeans_variables(km3.mouse.out, 1, 44, labels_class)
```



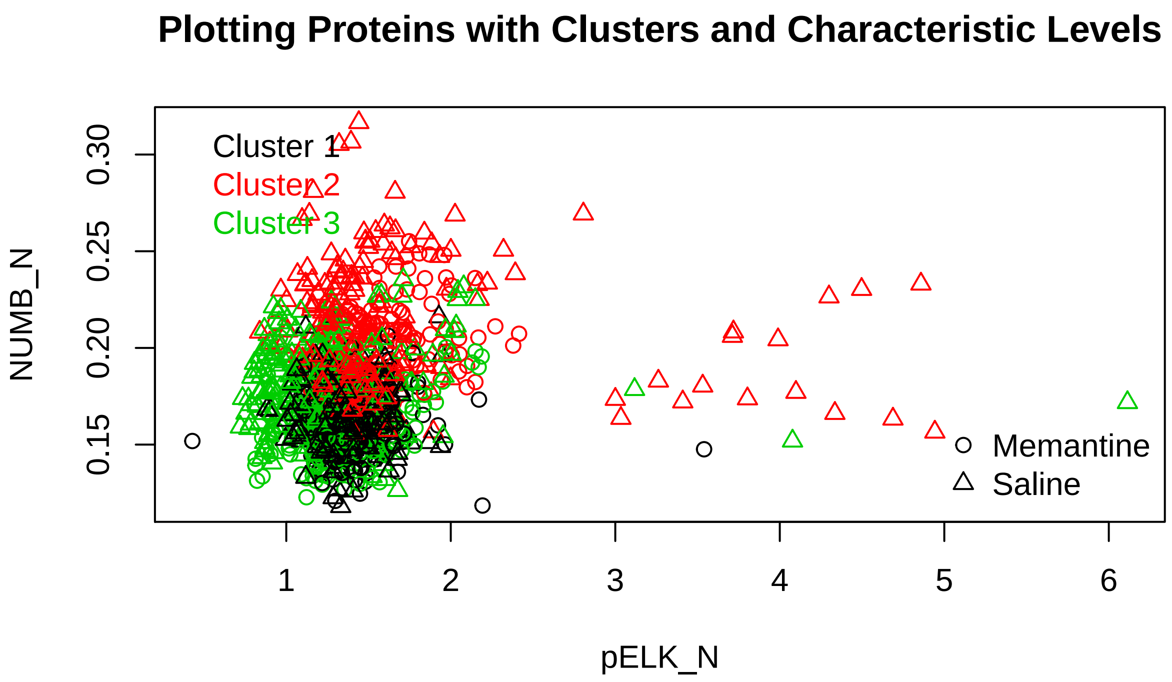
```
plot_kmeans_variables(km3.mouse.out, 1, 62, labels_behavior)
```



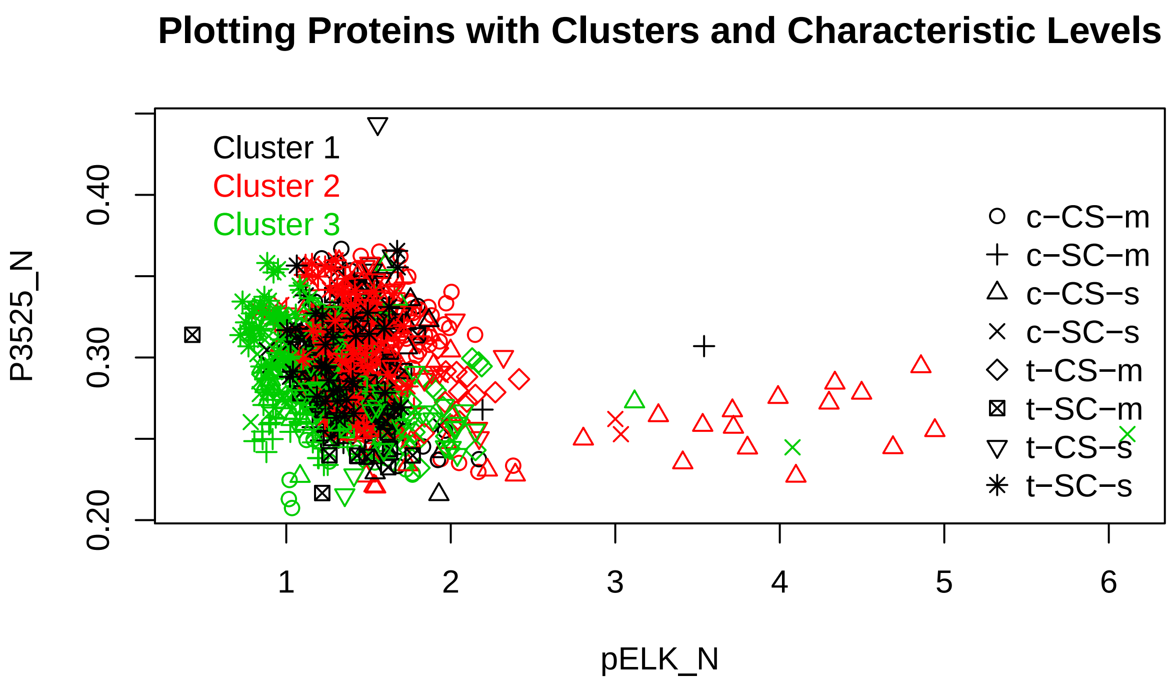
