Unsupervised Clustering

Jan Izquierdo

2024-10-27

Libraries and seed setting

```
library("dbscan")
```

```
##
## Attaching package: 'dbscan'

## The following object is masked from 'package:stats':
##
##     as.dendrogram
```

```
library("MASS")
library(mclust)
```

```
## Package 'mclust' version 6.1.1
## Type 'citation("mclust")' for citing this R package in publications.
```

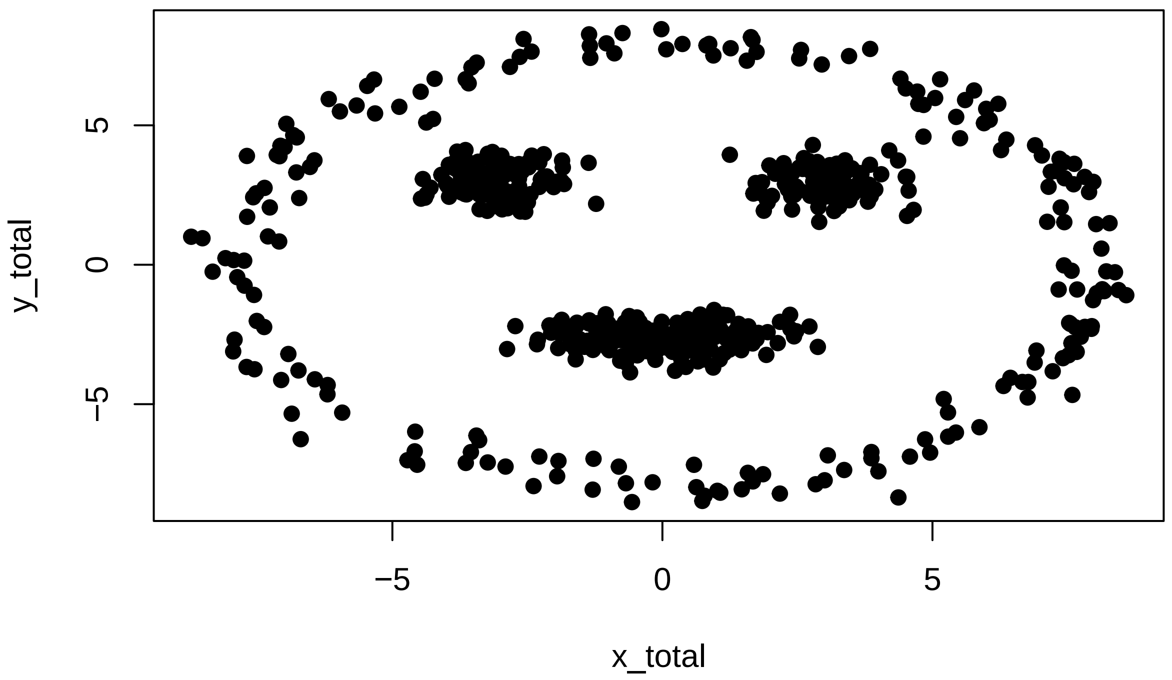
```
# Comment all the lines with informative comments.

set.seed(6)
```

Task 1)

Implement a dummy dataset with two dimensions that looks like Smiley.

```
#Covariance matrices, correlation noise x y
matrix_circle<-matrix(c(0.1, 0, 0, 0.1), ncol = 2)  # Low face noise
matrix_eyes<-matrix(c(0.3, 0, 0, 0.2), ncol = 2)    # More noise in the eyes
matrix_mouth<-matrix(c(0.7, 0, 0, 0.1), ncol = 2)   # And in the mouth too

#Generate angles
angles<-runif(200, 0, 2*pi)  #Make the circle: random angles
radius<-8 + rnorm(200, sd = 0.3)  #Face radius and noise
x_circle<-radius * cos(angles)  #x-cords
y_circle<-radius * sin(angles)  #y-cords


addNoise<-function(x, y, cov_matrix) {
  n<-length(x)
  noise<-mvrnorm(n, mu = c(0, 0), Sigma = cov_matrix)  #Generate multivariate noise
  x_n<-x + noise[,1]  #add noise to x
  y_n<-y + noise[,2]  #add noise to y
  return(list(x = x_n, y = y_n))  #give noised x and y
}
```

