Definitions :

Clustering : Clustering is statistique method in Data Analysis who organise values in group by their "degré de similitude". His objective is to identify and visualize sets of similar items in terms of define criteria. Organisation Strategy that partition a set of hetero gene data in homo gene sub-sets. Each sub-set is obtain by assembling elements with the same characteristic.
K-Means : With points and one int k, the algorithm aim to divide points in k group name clusters homogeneous and compact. We choose randomly k individuals in our values, and then we compare each points distance to all k chosen values. And then when all individuals have been looked at, we choose a new "gravity center" and we continue to look into our individuals whenever our center point doesn't move anymore.
Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. One can think of mixture models as generalizing k-means clustering to incorporate information about the covariance structure of the data as well as the centers of the latent Gaussians.
A mixture model is a probabilistic model for representing the presence of sub-populations within an overall population, without requiring that an observed data set should identify the sub-population to which an individual observation belongs. Formally a mixture model corresponds to the mixture distribution that represents the probability distribution of observations in the overall population. However, while problems associated with "mixture distributions" relate to deriving the properties of the overall population from those of the sub-populations, "mixture models" are used to make statistical inferences about the properties of the sub-populations given only observations on the pooled population, without losing sub-population identity information.
EM-Algorithm : The EM algorithm is an iterative approach that cycles between two modes. The first mode attempts to estimate the missing or latent variables, called the estimation-step or E-step. The second mode attempts to optimize the parameters of the model to best explain the data, called the maximization-step or M-step.
E-Step : Estimate the missing variables in the dataset.
M-Step : Maximize the parameters of the model in the presence of the data.
The EM algorithm can be applied quite widely, although is perhaps most well known in machine learning for use in unsupervised learning problems, such as density estimation and clustering. Perhaps the most discussed application of the EM algorithm is for clustering with a mixture model.
ACP(Principal Component Analysis) : Technique of Analysis Statistic, mostly descriptive, by modeling the most informations as possible in a table. It allows to visualize a space with p dimensions with space of smaller dimensions. Division of the data in some components into sub-space (orthogonal plan), and still keeping all the information.
Kernel PCA : PCA is a linear method. That is it can only be applied to datasets which are linearly separable. But, if we use it to non-linear datasets, we might get a result which may not be the optimal dimensionality reduction. Kernel PCA uses a kernel function to project dataset into a higher dimensional feature space, where it is linearly separable. Kernel Method will help us mesure the similarity between variables.
Example : Gaussian, Polynomial, Linear.
Spectral Clustering : In multivariate statistics, spectral clustering techniques make use of the spectrum (eigenvalues) of the similarity matrix of the data to perform dimensionality reduction before clustering in fewer dimensions. The similarity matrix is provided as an input and consists of a quantitative assessment of the relative similarity of each pair of points in the dataset. For example, you can apply it to a population, if you want to promote your product to a large part of the population you can use spectral clustering to target people who are influent in different communities and then use them in order to promote.

Ancien Partiel

3 - Express $t_{ik} = P(Z_{ik} = 1, X_i = x_i)$ as a function of the mixture parameters.
On utilise la formule de Bayes pour obtenir 3 probabilités que l'on peut calculer :

$$t_{ik} = P(Z_{ik} = 1, X_i = x_i) = \frac{P(X_i = x_i | Z_{ik} = 1)P(Z_{ik} = 1)}{P(X_i = x_i)} \tag{1}$$

On peut séparer ça en 3 composantes que l'on sait calculer.
- $P(Z_{ik} = 1) = \pi$, c'est la probabilité d'être dans le premier modèle.
- $P = \mathcal{N}(x_i, \mu_1, \sigma_1)$
- $P(X_i = x_i) = \sum_{i=1}^{K} P(X_i = x_i | Z_{ik} = 1)P(Z_{ik} = 1)$

On a donc :

$$t_{ik} = \frac{\pi_1 * \frac{1}{\sqrt{2\pi\sigma^2}} \exp(\frac{-1}{2}(\frac{x_i - \mu_1}{\sigma_1})^2)}{\sum_{k=1}^{K} P(X_i = x_i | Z_{ik} = 1)P(Z_{ik} = 1)} = \frac{\pi_1 * \frac{1}{\sqrt{2\pi\sigma^2}} \exp(\frac{-1}{2}(\frac{x_i - \mu_1}{\sigma_1})^2)}{\sum_{l=1}^{K} \frac{1}{\sqrt{2\pi\sigma_L^2}} \exp(\frac{-1}{2}(\frac{x_i - \mu_L}{\sigma_L})^2) * \pi_L} \tag{2}$$

