Exercise 5 Trashaj Alberto 1075402

alberto.trashaj

October 2023

1 Point 1

In this point was to cluster the glass splinters into different types of glass.
The dataset, after being loaded is analyzed: there are 9 variables and 214 observations, in particular there is one variable called RI that represent the refractive index, all the others variables are chemical components of the glass.
Let's check the correlation structure of the dataset:

```{r}
heatmap(cor(glass))
```

As we can see, the variable Ri and Ca are strongly correlated: this means that the information carried by Ca is sufficient and we can reduce the dimensionality of the dataset by disregarding the Ri variable.
Now, all the variables explain a part of the composition of the material, so there is no need to standardize the dataset.

The multi-collinearity between two variables can lead problems when we compute the Gaussian mixture model, so another reason to get rid of the RI variable is to avoid problems in computing the

```
reduced <- glass[,-1]
```

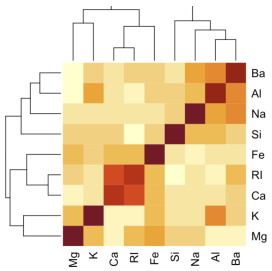


Figure 1: Correlation structure

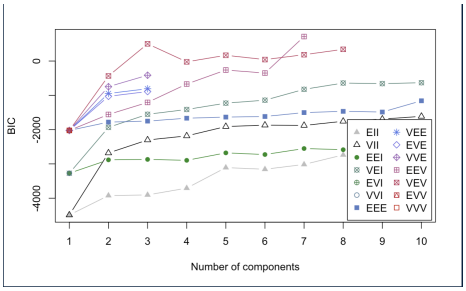


Figure 2: BIC-GMM

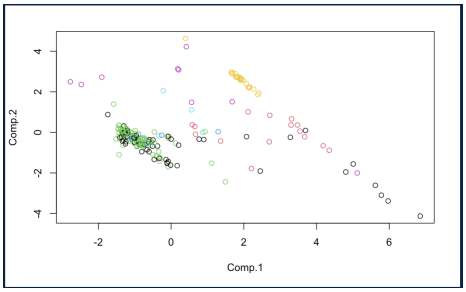


Figure 3: PCA with GMM

```
mreduced <- as.matrix(reduced)
gmmreduced<- (Mclust(mreduced, G = 1:10))
summary(gmmreduced)

#Mclust EEV (ellipsoidal, equal volume and shape) model with 7 components:

#Clustering table:
# 1  2  3  4  5  6  7
#60 15 96  7  5  9 22
```

The Mclust function gives as result that the EEV model with 7 components maximise the BIC.

```
plot(gmmreduced)
```

In this plot we can see that the EEV model takes maximum values with 7 clusters

To produce a visualization for this clustering I choose to take the PCA approach

```
pca_reduced <- princomp(reduced)
plot(pca_reduced$scores[,1:2], col = gmmreduced$classification)
```

The plot although presents different overlapping regions.

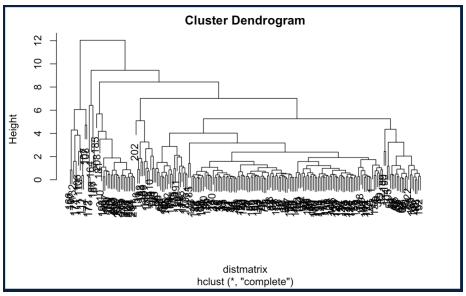


Figure 4: Complete method

Let's see now a different type of clustering, the hierarchical ones: single, complete, average linkage.
For each cluster I computed the opt.number of clusters with the ASW method.

```
method <- c("single", "average", "complete")
for (i in method){
  result_cluster <- hclust(distmatrix ,method = i)
  plot(result_cluster)
}
```

Here I show the dendrogram for the complete method:
Now we compute the ASW as we did during the course

```
asw_average <- NA
average_clusk <- list()
average_sil <- list()

# Look at K between 2 and 30:
for (k in 2:30){
# Hierarchical-average clustering:
  average_clusk[[k]] <- cutree(hclust(distmatrix, method = "average"), k = k)

# Computation of silhouettes:
  average_sil[[k]] <- silhouette(average_clusk[[k]],dist=distmatrix)

# ASW needs to be extracted:
  asw_average[k] <- summary(average_sil[[k]])$avg.width

}
# Plot the ASW-values against K:
plot(1:30,asw_average,type="l",
        xlab="Number of opt. clusters with average method",ylab="ASW")
plot(average_sil[[which.max(asw_average)]])
```

