Expectation-maximization

Nik Bear Brown

In this lesson, we cover expectation-maximization theory, apply and evaluate expectation-maximization based clustering.

Additional packages needed

To run the code in M04_Lesson_03.Rmd you may need additional packages.

- If necessary install these packages.

```
install.packages("mclust");
```

```
require(mclust)
```

```
## Loading required package: mclust
```

```
## Package 'mclust' version 5.1
```

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

Data

We'll be using the "diabetes" data that comes with the mclust (https://cran.r-project.org/web/packages/mclust/index.html) package.

```
data(diabetes)
summary(diabetes)
```

```
##       class       glucose       insulin            sspg
##  Chemical:36   Min.   : 70   Min.   :  45.0   Min.   : 10.0
##  Normal  :76   1st Qu.: 90   1st Qu.: 352.0   1st Qu.:118.0
##  Overt   :33   Median : 97   Median : 403.0   Median :156.0
##                Mean   :122   Mean   : 540.8   Mean   :186.1
##                3rd Qu.:112   3rd Qu.: 558.0   3rd Qu.:221.0
##                Max.   :353   Max.   :1568.0   Max.   :748.0
```

Data References

G.M. Reaven and R.G. Miller, Diabetologica 16:17-24 (1979).

Expectation-maximization (EM)

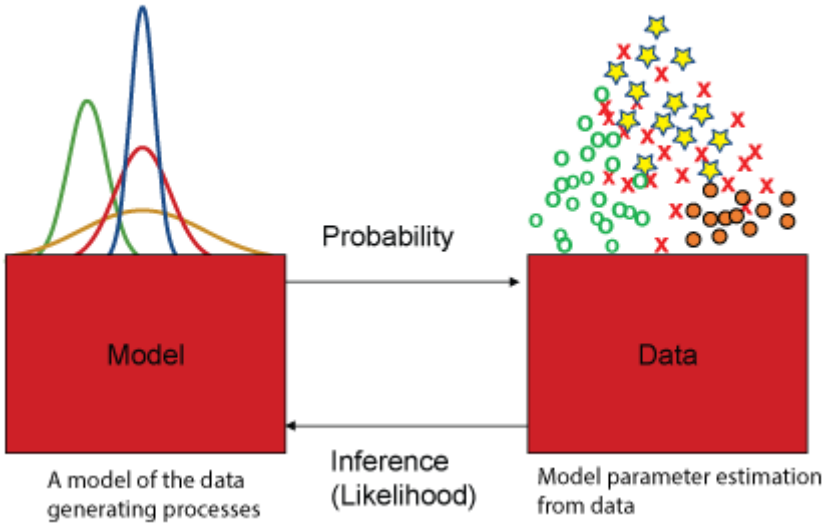
The expectation-maximization (EM) algorithm (https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm) is an iterative method for finding maximum likelihood or maximum a posteriori (MAP) estimates of parameters in statistical models, where the model depends on unobserved latent variables. Expectation Maximization (EM) is perhaps most often used algorithm for unsupervised learning.

It is based upon probabilistic model building with partial observations

Likelihood Function

$$P(Model|Data) = \frac{P(Data|Model)P(Model)}{P(Data)}$$

The Likelihood Function finds the *best* model given the data.