4 - Compute the joint Distribution $P(= 1, X_i = x_i)$.
Close to the previous method.

$$P(Z_{ik} = 1, X_i = x_i) = P(X_i = x_i | Z_i k = 1)P(Z_i k = 1) = \mathcal{N}(x_i; \mu_1, \sigma_1)\pi_1 \tag{3}$$

5 - Express the so called log-likehood of the mixture parameters $log(P(\mathcal{X}, \mathcal{Z}; \Theta))$ where $\Theta = (\mu_1, \sigma_1, \mu_2, \sigma_2, \pi)$

$$log(P(\mathcal{X}, \mathcal{Z}; \Theta)) = log(\Pi_{k=1}^{K} \Pi_{i=1}^{n} P(x_i, \mu_k, \sigma k; \Theta)) = \sum_{k=1}^{K} \sum_{i=1}^{n} log(P(X_i = x_i, Z_{ik} = k; \Theta_k)) \tag{4}$$

$$= \sum_{k=1}^{K} \sum_{i=1}^{n} log(\pi \frac{1}{\sqrt{2\pi\sigma_k^2}} exp(\frac{-1}{2}(\frac{x_i - \mu_L}{\sigma_L})^2)) \mathbb{1}_{Z_{ik}=1}$$

$$= \sum_{k=1}^{K} \sum_{i=1}^{n} (log(\pi) - \frac{1}{2} log(2\pi) - \log(\sigma_1) - \frac{1}{2} \frac{(x_i - \mu_i)^2}{\sigma^2}) \mathbb{1}_{Z_i = k}$$

6 - Provide the expression of $\mathcal{Q}(\Theta, \Theta^q) = E_{Z|X;\Theta^q}[logP(\mathcal{X}, \mathcal{Z}; \Theta^q]$.

$$\mathcal{Q}(\Theta, \Theta^q) = E_{Z|X;\Theta^q}[logP(\mathcal{X}, \mathcal{Z}; \Theta^q] = \sum_{k=1}^{K} \sum_{i=1}^{n} \frac{1}{2}(log(\pi) - log(2)) - log(\sigma_1) - \frac{1}{2} \frac{(x_i - \mu_i)^2}{\sigma^2}) E_{Z_i|X_i;\Theta^q} \mathbb{1}_{Z_i = k} \tag{5}$$

Exercice II - Kernel PCA

1 - Describe the KPCA in algorithm in a few lines.
Kernel PCA : PCA is a linear method. That is it can only be applied to datasets which are linearly separable. But, if we use it to non-linear datasets, we might get a result which may not be the optimal dimensionality reduction. Kernel PCA uses a kernel function to project dataset into a higher dimensional feature space, where it is linearly separable. Kernel Method will help us mesure the similarity between variables.
2 - Show that v can be expressed as a linear combinaison of the $\phi(x_i)$

3 - Assume you have a data matrix X size n x p where each row describes an object by p variables. write a R code which produce a Gaussian kernel matrix K from X. Let x_i and x_j be 2 rows of X the gaussian kernel is defined by $exp(-\beta||x_i - x_j||^2)$.

```
 K <- matrix(0, nrow = n, ncol = p)
for(i in 1:n)
for(j in 1:p)
K[i, j] <- nrow(X[i,] - X[j,], type = "2")
K <- exp(-βK^2)
```

Exercice III - PCA

R Code
Let X be a n x p matrix whose x_i describes an individual.
a - What are the input and the output of the PCA.

- **Input** : Matrix X
- **Output** : Matrice C des principales composantes de X. Cercle de corrélation et graphique des individus et de leur variance(axes).

b - Write your own pseudo code for PCA.

PCA < - function(X) —— $A_{ij} = \delta(X_i, X_j)$
X < - scale(X, center = TRUE, scale = TRUE) —— $J = I - \frac{1}{n} 1_{(n,n)}$
spec < - eigen(var(X)) —— $B = -\frac{1}{2} J \Delta J$
U < - specDOLLvectors
C < - as.matrix(X) Percent * Percent U
return(c)

$$K_\phi(X_i, X_j) = exp(-\frac{1}{2\sigma^2} * ||X_i - X_j||_2^2) \text{Equation for spectral Clustering}$$
$$= exp(-\sigma||X_i - X_j||)$$
$$\text{with : } \sigma = 1$$

Consider the Iris data set. Write a R code wich produces the partition matrix. Compute the gravity centers of the quantitative variables in the three classes using a matrix formula. With matrix multiplication P*P instead or percent*percent.

data(iris)
X<-iris[,1:4]
library(nnet)
C<-class.ind(irisDOLLARSpecies) Matrice partition
t((t(X) P*P C))/diag(t(C) P*P C)
image(C)