```r
plot_kmeans_variables(km3.mouse.out, 1, 72, labels_genotype)
```



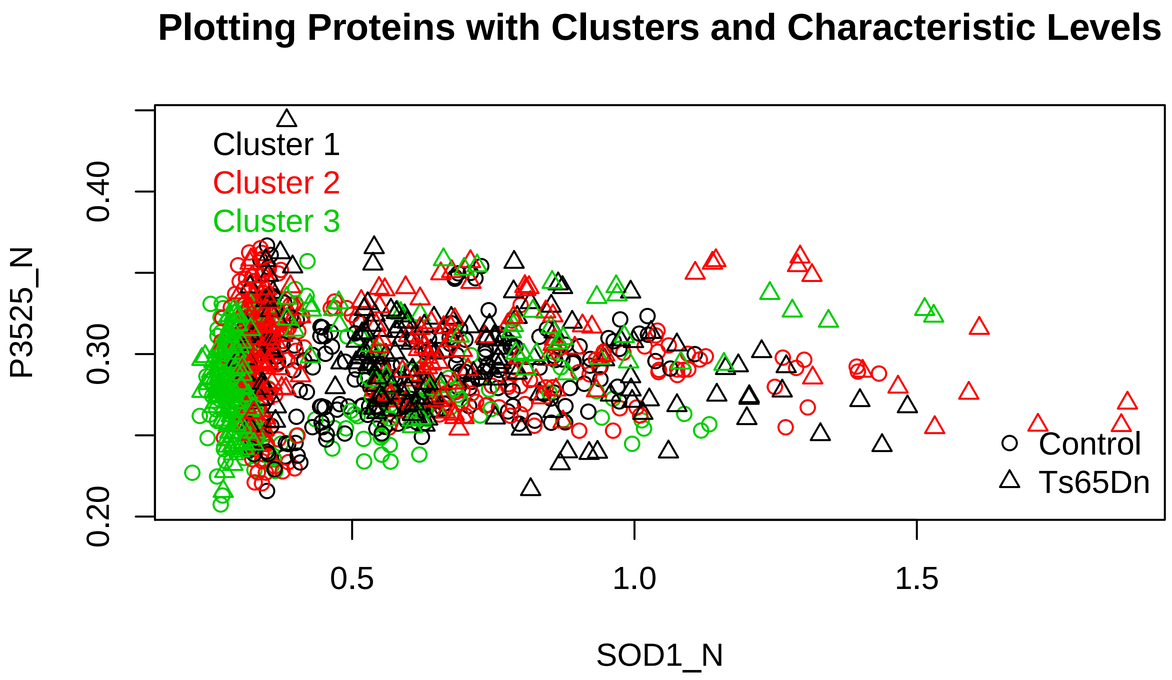
```r
plot_kmeans_variables(km3.mouse.out, 10, 44, labels_treatment)
```