The optimal number of clusters with the average method is 2, same result we got with the complete method. Although, also when the number of clusters

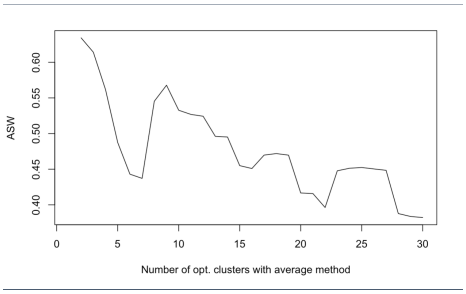


Figure 5: ASW-average

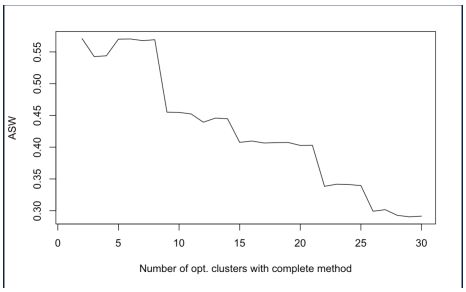


Figure 6: ASW complete

is between 6 and 9 we see a high value of the ASW with the complete method as we can see in the plot below.

With this for loop I plotted a visualization using the MDS

```
for (i in 3:9){
  complete_cut <- cutree(hclust(distmatrix, method = "complete"), k = i)
  plot(mds(distmatrix)$conf, col = complete_cut,
       pch = clusym[complete_cut])
}
```

This visualization looks better than the GMM one, we can see no overlapping regions and the cluster seems to be well separated.

I would say that the clustering with hierarchical methods (average and complete show almost the same results) works better, because the cluster seems to be better separated.

As mentioned before, the GMM may does not work very well given the correlation structure of the dataset.

Let's apply now the DBSCAN algorithm:

```
library(dbscan)
```

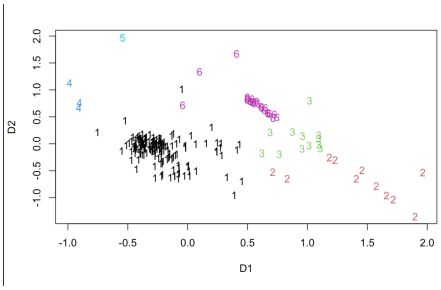


Figure 7: Complete method 7 cluster MDS

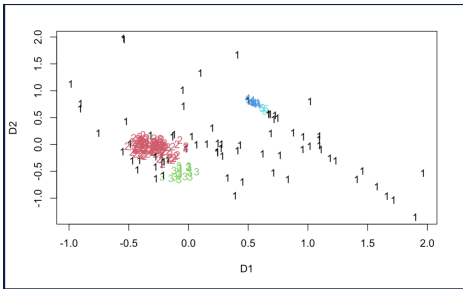


Figure 8: DBSCAN with eps=0.5, minPts = 5

```
dbscan_res <- dbscan(mreduced, eps = 0.5, minPts = 5)
clusters <- dbscan_res$cluster
plot(mds(distmatrix)$conf, col = 1+ dbscan_res$cluster,
        pch = clusym[1+ dbscan_res$cluster])
```

Even this visualization present some overlapping regions and the cluster labelled as "1" seems too spread out in the mds space.
To make some experiment on the tuning parameters let's iterate over two for loop different parameters of the distance and the minimum number of points.

```
clusym <- c(16, 17, 18, 19, 20)  # Symbol for each cluster

dbscan_results <- list()
plots <- list()

for (i in seq(0.1, 1, by = 0.2)) {
  for (j in seq(5, 9, by = 1)) {

    dbscan_result <- dbscan(mreduced, eps = i, minPts = j)
    dbscan_results[[paste0("i", i, "_j", j)]] <- dbscan_result

    plot_data <- mds(distmatrix)$conf
```