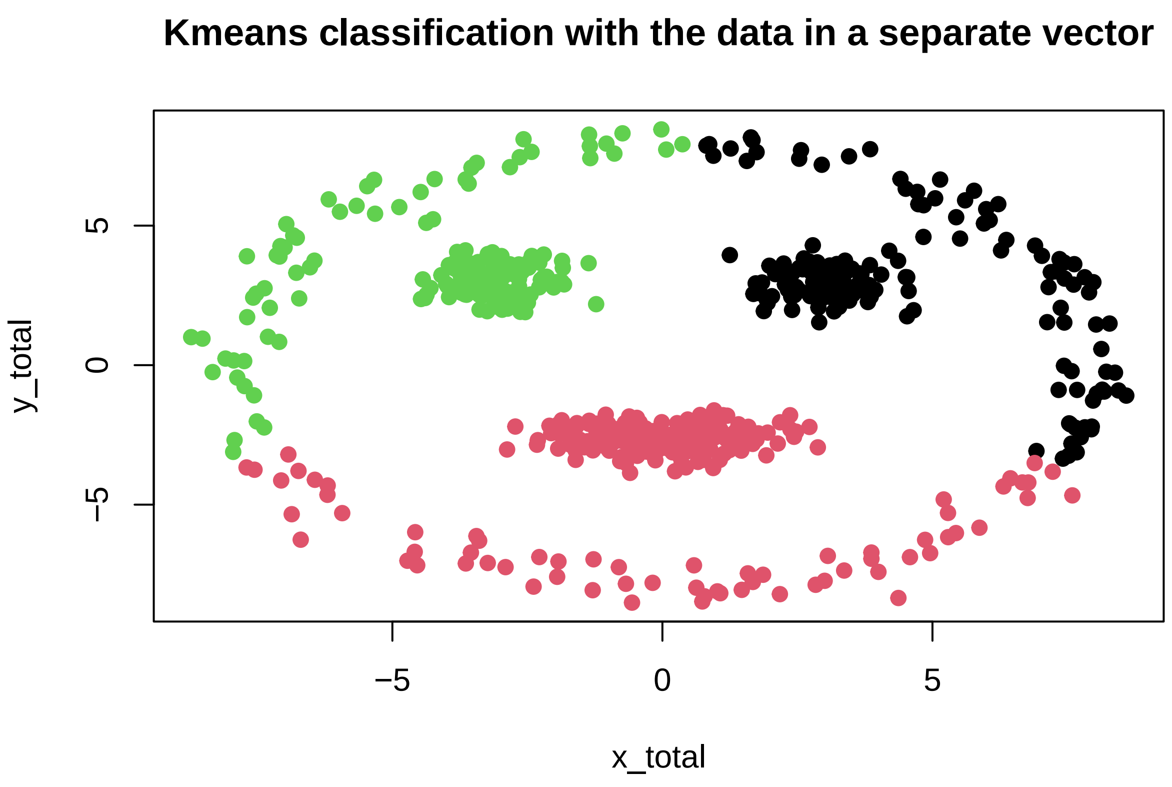
```r
#multivariate noise to the face circle
Ncircle<-addNoise(x_circle, y_circle, matrix_circle)

#Left eye
x_eye_l<-rnorm(100, mean = -3, sd = 0.3)  #xcords of left eye
y_eye_l<-rnorm(100, mean = 3, sd = 0.3)   #ycords of left eye

#Multivariate Noise
Neye_l<-addNoise(x_eye_l, y_eye_l, matrix_eyes) #Disperse the points of the eye

#right eye
x_eye_r<-rnorm(100, mean = 3, sd = 0.3)  #xcords of right eye
y_eye_r<-rnorm(100, mean = 3, sd = 0.3)   #ycords of right eye

#Multivariate Noise
Neye_r<-addNoise(x_eye_r, y_eye_r, matrix_eyes) #Disperse the points of the eye

#create mouth
#Mouth needs a circle, generate the angles to create a circle
mouth_angles<-seq(pi, 2*pi, length.out = 200)
mouth_radius<-1 + rnorm(200, sd = 0.2)  #small radius, if not it will cover face
x_mouth<-mouth_radius*cos(mouth_angles) #xcords for the mouth
#ycords of the mouth (-2 sets it in bottom of the face)
y_mouth<-mouth_radius*sin(mouth_angles)-2

#Multivariate Noise
Nmouth<-addNoise(x_mouth, y_mouth, matrix_mouth) #Disperse the points of the mouth

#Combine parts for x and for y
x_total<-c(Ncircle$x, Neye_l$x, Neye_r$x, Nmouth$x)
y_total<-c(Ncircle$y, Neye_l$y, Neye_r$y, Nmouth$y)

#Join the face as a dataframe
smiley<-data.frame(x_total, y_total)

#plot the face
plot(smiley, pch=19)
```