expectation-maximization (EM) algorithm

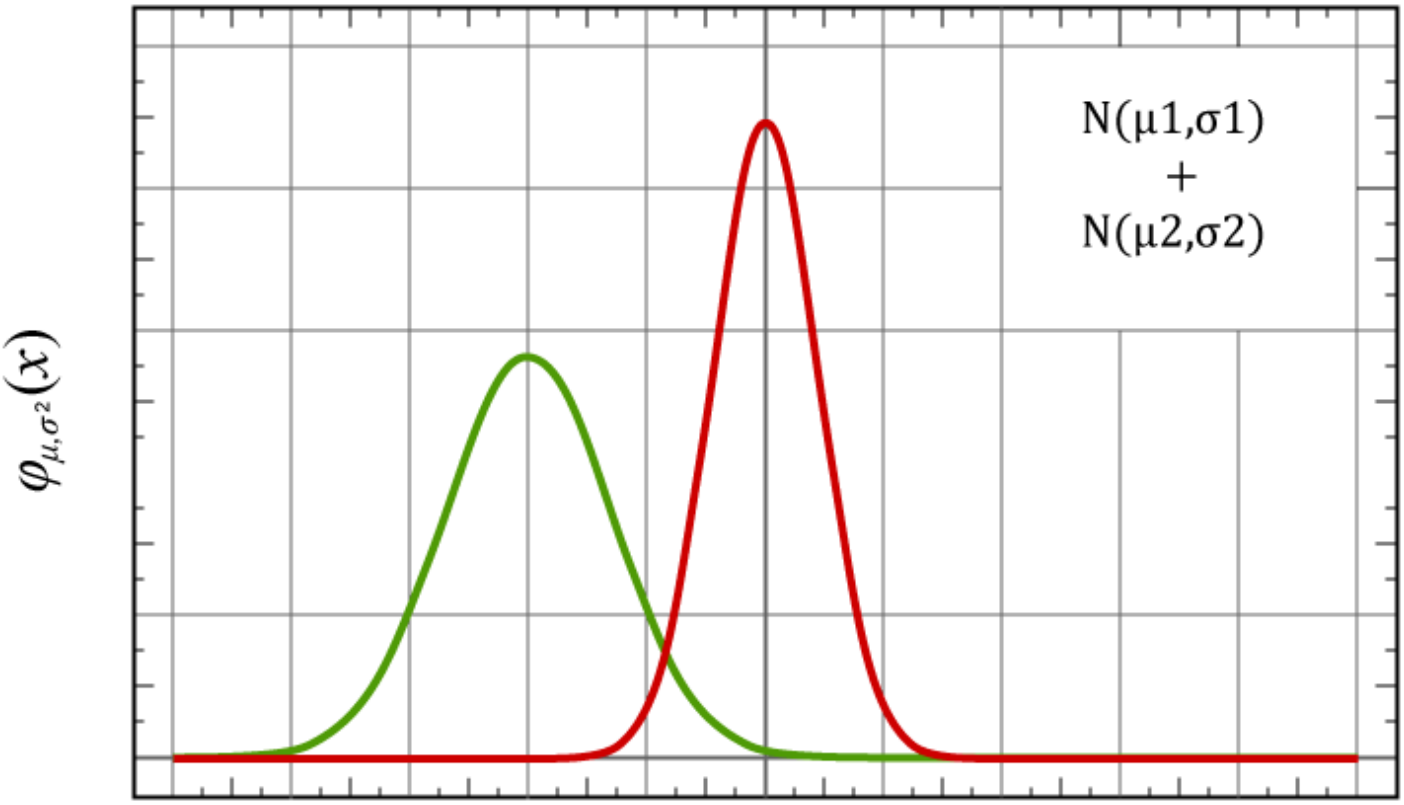
Suppose we flip a coin and get the following - 0,1,1,0,0,1,1,0,0,1. We may choose a Bernoulli distribution (https://en.wikipedia.org/wiki/Bernoulli_distribution)

$$Pr(X = k) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0, 1\}.$$

And estimate the most likely p.

Or if we have data of heights of people in cm (both male and female). Heights follow a normal distrution but men are (on average) taller than women so this would suggest a mixture model (https://en.wikipedia.org/wiki/Mixture_model) of two Gaussian distributions (https://en.wikipedia.org/wiki/Normal_distribution).

$$\mathcal{N}(\mu_1, \sigma_1) + \mathcal{N}(\mu_2, \sigma_2)$$



Mixture model of two Gaussian distributions

Maximum likelihood

maximum likelihood idea:

- If $A_1, \ldots A_n$ are independent then for every n-element subset A_i,

$$\mathrm{P}\left(\bigcap_{i=1}^{n} A_i\right) = \prod_{i=1}^{n} \mathrm{P}(A_i).$$

This is the multiplication rule for independent events.

- The log function is monotonically increasing. Therefore if a function $f(x) \geq 0$, achieves a maximum at x_1, then $log(f(x))$ also achieves a maximum at x_1.

Example of MLE

Suppose we flip a coin and get the following - 0,1,1,0,0,1,1,0,0,1.

$$L(p) = P(0, 1, 1, 0, 0, 1, 1, 0, 0, 1|p)$$

$$= P(0|p)P(1|p)\ldots P(1|p)$$

$$= p^5(1-p)^5$$

choose p which maximizes L(p). Instead we will maximize $l(p) = logL(p)$

$$l(p) = logL(p) = 5log(p) + 5log(1-p)$$

$$= \frac{5}{p} - \frac{5}{1-p} \equiv 0(at \quad max)$$

$$\Rightarrow p = \frac{1}{2}$$

EM Algorithm

Given a statistical model which generates a set $\mathbf{X}$ of observed data, a set of unobserved latent data or missing values $\mathbf{Z}$, and a vector of unknown parameters $\boldsymbol{\theta}$, along with a likelihood function $L(\boldsymbol{\theta}; \mathbf{X}, \mathbf{Z}) = p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$, the maximum likelihood estimate (MLE) of the unknown parameters is determined by the marginal likelihood of the observed data

$$L(\boldsymbol{\theta}; \mathbf{X}) = p(\mathbf{X}|\boldsymbol{\theta}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$$

Create an initial model, θ_0.

Decide on a model with arbitrary paramters.

The EM phase seeks to find the MLE of the marginal likelihood by iteratively applying the following two steps:

- *Expectation step*: the (missing) data are estimated given the observed data and current estimates of model parameters

Calculate the expected value of the log likelihood function, with respect to the conditional distribution of $\mathbf{Z}$ given $\mathbf{X}$ under the current estimate of the parameters $\boldsymbol{\theta}^{(t)}$:

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) = \mathrm{E}_{\mathbf{Z}|\mathbf{X},\boldsymbol{\theta}^{(t)}}\left[\log L(\boldsymbol{\theta}; \mathbf{X}, \mathbf{Z})\right]$$

- *Maximization step*: The likelihood function is maximized under the assumption that the (missing) data are known

Maximization step (M step): Find the parameter that maximizes this quantity:

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$$

Repeat the above EM steps until reaching a local maximum.