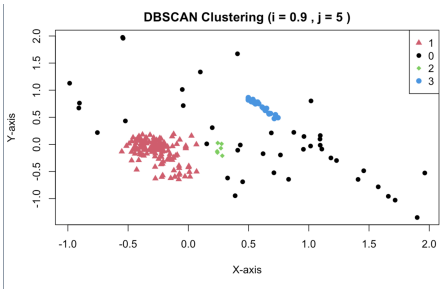


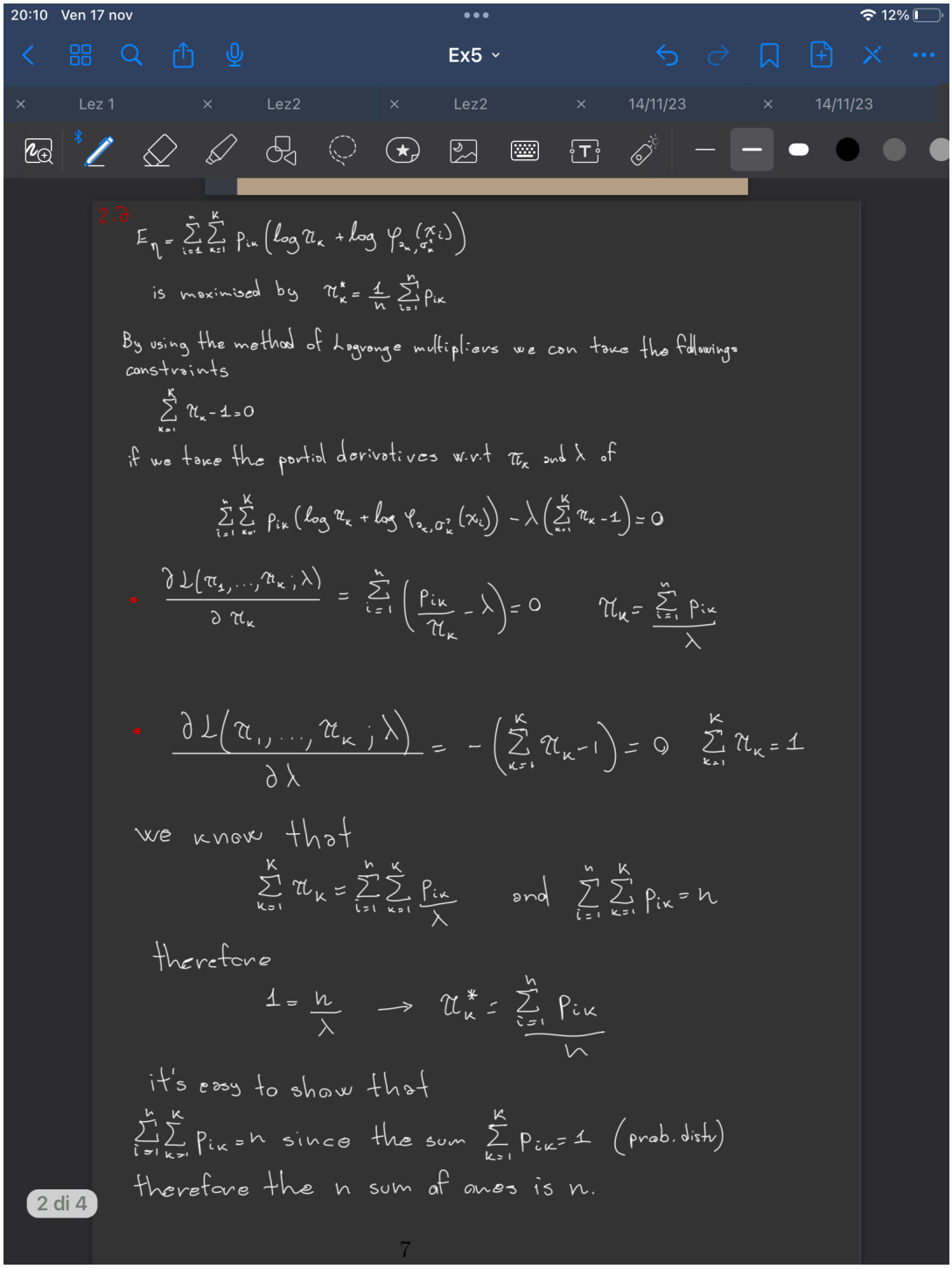
Figure 9: DBSCAN with eps = 0.9 and minPts = 5

```
    plot_colors <- 1 + dbscan_result$cluster
    plot_symbols <- clusym[plot_colors]

    plots[[paste0("i", i, "_j", j)]] <- plot(
      plot_data,
      col = plot_colors,
      pch = plot_symbols,
      main = paste("DBSCAN Clustering (i=", i, ", j=", j, ")"),
      xlab = "X-axis",
      ylab = "Y-axis"
    )

    legend("topright", legend = unique(dbscan_result$cluster),
           col = unique(plot_colors), pch = unique(plot_symbols))
  }
}
```