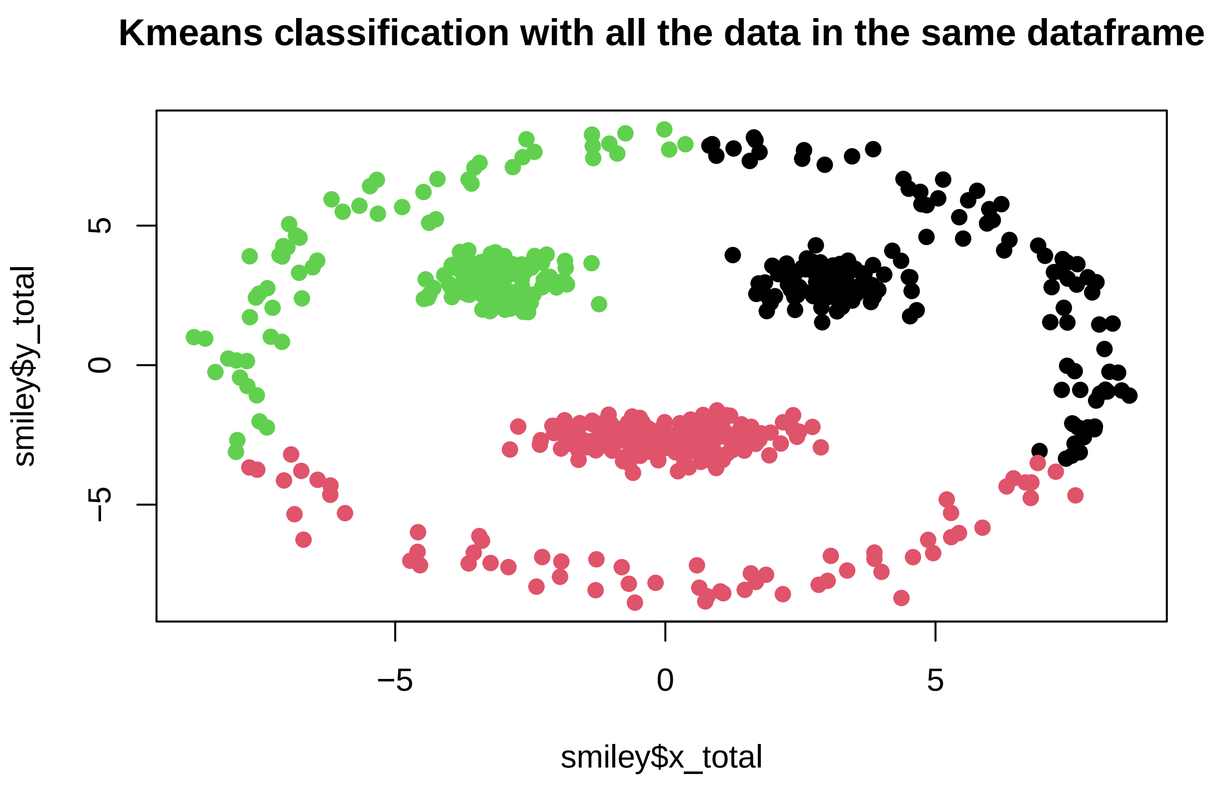
Task 2)

Apply to the Smiley face the kmeans algorithm and K = 3. Paint the result

```
k<-3 #k definition
kmeans_smile<-kmeans(smiley, centers = k) #execute algorithm

#plot the data with the color according to the kmeans classification
plot(smiley, pch=19, col=kmeans_smile$cluster,
     main="Kmeans classification with the data in a separate vector")
```



```r
#Its better if we add a column to the dataframe to represent the kmeans classification
#   instead of having a separate vector
smiley$cluster<-as.factor(kmeans_smile$cluster) #as a factor is better for coloring

#plot the data with no separate vectors
plot(smiley$x_total, smiley$y_total, pch=19, col=smiley$cluster,
     main="Kmeans classification with all the data in the same dataframe")
```



Do we get meaningfull clusters? Why?