Mixture of Gaussians

A very common “mixture model” is a mixture of Gaussians. For example, let $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ be a sample of n independent observations from a mixture of two multivariate normal distributions of dimension d, and let $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ be the latent variables that determine the component from which the observation originates.

$$X_i|(Z_i = 1) \sim \mathcal{N}_d(\boldsymbol{\mu}_1, \Sigma_1) \quad and \quad X_i|(Z_i = 2) \sim \mathcal{N}_d(\boldsymbol{\mu}_2, \Sigma_2)$$

where

$$\mathrm{P}(Z_i = 1) = \tau_1 \quad and \quad \mathrm{P}(Z_i = 2) = \tau_2 = 1 - \tau_1$$

The aim is to estimate the unknown parameters representing the “mixing” value between the Gaussians and the means and covariances of each:

$$\theta = \left(\boldsymbol{\tau}, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \Sigma_1, \Sigma_2\right)$$

where the incomplete-data likelihood function is

$$L(\theta; \mathbf{x}) = \prod_{i=1}^{n} \sum_{j=1}^{2} \tau_j \, f(\mathbf{x}_i; \boldsymbol{\mu}_j, \Sigma_j),$$

and the complete-data likelihood function is

$$L(\theta; \mathbf{x}, \mathbf{z}) = P(\mathbf{x}, \mathbf{z}|\theta) = \prod_{i=1}^{n} \sum_{j=1}^{2} \mathbb{I}(z_i = j) \, f(\mathbf{x}_i; \boldsymbol{\mu}_j, \Sigma_j) \tau_j$$

or

$$L(\theta; \mathbf{x}, \mathbf{z}) = \exp\left\{ \sum_{i=1}^{n} \sum_{j=1}^{2} \mathbb{I}(z_i = j) \left[\log \tau_j - \tfrac{1}{2} \log |\Sigma_j| - \tfrac{1}{2} (\mathbf{x}_i - \boldsymbol{\mu}_j)^\top \Sigma_j^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_j) - \tfrac{d}{2} \log(2\pi) \right] \right\}.$$

where $\mathbb{I}$ is an indicator function and f is the probability density function of a multivariate normal.

Expectation-maximization in R

Expectation-maximization clustering probabilistically assigns data to different clusters. This is sometimes called “soft-clustering” (as oppossed to “hard-clustering” in which data only belongs to one cluster). I fairly tall person may be 55% likely to be a “man” and 45% likely to be a woman. This property may be useful in data which may share classes. For example, a song may be mostly “country” but a little bit “rock and roll.”

R has the Mclust function from the mclust library to provide the estimating the parameters in a statistical model using the Expectation-maximization (EM) algorthm.

We’ll go over a simple example, but Mclust has many tools normal mixture modeling using the EM algorithm ,for model-based clustering, classification, for density estimation, and Bayesian approaches to maximum liklihood estimation (Baysian probability will be covered in a future module).

mclust process

Given a model (a mixture Guassians, Binomial, etc.), we have the following parameters:

- X: This is a set of observed data. (Doesn’t change)
- Z: This is a set of estimates for unobserved values
- T: This is a set of unknown parameters for our model

The expectation-maximization steps:

- Initialize the unknown parameters T to random values.
- Compute the best fit for the missing values Z using the existing parameter values.
- Use the best fit missing values Z to generate a better estimate for the unknown parameters T

Iterate until we have a convergence, usually when Z and T don’t change much or after a fixed number of steps.

mclust Model-Based Clustering usage

Usage

Mclust(data, G = NULL, modelNames = NULL, prior = NULL, control = emControl(), initialization = NULL, warn = FALSE, …)

Arguments

- data - A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- G - An integer vector specifying the numbers of mixture components (clusters) for which the BIC is to be calculated. The default is G=1:9.
- modelNames - A vector of character strings indicating the models to be fitted in the EM phase of clustering. The default is:
    - for univariate data c(“E”, “V”)
    - for multivariate data (n > d) mclust.options(“emModelNames”)
    - for multivariate data (n <= d) the spherical and diagonal models c(“EII”, “VII”, “EEI”, “EVI”, “VEI”, “VVI”)

The help file for mclustModelNames (http://finzi.psych.upenn.edu/R/library/mclust/html/mclustModelNames.html) describes the available models.

- prior - The default assumes no prior, but this argument allows specification of a conjugate prior on the means and variances through the function priorControl.
- control - A list of control parameters for EM. The defaults are set by the call emControl().
- initialization - A list containing zero or more of the following components:
    - hcPairs - A matrix of merge pairs for hierarchical clustering such as produced by function hc. For multivariate data, the default is to compute a hierarchical clustering tree by applying function hc with modelName = “VVV” to the data or a subset as indicated by the subset argument. The hierarchical clustering results are to start EM. For univariate data, the default is to use quantiles to start EM.
    - subset - A logical or numeric vector specifying a subset of the data to be used in the initial hierarchical clustering phase.
    - noise - A logical or numeric vector indicating an initial guess as to which observations are noise in the data. If numeric the entries should correspond to row indexes of the data. If supplied, a noise term will be added to the model in the estimation.
- warn - A logical value indicating whether or not certain warnings (usually related to singularity) should be issued. The default is to suppress these warnings.