In this graphic we can see that there is 3 groups actually, this we lead us to our result. We need to understand why we do have 3 groups, there is actually 3 types of flowers so it s normal that there is 3 groups. We will use the kmeans method to try to plot our data :

kmeans.res <- iris P>P select(c(-Species,-Sepal.Length,-Sepal.Width)) P>P kmeans(3,nstart = 10)
cluster<-as.factor(kmeans.resDOLLARcluster)
centers <- as.data.frame(kmeans.resDOLLARcenters)

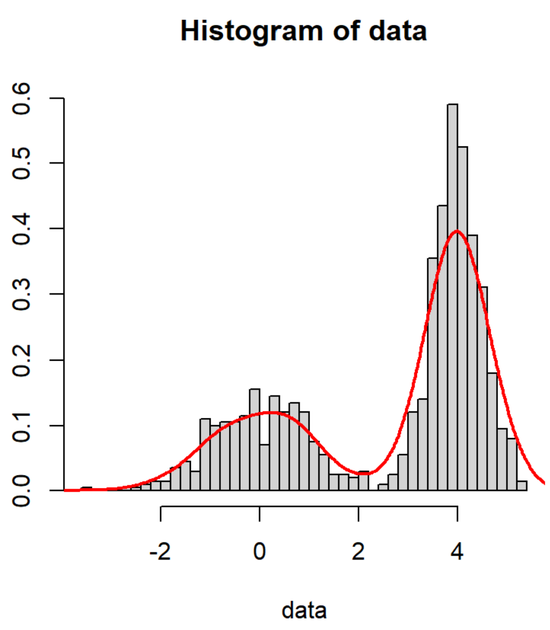
When the number of cluster is increasing the sum of square slowly tend to 0.

n<-1000
Proportions<-c(1,2)/3
parameters<-list()
parameters[[1]]<-list(mean=0,sd=1)
parameters[[2]]<-list(mean=4,sd=1/2)

simu-mixture<-function(n=100,parameters,Proportions)
z<-t(rmultinom(n,1,prob = Proportions))
z<-apply(z,1,which.max)
x<-matrix(0,n,1)
for (i in 1:length(z))
x[i,1]<-rnorm(1,mean=parameters[[z[i]]]DOLLARmean, sd=parameters[[z[i]]]DOLLARsd)
return(list(x=x,z=z))

hist(simu-mixture(n=1000, parameters=parameters, Proportions = Proportions)DOLLARx,breaks=50)

By doing this we create a vector with all of ours parameters and we can now use a histogram to plot our values :



We can see here that we have two major values that are highlight here, 0 and 4 the two means given for this mixture. Now we are using *K-Means* algorithm :

res<-kmeans(simulationDOLLARx,2,nstart = 30)

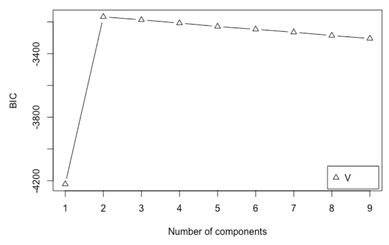
Mclust with model E and V :
So we do apply our Algorithm EV, we apply it with constant variance, here and see what we got :

model-Mclust-E <- Mclust(simulationDOLLARx, G=2, modelNames="E") equal variance
model-Mclust-EDOLLARparameters

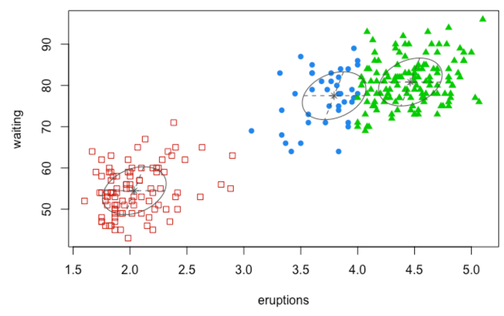
model-Mclust-V <- Mclust(simulationDOLLARx, G=2, modelNames="V") equal variance
model-Mclust-VDOLLARparameters

plot(model-Mclust-VDOLLARBIC)

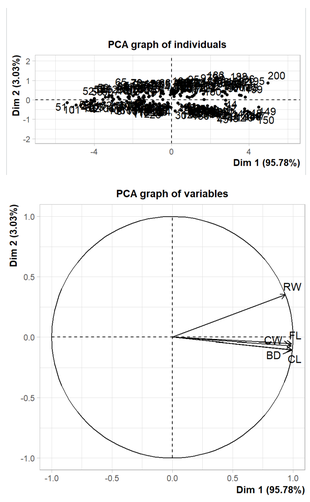
So after applying our algorithm, our goal is to compare each one of the 3 answers we got and try to understand why do we got same or close values. We do plot BIC, and see the difference between the number of composents :



data(faithful) , here we find the number of cluster
summary(faithful), actually Mclust choose it for us
plot(faithfulDOLLARwaiting, faithfulDOLLAReruptions)
model <- Mclust(faithful)
plot(model)