```r
plot_kmeans_variables(km3.mouse.out, 10, 62, labels_class)
```



```r
plot_kmeans_variables(km3.mouse.out, 33, 62, labels_genotype)
```



8.3 Appendix B.1. Data Cleaning Code

```r
# Read in the data

data0 = read.csv("Data_Cortex_Nuclear.csv")
dim(data0)

# str(data0)
# unique(data0$num)
# Naming the variables.

varnames = names(data0)
varnames

# colnames(data0) = var_names
#
# var_names = c("age", "sex", "cp", "trestbps", "chol", "fbs","restecg",
#                "thalach", "exang", "oldpeak", "slope", "ca", "thal", "num")
#
# num_var =  c("age","trestbps", "chol",
#                "thalach", "oldpeak", "ca")
#
# factor_var = c( "sex", "cp", "fbs","restecg",
#                 "exang", "slope", "ca", "thal", "num")
#
# colnames(data0) = var_names


# Performing checks on the data

#summary(data0)


# Look at the summary data

# Looking at how many missing data there are indicated by "?".
data1 = data0

#which( data1 == "?", arr.ind = T)


# Looking at the missing data.
str(data0)
summary(data0)

md.pattern(data0)


# We will impute the missing data using multiple imputation


# We will impute the missing data using
# We will remove the variable MouseID before imputation.
data1 = select(data0,-MouseID)

dim(data0)
dim(data1)
set.seed(111)

#data2 = missForest(data1)

names(data2)

class(data2$ximp)

data2$OOBerror

data3 = data2$ximp

MouseID = data0$MouseID
data4 = cbind(MouseID, data3)
md.pattern(data4)
write.csv(data4, file = "rf_imputed_data.csv", row.names = F )
```

8.4 Appendix B.2. Helper Functions to create 2-D PCA visualization of data with Clustering.

```
#################### Helper functions


##################### Function 1
# We create a function that will take in as arguments:
# the kmeans clustering output, the variable column positions
# and the categorical variable to be used for plotting
# the clusters and the factors of the categorical variables
# on the scatter plot of the two indicated variables.

plot_kmeans_variables = function( km.out, x, y, labels_type){

  # Plot the points with clusters and labels
  plot(data_unscaled[, c(x,y)],col=km.out$cluster,
       pch = sort(as.numeric(unique(labels_type)))[labels_type],
       main = "Plotting Proteins with Clusters and Characteristic Levels")


  # Plot the legend
  unique_clusters = sort(unique(km.out$cluster))
  cluster_names = paste("Cluster", as.character(unique_clusters))
  label_names = as.character(unique(labels_type))

  legend("topleft",
         legend = cluster_names,
         text.col = unique_clusters,
         bty = "n")
```

```
    legend("bottomright",
         legend = label_names,
         text.col = "black",
         pch = as.numeric(unique(labels_type)),
         bty = "n")

}


###################### Function 2
# Creating a function that will plot the k-clustering
# with the PCA and labels.
# The function will take in arguments:
# the kmeans clustering output, and
# the categorical variable (labels_type),
# to be used for plotting the clusters as different labels/shapes
# such as triagles, circles, etc.
# The categorical variables levels will be the different colours
# These will be plotted on the first two pc's.

plot_PCA_kmeans = function( km.out, labels_type) {

  # Extract all the info from the km clustering output
  clusters <- km.out$cluster
  unique_clusters = sort(unique(clusters))