… Catches unused arguments in indirect or list calls via do.call.

Value

An object of class “Mclust” providing the optimal (according to BIC) mixture model estimation.

The details of the output components are as follows:

- call - The matched call
- data - The input data matrix.
- modelName - A character string denoting the model at which the optimal BIC occurs.
- n - The number of observations in the data.
- d - The dimension of the data.
- G - The optimal number of mixture components.
- BIC - All BIC values.
- bic - Optimal BIC value.
- loglik - The loglikelihood corresponding to the optimal BIC.
- df - The number of estimated parameters.
- hypvol - The hypervolume parameter for the noise component if required, otherwise set to NULL (see hypvol).
- parameters - A list with the following components:
    - pro - A vector whose kth component is the mixing proportion for the kth component of the mixture model. If missing, equal proportions are assumed.
    - mean - The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.
    - variance - A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.
- z - A matrix whose [i,k]th entry is the probability that observation i in the test data belongs to the kth class.
- classification - The classification corresponding to z, i.e. map(z).
- uncertainty - The uncertainty associated with the classification.

Bayesian information criterion (BIC)

The Bayesian information criterion (BIC) (https://en.wikipedia.org/wiki/Bayesian_information_criterion) is used my mclust with is a test used to assess the fit of a model, The Bayesian information criterion (BIC) or Schwarz criterion (also SBC, SBIC) is a criterion for model selection among a finite set of models; the model with the lowest BIC is preferred. It is based, in part, on the likelihood function and it It is similar to the Akaike information criterion (AIC) (https://en.wikipedia.org/wiki/Akaike_information_criterion) which is founded on information theory: it offers a relative estimate of the information lost when a given model is used to represent the process that generates the data. In doing so, it deals with the trade-off between the goodness of fit of the model and the complexity of the model.

When fitting models, it is possible to increase the likelihood by adding parameters, but doing so may result in overfitting. Both BIC and AIC try to resolve this problem by introducing a penalty term for the number of parameters in the model; the penalty term is larger in BIC than in AIC. The BIC was developed by Gideon E. Schwarz and published in a 1978 paper, where he gave a Bayesian argument for adopting it.

mclust also uses a Marginal likelihood (https://en.wikipedia.org/wiki/Marginal_likelihood) called the Integrated Complete X Likelihood to assess classification fits.

mclust example with diabetes data

EM clustering with diabetes data using mclust. Note that the summary command with an mclust object generates:

- log.likelihood: This is the log likelihood of the BIC value
- n: This is the number of X points
- df: This is the degrees of freedom
- BIC: This is the Bayesian information criteria; low is good
- ICL: Integrated Complete X Likelihood-a classification version of the BIC.

```
data(diabetes)
head(diabetes)
```

```
##    class glucose insulin sspg
## 1 Normal      80     356  124
## 2 Normal      97     289  117
## 3 Normal     105     319  143
## 4 Normal      90     356  199
## 5 Normal      90     323  240
## 6 Normal      86     381  157
```