Not really, because of the center being in the middle all point can belong to any cluster

Task 3)

Apply the kmeans with an increasing number of clusters (say, from 2 to 20).

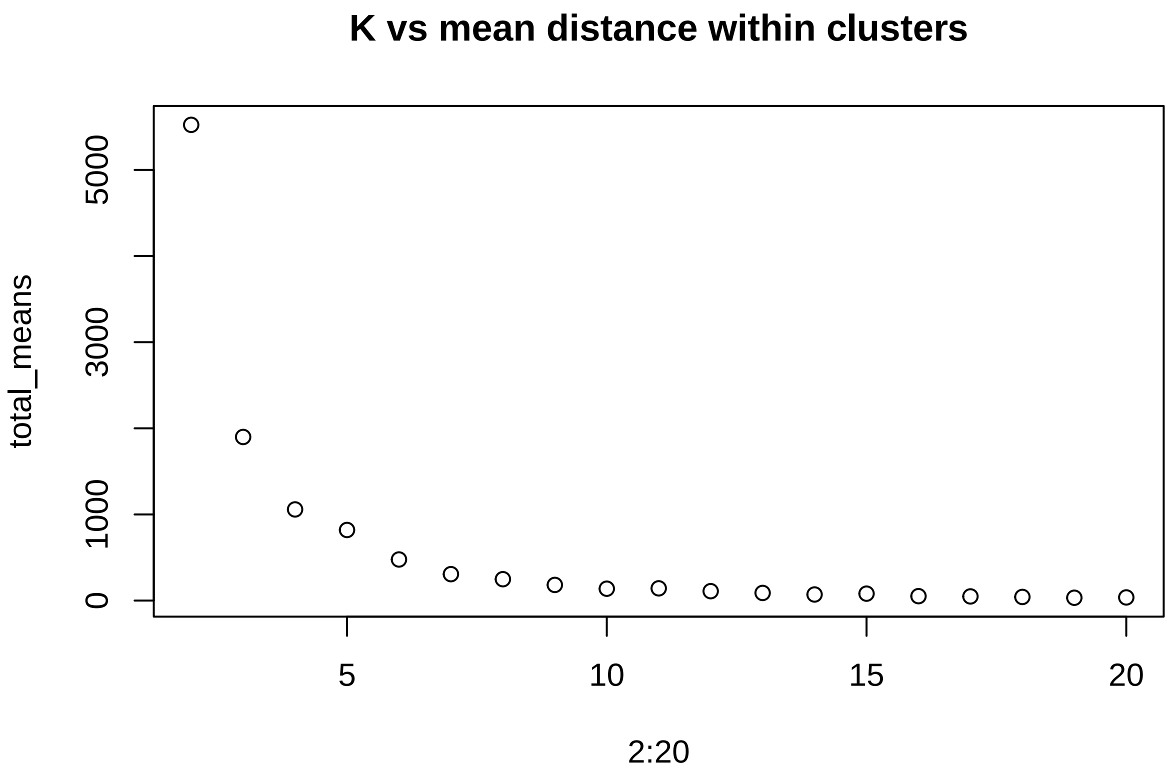
For each of the proposed k, compute the mean distance within each cluster.