  # Perform PCA
  nba2d <- prcomp(kdata.scale, center=TRUE)
  twoColumns <- nba2d$x[,1:2]

  # Plot PCA with the labels_type i.e. behaviour, etc.
  clusplot(twoColumns, unique_clusters[clusters], col.p = c("deeppink", "blue")[labels_type], col.clus

  # Create variables for the legend.
  cluster_names = paste("Cluster", as.character(unique_clusters))
  label_names = as.character(unique(labels_type))
  black = rep("black", length(unique_clusters))


  # Plot the legend
  legend("topright",
         legend = c(cluster_names, label_names),
         col = c(black, "deeppink", "blue"),
         pch = c( unique_clusters, 15, 15),
         bty = "n",
         text.col = "black")

}


# We create the same functions as above, however,
# we slightly generalize them a bit more to be more
# flexible.
```

```r
##################### Function 1

plot_kmeans_variables1 = function( cluster_output, x, y, labels_type){

  # Plot the points with clusters and labels
  plot(data_unscaled[, c(x,y)],col=cluster_output,
       pch = sort(as.numeric(unique(labels_type)))[labels_type],
       main = "Plotting Proteins with Clusters and Characteristic Levels")

  # Plot the legend
  unique_clusters = sort(unique(cluster_output))
  cluster_names = paste("Cluster",
                        as.character(unique_clusters))
  label_names = as.character(unique(labels_type))


  legend("bottomright",
         legend = label_names,
         text.col = "black",
         pch = as.numeric(unique(labels_type)),
         bty = "n")

    legend("topleft",
         legend = cluster_names,
         text.col = unique_clusters,
         bty = "n")

}


###################### Function 2

plot_PCA_kmeans1 = function( cluster_output, labels_type) {

  # Extract all the info from the km clustering output
  clusters <- cluster_output
  unique_clusters = sort(unique(clusters))

  # Perform PCA
  nba2d <- prcomp(kdata.scale, center=TRUE)
  twoColumns <- nba2d$x[,1:2]

  # Plot PCA with the labels_type i.e. behaviour, etc.
  clusplot(twoColumns, unique_clusters[clusters], col.p = c("deeppink", "blue")[labels_type],
           col.clus = "black"  )

  # Create variables for the legend.
  cluster_names = paste("Cluster", as.character(unique_clusters))
  label_names = as.character(unique(labels_type))
  black = rep("black", length(unique_clusters))

  # Plot the legend
  legend("topright",
         legend = c(cluster_names, label_names),
         col = c(black, "deeppink", "blue"),
         pch = c( unique_clusters, 15, 15),
         bty = "n",
         text.col = "black")
}
```

8.5 Appendix B.3. Code for remaining analysis in the same order as the analysis was performed.

This presents the rest of the code in the same order that was used to do the analysis with an effort to keep the same headings for readability.

Data exploration

```
set.seed(2)

# The detailed data cleaning is in the appendix
data5 = read.csv("rf_imputed_data.csv")

# Create a data set with categorical variables.
categorical_data = data5[, 79:82]


# Creating labels for the categories to be used later.
labels_genotype = data5$Genotype
labels_treatment = data5$Treatment
labels_behavior = data5$Behavior
labels_class = data5$class

# Creating the data set with the numerical variables.
datta = data5[, -c(1, 79:82)] # Remove the categorical variables.
data_unscaled = datta
data_scaled = scale(datta) # Scaling the data.


# From the below summary we can see that all the categories are well balanced.
summary(categorical_data)
```

Below we graph the correlation plot to get a sense of how correlated our data is.

```
set.seed(2)
data =  data_unscaled
data = data_scaled
kdata.scale = data_scaled

M = cor(data)
corrplot(M, method="color", main = "Correlation Plot of the Numerical Variables")
```

Below we will produce the pairwise scatter plots of some of the proteins that appeared to have an interesting pattern.