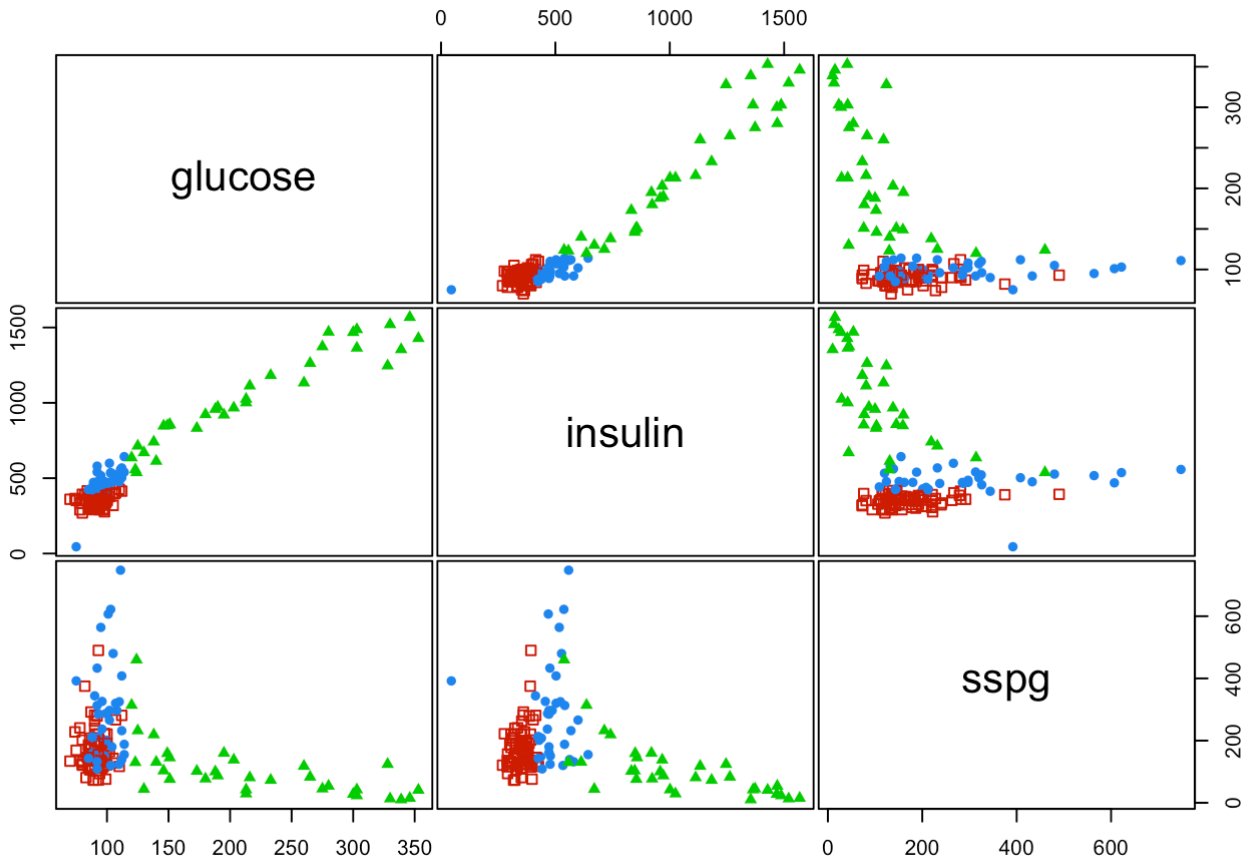
```
class.d = diabetes$class
table(class.d)
```

```
## class.d
## Chemical   Normal    Overt
##       36       76       33
```

```
X = diabetes[,-1]
head(X)
```

```
##   glucose insulin sspg
## 1      80     356  124
## 2      97     289  117
## 3     105     319  143
## 4      90     356  199
## 5      90     323  240
## 6      86     381  157
```

```
clPairs(X, class.d)
```




```
fit <- Mclust(X)
fit
```

```
## 'Mclust' model object:
##  best model: ellipsoidal, varying volume, shape, and orientation (VVV) with 3 compon
ents
```

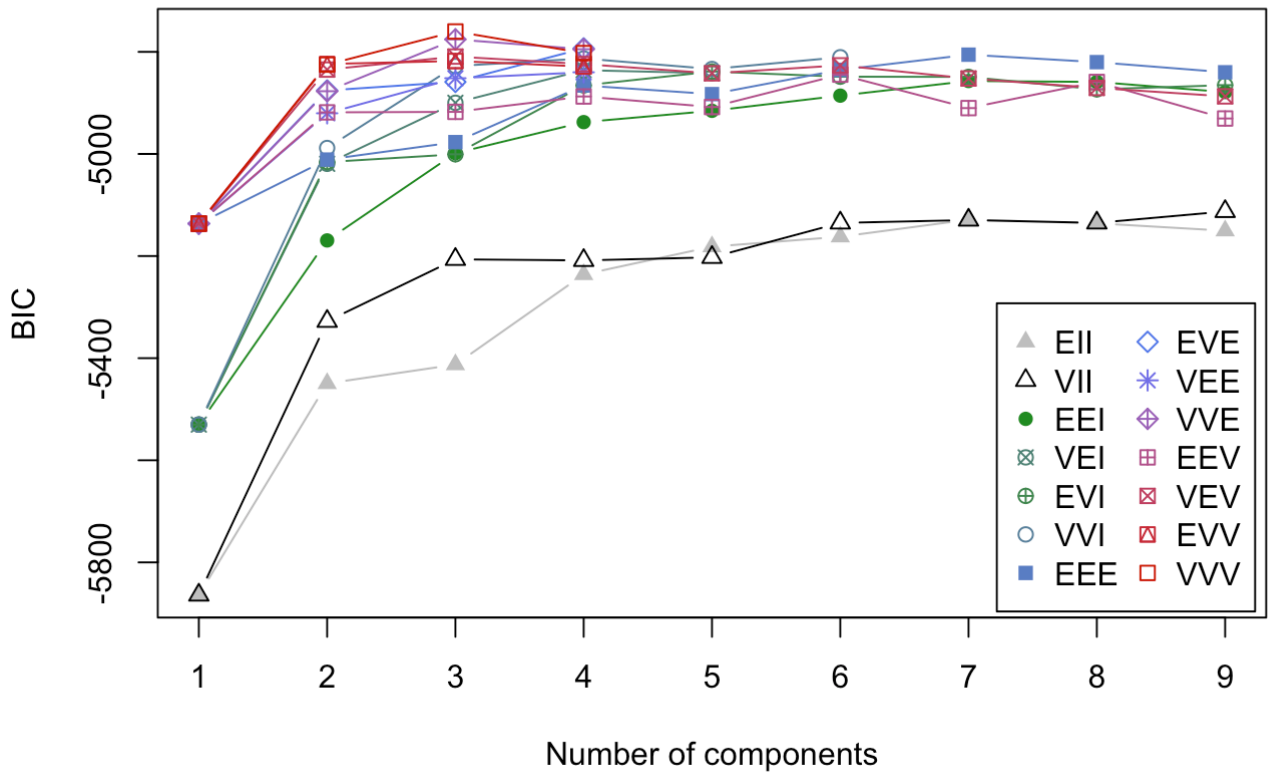
```
summary(fit)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model with 3 compon
ents:
##
##  log.likelihood   n df       BIC       ICL
##       -2307.883 145 29 -4760.091 -4776.086
##
## Clustering table:
##  1  2  3
## 82 33 30
```