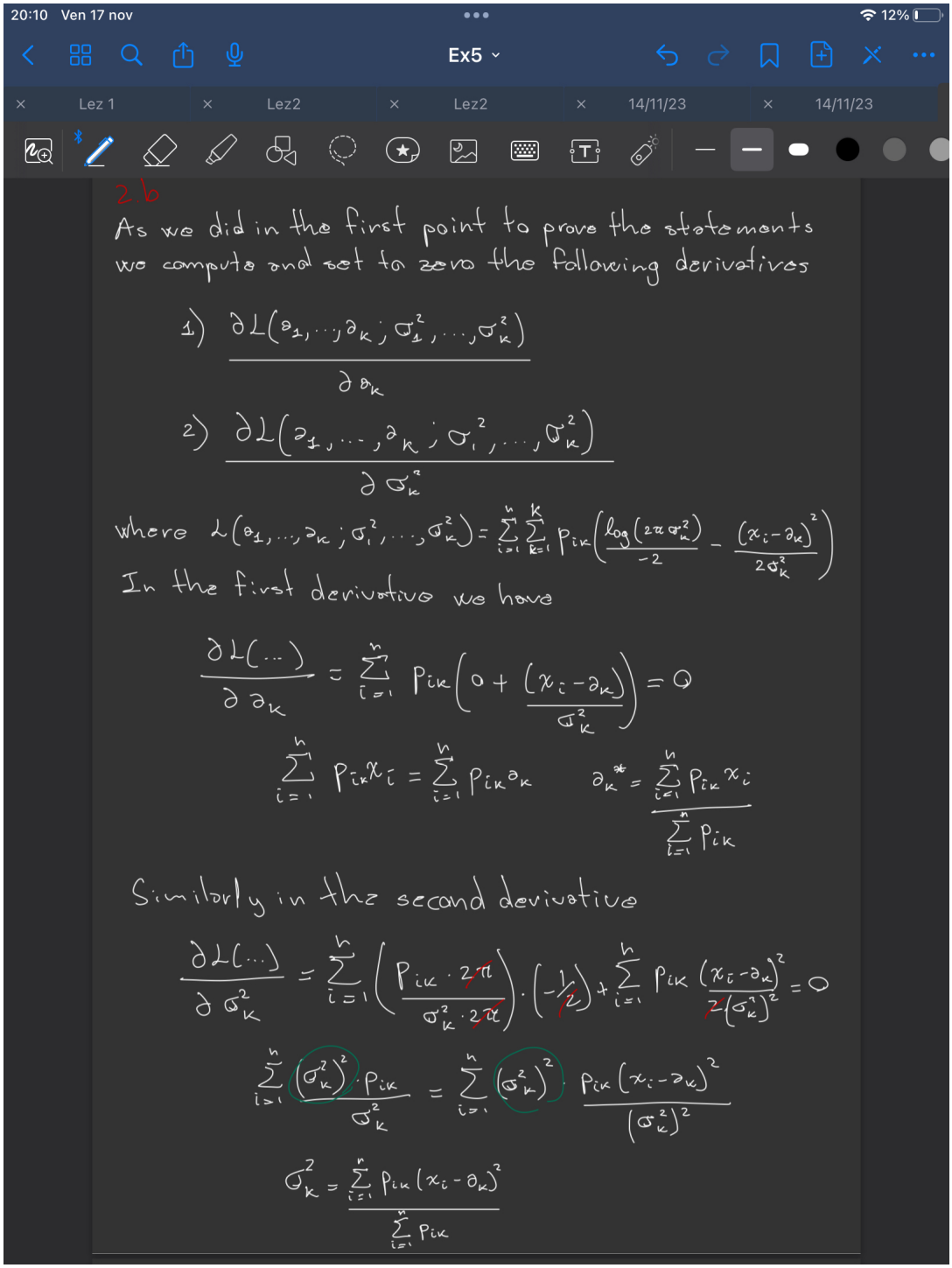
With this for loop we have different plots, here I present one of the better good looking plot that come out when the parameters were 0.9 for the distance and 5 for the number of neighbours.