```
par(mfrow=c(3,1))

```

```r
#pairs(data[, c(1,2,10,33)])
pairs(data[, c(1,10,62,71)])
pairs(data[, c(2,33,44)])

# Variables that appear to be interesting:
interesting_var = c(1,2,10,33,44,62,71)
#`r names(data_unscaled[,interesting_var])`
```

Heatmap

```r
y = t(as.matrix(data_unscaled))

heatmap.2(y,trace="none")
```

Silhouette: Comparing Clustering Methods and Number of Clusters

```r
# let's investigate the number of clusters for kmeans
sil_width1 <- c()

for(i in 1:4){
  km_out = kmeans(kdata.scale,i+1,nstart=40)
  si <- silhouette(km_out$cluster,dist(kdata.scale))
  ssi <- summary(si)

  # with pam fit, sil info is provided as part of the output
  # for other clustering methods we would have to extract it with the
  # silhouette function
  sil_width1[i] <- ssi$avg.width
}

# Plot sihouette width (higher is better)
plot(2:5, sil_width1,
     main = "Silhouette Width vs. Number of Clusters for K-Means Clustering",
     xlab = "Number of clusters",
     ylab = "Average Silhouette Width")
lines(2:5, sil_width1)
```

We now plot the hierarchical clustering average silhouette width against the number of clusters in the graph below with k going from 2 to 5. For hierarchical clustering we use complete linkage.

```r
# let's investigate the number of clusters for Hierarchical Clustering
sil_width2 <- c()

hc.complete <- hclust(dist(kdata.scale), method="complete")

for(i in 1:4){
  cut.tree.complete <- cutree(hc.complete,k=i+1)
  si <- silhouette(cut.tree.complete,dist(kdata.scale))
  ssi <- summary(si)

  # with pam fit, sil info is provided as part of the output
  # for other clustering methods we would have to extract it with the
  # silhouette function
  sil_width2[i] <- ssi$avg.width
}
```

```r
# Plot sihouette width (higher is better)
plot(2:5, sil_width2,
     main = "Silhouette Width vs. Number of Clusters for Hierarchical Clustering",
     xlab = "Number of clusters",
     ylab = "Average Silhouette Width")
lines(2:5, sil_width2)
```

We now plot the Partitioning Around Medoids clustering average silhouette width against the number of clusters in the graph below with k going from 2 to 5. For hierarchical clustering we use complete linkage.

```r
# let's investigate the number of clusters for PAM

y = kdata.scale # We use our scaled data.

sil_width3 <- c()
for(i in 1:4){
  pam_fit <- pam(y, k=i+1)
  # with pam fit, sil info is provided as part of the output
  # for other clustering methods we would have to extract it with the
  # silhouette function
  sil_width3[i] <- pam_fit$silinfo$avg.width
}

# Plot sihouette width (higher is better)

plot(2:5, sil_width3,
     main = "Silhouette Width vs. Number of Clusters for PAM",
     xlab = "Number of clusters",
     ylab = "Silhouette Width")
lines(2:5, sil_width3)
```

K-Means Clustering

Summary of what we will do for k-means clustering

Exploratory k-means clustering

We present the behaviour rdPCA plots below.

```r
km2.mouse.out = kmeans(kdata.scale,2,nstart=40)
plot_PCA_kmeans(km2.mouse.out, labels_behavior)
```

```r
km3.mouse.out = kmeans(kdata.scale,3,nstart=40)
plot_PCA_kmeans(km3.mouse.out, labels_behavior)
```

```r
km4.mouse.out = kmeans(kdata.scale,4,nstart=40)
plot_PCA_kmeans(km4.mouse.out, labels_behavior)
```

```r
plot_PCA_kmeans(km2.mouse.out, labels_treatment)
```

We will take a closer look at k = 2 clusters and k = 3 cluster.

For now we look more deeply into do k = 2 clusters first.

Now we will plot the two clusters and the classes against different variables.

```r
plot_kmeans_variables(km2.mouse.out, 1, 3, labels_treatment)
```

```
data_clus = data.frame(data_unscaled, cluster = km2.mouse.out$cluster)
boxplot(BDNF_N ~ cluster,data_clus,
        main = "Boxplot of BDNF_N Protein for the two Clusters",
        xlab = "Cluster",
        ylab = "BDNR_N")
```

We will now do the t test.