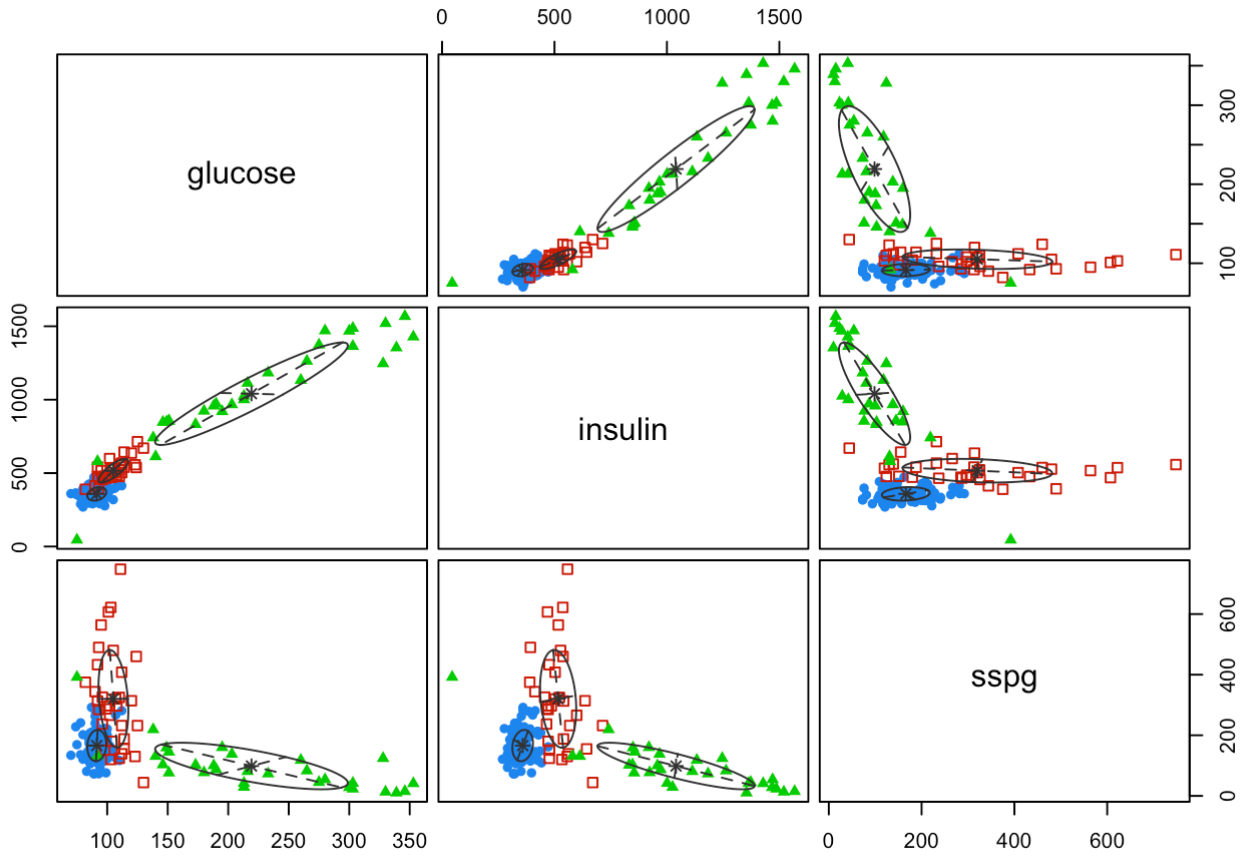
The plot command for EM produces the following four plots:

- The BIC values used for choosing the number of clusters
- A plot of the clustering
- A plot of the classification uncertainty
- The orbital plot of clusters