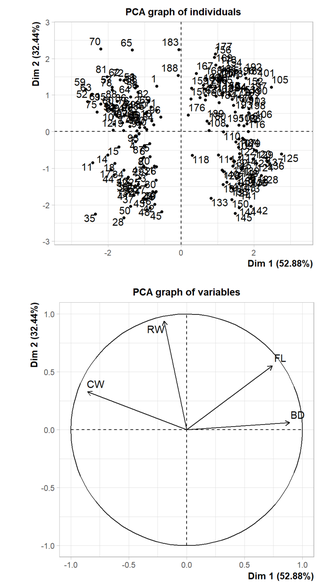


data(crabs)
crabsquant <- crabs[,4:8]
plot(crabsquant)
PCA(crabsquant)
prcomp(crabsquant)



We can see that one of our axis is explaining about 95 percent of our data. This result is obtained because those variable are correlated, to try to spread out our analysis we need to decorrelate it. So we need to find the most correlated variable to divide all of the others by this one in order to obtain a new dataset to work on, this is in sort of a normalization. In order to obtain this column u can plot the matrix of correlation. So we do :

crabsquant-decorel <-
crabsquant/crabsquantDOLLARCL
crabsquant-decorel <- crabsquant-decorel[,-3]
PCA-crabs <- PCA(crabsquant-decorel)



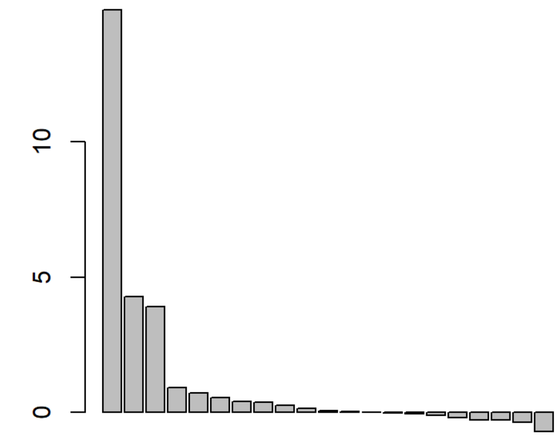
Blue/Male Orange/Male
Blue/Female Blue/Female

Correlation Circle

And now 85 percent of the variance is explained with 2 axis.

D = Dissimilarity Matrix. This matrix contains the distance between x_j and x_i.
This matrix has a diagonal null, is defined positive and symetric. Δ is a matrix with the square of every A coefficient. Cf behind for J and B matrix

tmp <- eigen(B)
J <- tmpDOLLvectors
A <- diag(tmpDOLLvalues)



In Rstudio, eigen value are ranged by decreasing values and we choose to keep 3 of them because they will mostly explained about 85 percent of the dataset. In order to reduce the dimension and to explain more of the dataset. The eigen vectors with the highest eigen value is the one that does explain the most about the model. Each vector will be orthogonal to the previous in order to create a space with m dimensions.

P = percent
library(dplyr)
spam-quant <- spam P>P select(-"type")
spam-quant-norm <- scale(spam-quant)
res.PCA <- PCA(spam-quant-norm, graph = F)
eigvalues <- data.frame(res.PCADOLLeig)
barplot(eigvaluesDOLLpercentage.of.variance, names.arg = row.names(eigvalues))
X <- res.PCADOLLindDOLLcoord
plot(X[,1], X[,2],
col = as.numeric(spamDOLLtype)+2)

m = 100 N =nrow(spam-quant-norm)
Tirage <- sample(1:N, m, replace = FALSE)
spam-subset = spam-quant-norm[Tirage,]
kpc <- kpca(., data = as.data.frame (spam-subset), kernel = "rbfdot", kpar = list(sigma=0.01))
kpvc <- pcv(kpc)
plot(rotated(kpc)[,1:2], xlab = "1st Principale Comp.", ylab = "2nd Principale Comp.")
barplot(eig(kpc)/sum(eig(kpc)))

Here we have done kernel PCA with a '*RBF*' a linear model, with now need to work with different type of kernel to compare our result. Around 25 percent of variance is explained so we need to apply few more methods. In the second plot with a better distribution of the values in the barplot.