```
t.test(BDNF_N ~ cluster,data_clus)
```

We will now do a t-test for all the 77 proteins comparing their means in the two clusters.

```
numeric_var_names = colnames(data_unscaled)
test_data = lapply(data_clus[,numeric_var_names], function(x) t.test(x ~ data_clus$cluster))
pvalues_data = data.frame(p.value = sapply(test_data, getElement, name = "p.value"))
estimate_data = t(data.frame(sapply(test_data, getElement, name ="estimate")))
diff_in_mean = estimate_data[,2] - estimate_data[,1]

final_data = data.frame("Difference in mean of the two clusters" = diff_in_mean, pvalues_data)

protein_w_sig_pvalues = subset(final_data  , p.value < 0.05/77 )

sig_proteins = rownames(protein_w_sig_pvalues)
```

Here below is the table with the mean difference and the p-values.

```
kable(protein_w_sig_pvalues, caption = "Proteins with Statisitically Significant
      mean difference between the two clusters."  )
```

We will now describe the two clusters in the table below for all the variables of the data.

```
# Helper Function based on CreateTableOne() function to create tables in
# Rmarkdown/Rsweave using kable.
KreateTableOne = function(x, ..., printSMD = TRUE){
  t1 = tableone::CreateTableOne(data=x, ...)
  t2 = print(t1, quote=TRUE, ...)
  rownames(t2) = gsub(pattern='\\"', replacement='', rownames(t2))
  colnames(t2) = gsub(pattern='\\"', replacement='', colnames(t2))
  return(t2)
}
```

```
table1_data = data.frame(data5[,-1], cluster = km2.mouse.out$cluster)
vars_names = c(names(categorical_data), names(data_unscaled))
fvar = names(categorical_data)
table1 = KreateTableOne(x = table1_data, vars = vars_names, strata= "cluster" )

#table1 = print(table1, showAllLevels)
```

```
kable(table1, caption = "Describing the two clusters.")
```

We looked at several more plots and the found the following two very interesting.

```
plot_kmeans_variables(km2.mouse.out, 1, 3, labels_behavior)
```

```
plot_kmeans_variables(km2.mouse.out, 2, 44, labels_behavior)
```

We investigate this a bit further and perform a t-test to verify our finding.

```r
t.test(ITSN1_N ~ Behavior, data5)
```

We now do a t-test for the protein BDNF_N and compare the mean difference between the behavior groups of C/S and S/C.

```r
t.test(BDNF_N ~ Behavior, data5)
```

Now we do k-means clustering for 3 clusters.

```r
plot_kmeans_variables(km3.mouse.out, 1, 3, labels_behavior)
```

```r
t.test(DYRK1A_N ~ Behavior, data5)
```

We now plot the clustering with the proteins pELK_N and P3525_N.

```r
plot_kmeans_variables(km3.mouse.out, 10, 72, labels_behavior)
```

We present the concordance tables of the k = 3 k-means clustering.

```r
## The concordance matrix for k = 3.
# Make a cross-tabulation table that shows the concordance.
table(km3.mouse.out$cluster,labels_genotype); table(km3.mouse.out$cluster,labels_behavior); table(km3.mo
```

Hierarchical clustering

We will now perform hierarchical clustering.

First, we will do clustering using the three linkages: complete, single, and average, and plot the clustering.

```r
####################################################
####################################################
##### Hierarchical clustering ############

## Do not need the distances not the data, to run the whole thing.

## Creates a distance matrix.
hc.complete <- hclust(dist(kdata.scale), method="complete")
hc.average <- hclust(dist(kdata.scale), method="average")
hc.single <- hclust(dist(kdata.scale), method="single")

par(mfrow=c(1,3))

plot(hc.complete,main="Complete Linkage", xlab="", sub="",
       cex =.9) ## WHat is cex?
plot(hc.average , main =" Average Linkage ", xlab="", sub ="",
       cex =.9)
plot(hc.single , main=" Single Linkage ", xlab="", sub ="",
       cex =.9)
```

Further, for k = 2, we will plot the resulting clusters for the proteins BRAF_N and ERBB4_N below.