2 Point 2

2.1 Point 2.a



2.2 Point 2.b


20:10 Ven 17 nov
12%
Ex5
Lez 1
Lez2
Lez2
14/11/23
14/11/23
2.b
As we did in the first point to prove the statements we compute and set to zero the following derivatives
1) $\frac{\partial \mathcal{L}(a_1,\dots,a_K;\sigma_1^2,\dots,\sigma_K^2)}{\partial a_K}$
2) $\frac{\partial \mathcal{L}(a_1,\dots,a_K;\sigma_1^2,\dots,\sigma_K^2)}{\partial \sigma_K^2}$
where $\mathcal{L}(a_1,\dots,a_K;\sigma_1^2,\dots,\sigma_K^2) = \sum_{i=1}^{n}\sum_{k=1}^{K} p_{ik}\left(\frac{\log(2\pi\sigma_k^2)}{-2} - \frac{(x_i-a_k)^2}{2\sigma_k^2}\right)$
In the first derivative we have
$\frac{\partial \mathcal{L}(\dots)}{\partial a_K} = \sum_{i=1}^{n} p_{ik}\left(0 + \frac{(x_i-a_k)}{\sigma_k^2}\right) = 0$
$\sum_{i=1}^{n} p_{ik}x_i = \sum_{i=1}^{n} p_{ik}a_k \qquad a_k^* = \frac{\sum_{i=1}^{n} p_{ik}x_i}{\sum_{i=1}^{n} p_{ik}}$
Similarly in the second derivative
$\frac{\partial \mathcal{L}(\dots)}{\partial \sigma_K^2} = \sum_{i=1}^{n}\left(\frac{p_{ik}\cdot 2\pi}{\sigma_K^2\cdot 2\pi}\right)\cdot\left(-\frac{1}{2}\right) + \sum_{i=1}^{n} p_{ik}\frac{(x_i-a_k)^2}{(\sigma_k^2)^2} = 0$
$\sum_{i=1}^{n}\frac{(\sigma_k^2)^2\cdot p_{ik}}{\sigma_k^2} = \sum_{i=1}^{n}(\sigma_k^2)^2\cdot\frac{p_{ik}(x_i-a_k)^2}{(\sigma_k^2)^2}$
$\sigma_k^2 = \frac{\sum_{i=1}^{n} p_{ik}(x_i-a_k)^2}{\sum_{i=1}^{n} p_{ik}}$

3 Point 3

3.1 Point 3.a

DBSCAN algorithm identifies the cluster based on the density of the data points: it does that in the following way.

1. Randomly pick a data point and look the neighborhood: if there are a minimum number of data near this point within a certain distance, this point is considered a "core point".
2. Then all the points in the neighborhood of the core point are included in the cluster represented by the core point.
3. The algo continues to include points in the region to form a dense cluster in the same way
4. All the points that are within the distance but does not have enough data points near to them to become a core point are called "border points"
5. Then, all the other points that are nor a core point nor a border points are considered as "noise" and do not belong to any cluster.

3.2 Point 3.b

According to the authors DBSCAN has the advantage of not requiring a prior knowledge of the cluster parameters (specially this problem arise when dealing with large spatial databases); the other argument is that DBSCAN requires only 1 input parameter and also support the user in determining suitable values for it.
I would say that the arguments are convincing.

3.3 Point 3.c

Inside the point 1