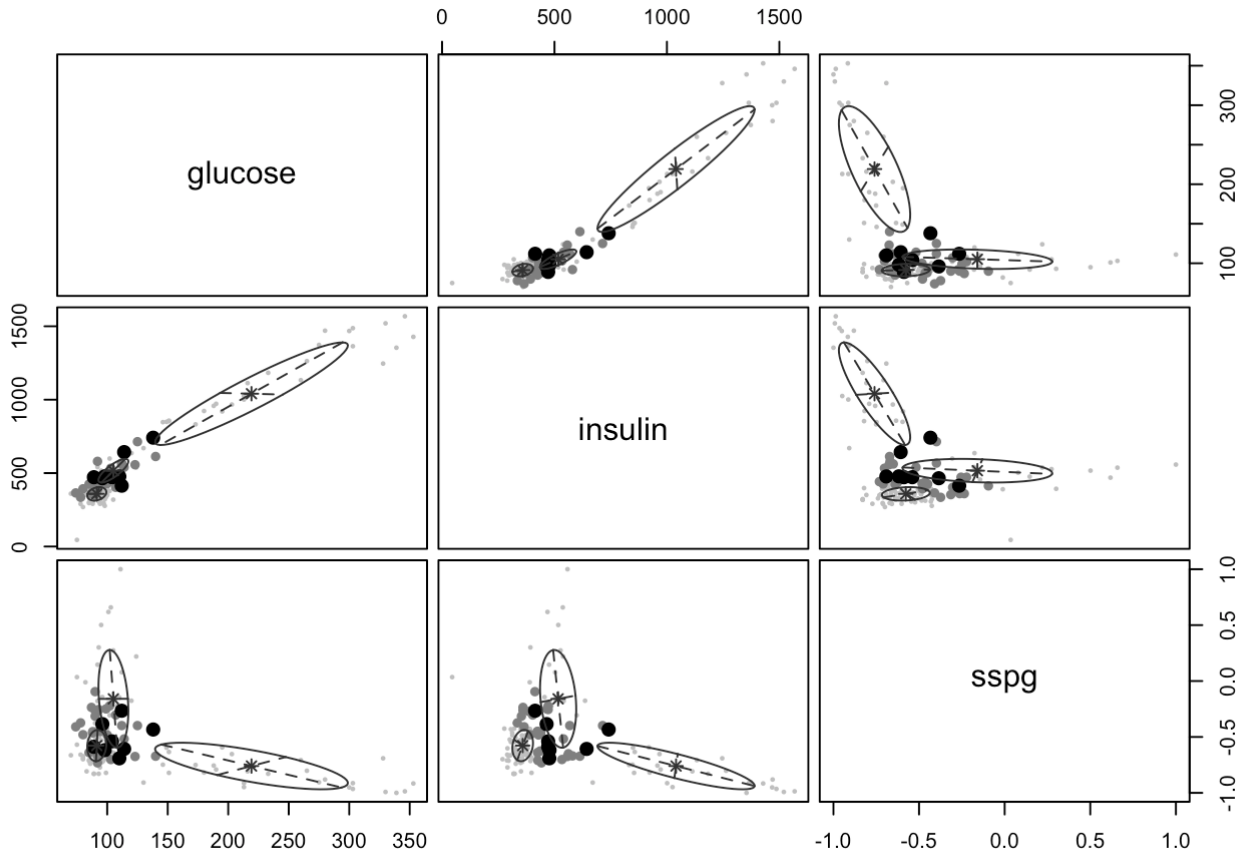
```
# 1: BIC
plot(fit, what = "BIC")
```



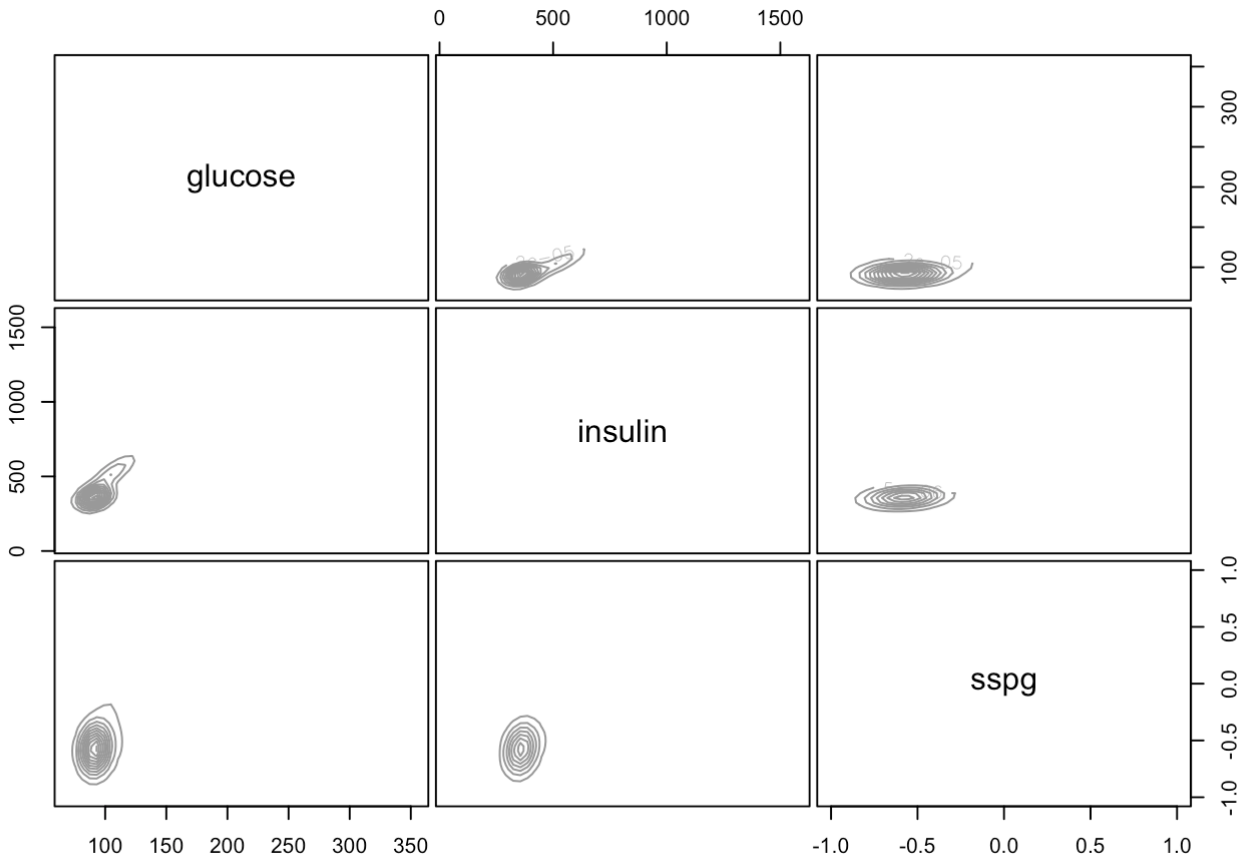

```
#table(class, fit$BIC)
#2: classification
plot(fit, what = "classification")
```




```
#table(class, fit$classification)
# 3: uncertainty
plot(fit, what = "uncertainty")
```




```
#table(class, fit$uncertainty)
# 4: density
plot(fit, what = "density")
```