```r
par(mfrow=c(1,3))

cut.tree.com <- cutree(hc.complete,k=2)
cut.tree.av <- cutree(hc.average,k=2)
cut.tree.sin <- cutree(hc.single,k=2)

plot(data_scaled[, c(21, 55)],
     col=cut.tree.com,main="complete")
```

```
plot(data_scaled[, c(21, 55)],
     col=cut.tree.av,main="average")

plot(data_scaled[, c(21, 55)],
     col=cut.tree.sin,main="single")
```

Now, we will do a similar visualization of the clustering using the PCA dimension reduction trick.

```
# How about k=2 ?

cut.tree.com <- cutree(hc.complete,k=2)
cut.tree.av <- cutree(hc.average,k=2)
cut.tree.sin <- cutree(hc.single,k=2)

plot_kmeans_variables1(cut.tree.com, 23, 24, labels_behavior)
```

```
plot_PCA_kmeans1(cut.tree.com, labels_behavior)
```

We present the concordance table below, after correcting for this inversion.

```
cut.tree.com = cut.tree.com - 1
cut.tree.com[ which(cut.tree.com == 0)] = 2

table(cut.tree.com, labels_behavior )
table(cut.tree.com, labels_treatment)
table( cut.tree.com, labels_genotype)
table( cut.tree.com, labels_class)
```

We compare the results of the k=2 hierarchical clustering with the k-means clustering.

```
km2.clusters =km2.mouse.out$cluster
cut.tree.compl <- cut.tree.com
tree.complete.clusters = cut.tree.compl

d = table(tree.complete.clusters, km2.clusters)

kable(xtable(d), row.names = TRUE,
      col.names = c("Kmeans Cluster 1", "Kmeans Cluster 2"),
      caption = "Comparing clustering of hierarchical clustering (rows) with k-means clustering for k =
```

We now do some clustering for k = 3.

```
# How about k=3?

cut.tree.com <- cutree(hc.complete,k=3)

par(mfrow=c(1,3))


plot(data_scaled[, c(4, 5)],
     col=cut.tree.com,main="complete")

plot(data_scaled[, c(1, 77)],
     col=cut.tree.com,main="complete")

plot(data_scaled[, c(31, 32)],
```

```r
     col=cut.tree.com,main="complete")
```

```r
par(mfrow=c(1,1))

plot_PCA_kmeans1(cut.tree.com, labels_behavior)
```

```r
table(cut.tree.com, labels_behavior )
table(cut.tree.com, labels_treatment)
table( cut.tree.com, labels_genotype)
table( cut.tree.com, labels_class)
```

```r
cut.tree.com <- cutree(hc.complete,k=8)
table( cut.tree.com, labels_class)
```

```r
cut.tree.com <- cutree(hc.complete,k=12)
table( cut.tree.com, labels_class)
```

PCA and Biplots.

```r
pr.out <- prcomp(data_unscaled, scale=TRUE) #We have scaled the data
pr.var <- pr.out$sdev^2
pve <- pr.var/sum(pr.var)

plot(pve, xlab="Principal Component", ylab="Proportion of
     Variance Explained ", type="b")

# there is a drop after about 9 pc's
```

```r
plot(cumsum(pve), xlab="Principal Component", ylab="
     Cumulative Proportion of Variance Explained ",
     ylim=c(0,1), type="b")
# not a drastic change in terms of cummulative variance explained

# summary contains most of this info
```

```r
vars <- apply(pr.out$x, 2, var)
props <- vars / sum(vars)
cumsum(props)[1:9]
```

```r
biplot(pr.out, scale=0 )
```