Plot the result of K vs the average distance within

```r
#Check cluster class
class(smiley$cluster)
```

```
## [1] "factor"
```

```r
#change it from factor to numeric
smiley$cluster<-as.numeric(smiley$cluster)

total_means<-c() #define vector for mean storing
for (k in 2:20){ #iterate thorugh all desired Ks
  kmeans_res<-kmeans(smiley, centers = k) #execute algorithm
  mean_res<-mean(kmeans_res$withinss) #get mean of distances in the clusters
  total_means<-c(total_means, mean_res) #store mean in Means vector
}

plot(2:20, total_means, main="K vs mean distance within clusters")
```



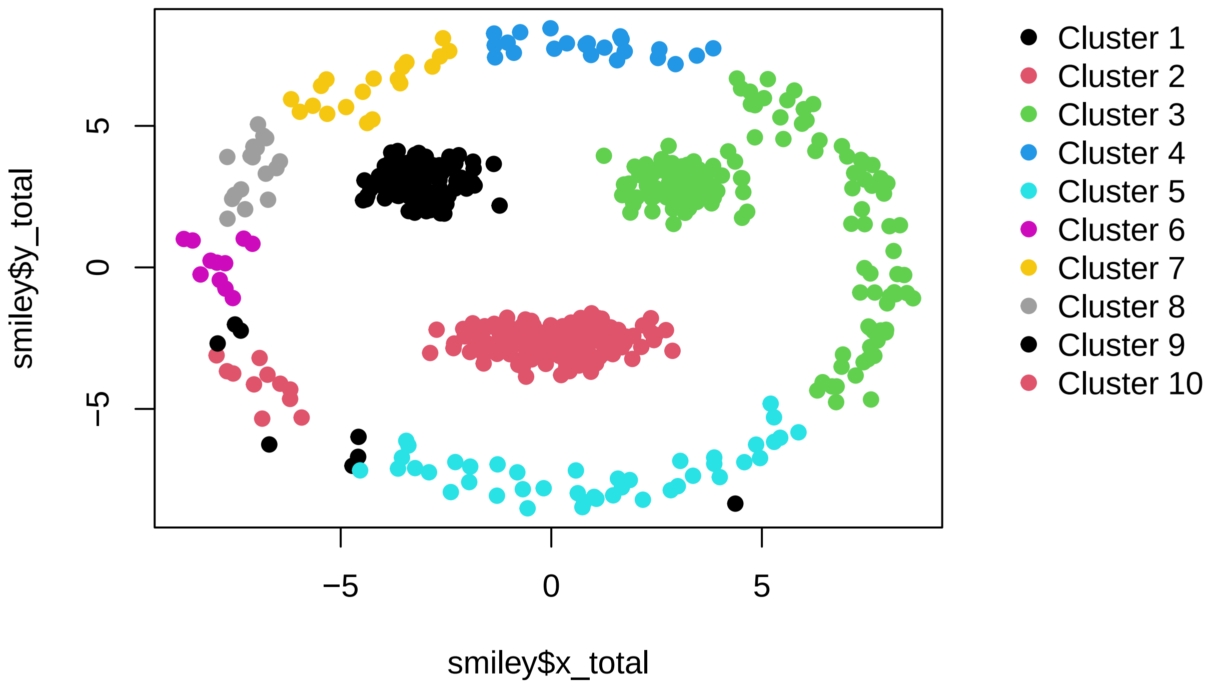
What is happening with the distance within each cluster? Why? What would be the minimum distance?

The distance is reduced with each K increase, before k=5 its very noticeable, after that the changes are progressively less noticeable. This is because there are more clusters so they are formed by closer points (which makes the distances smaller) It seems to follow a logarithmic function. The minimum distance would be the one where k=20.

Task 4)

Repeat the clustering with dbscan. Use minPts = 5. Do we need to specify the number of clusters? Why?

```
#Use dbscan, as eps is size of the neighborhood, set it to 1 (min value)
dbscan_cluster<-dbscan(smiley[,1:2], minPts = 5, eps=1)

#Add dbscan cluster results to the dataframe
smiley$dbscanR<-as.factor(dbscan_cluster$cluster)

#How many clusters?
num_clusters<-length(unique(dbscan_cluster$cluster))

#Plot the clustering in the dataset
par(mar = c(5, 4, 4, 8))
plot(smiley$x_total, smiley$y_total, col=smiley$dbscanR, pch=19)
legend("topright", legend = paste("Cluster", 1:num_clusters), col = 1:num_clusters,
       pch = 19, inset=c(-0.35,-0.02), xpd=TRUE, bty="n")
```



We don't specify the number of clusters, dbscan creates them based on the number of neighborhood points and size.

How many clusters do you identify? What is happening?

I identify 10 clusters.

Task 5)

Create a dummy dataset of two features and 150 observations. Each 50 observations come from a different multivariate normal distribution with its own mean vector and variance covariance matrix

```r
set.seed(42) #Set the hitchhiker seed

#Obs=150 in groups of 50 -> 3 groups
n<-50

#Mean vectors and covariance matrices for all distributions
mean1<-c(2, 4) #x, y
cov1<-matrix(c(1, 0.2, 0.7, 0.5), nrow = 2)

mean2<-c(5, 8)
cov2<-matrix(c(2, -0.4, 0.3, 1), nrow = 2)

mean3<-c(3, 9)
cov3<-matrix(c(1, 0.7, 1, 0.6), nrow = 2)

#Use mvrnorm to create 150(50 each) sample from the means and matrices
s1<-mvrnorm(n, mu = mean1, Sigma = cov1)
s2<-mvrnorm(n, mu = mean2, Sigma = cov2)
s3<-mvrnorm(n, mu = mean3, Sigma = cov3)

#add labels to the generated data
```

```r
s1<-cbind(s1, rep(1, each=n))
s2<-cbind(s2, rep(2, each=n))
s3<-cbind(s3, rep(3, each=n))

#combine data into a data frame
data<-data.frame(rbind(s1,s2,s3))
```

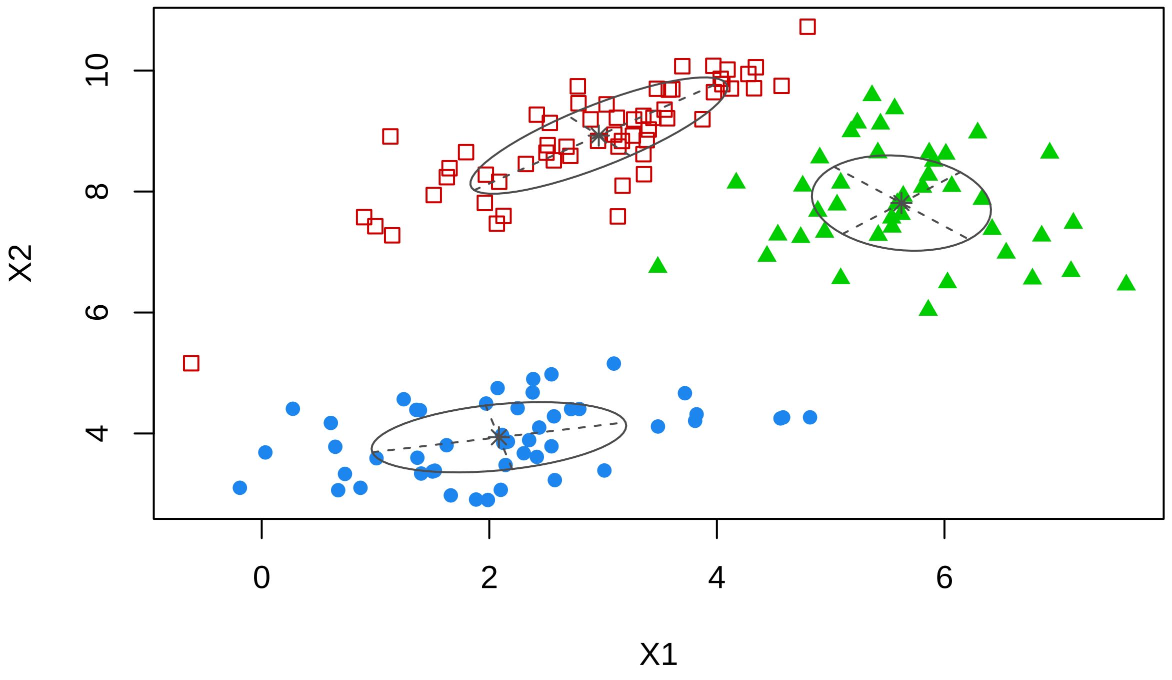
Task 6)

Apply mclust, which applies a mixture of Gaussian distributions. Check the Mclust command. What is G? How can you use it?

```r
#Model with Mclust only for the columns that contain data (NO LABELS)
mc_model<-Mclust(data[, c(1, 2)])

#check model
summary(mc_model)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust EVV (ellipsoidal, equal volume) model with 3 components:
##
##  log-likelihood   n df       BIC       ICL
##       -511.5395 150 15 -1098.238 -1101.533
##
## Clustering table:
##  1  2  3
## 49 59 42
```

```r
#plot the model, classification gives clearest results
plot(mc_model, what = "classification")
```



```r
#Get number of clusters in the model
n_clust<-mc_model$G
#G is the number of clusters used in the model
```

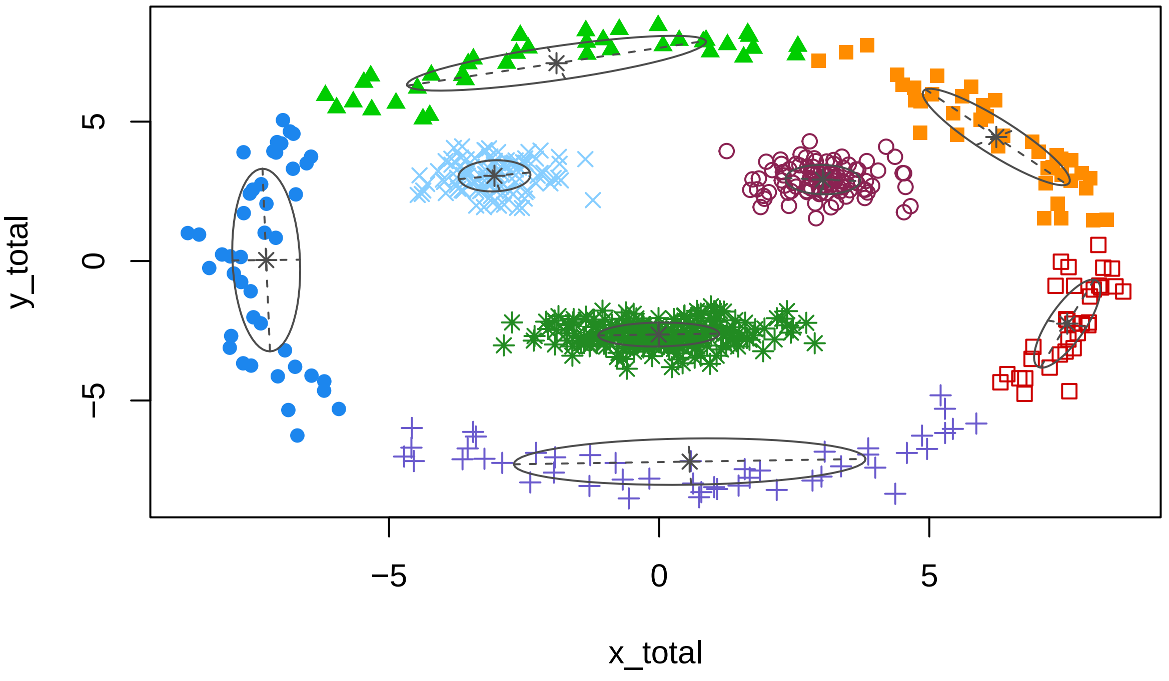
G is the number of clusters in the model, in this case G=3

What would happen if you try with the smiley.face? Why is this happening?

```r
#repeat the process

#Check what columns to take
summary(smiley)
```

```
##     x_total           y_total            cluster          dbscanR
##  Min.   :-8.7252   Min.   :-8.5116   Min.   :1.000   9      :200
##  1st Qu.:-2.5477   1st Qu.:-2.7537   1st Qu.:1.000   2      :169
##  Median : 0.1253   Median : 0.2018   Median :2.000   8      :100
##  Mean   : 0.2269   Mean   : 0.1795   Mean   :1.977   4      : 43
##  3rd Qu.: 2.8435   3rd Qu.: 3.1948   3rd Qu.:3.000   3      : 22
##  Max.   : 8.5885   Max.   : 8.4477   Max.   :3.000   6      : 19
##                                                      (Other): 47
```

```r
mc_sm_model<-Mclust(smiley[, c(1, 2)])

#plot the model, classification gives clearest results
plot(mc_sm_model, what = "classification")
```



G is the number of clusters in the model, in this case G=8

It has a different number of clusters than before as the algorithm we have used is different, and thus the clusters are calculated differently