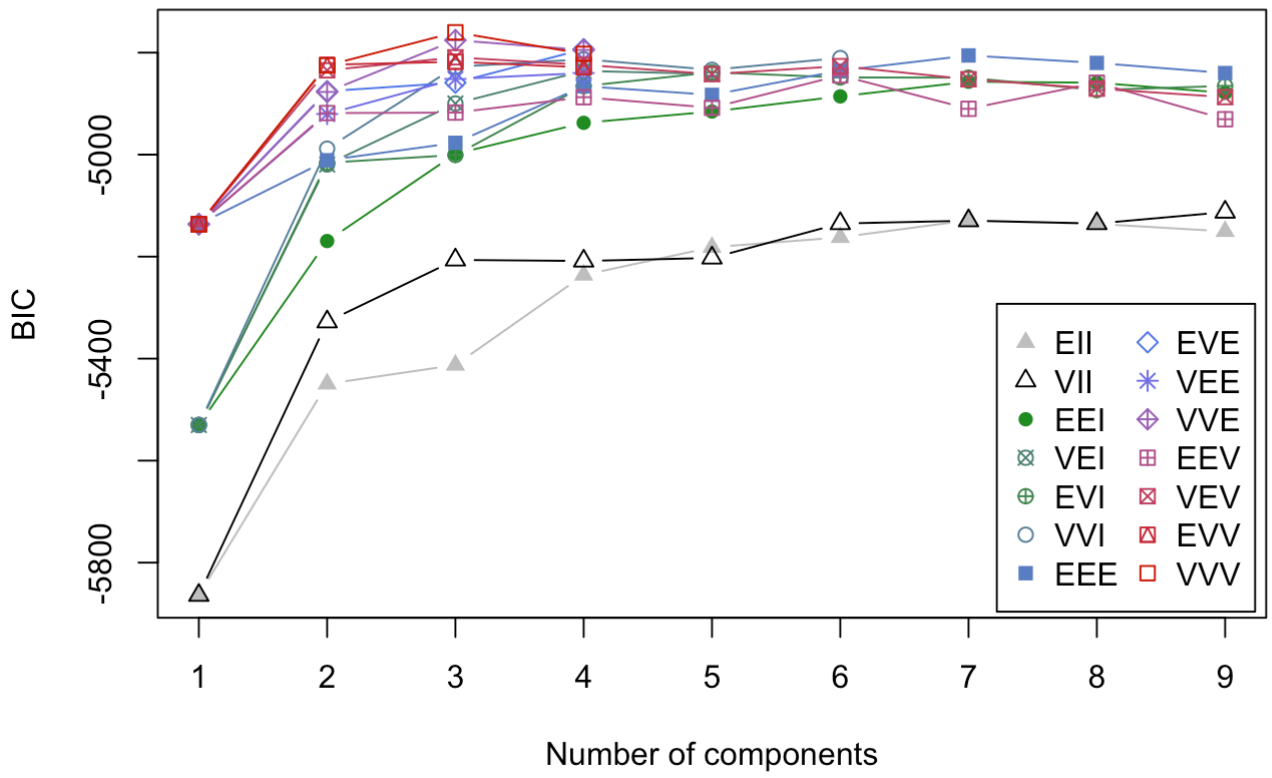

```
#table(class, fit$density)
```

The mclustBIC (http://svitsrv25.epfl.ch/R-doc/library/mclust/html/mclustBIC.html) from mclust uses BIC for EM initialized by model-based hierarchical clustering for parameterized Gaussian mixture models.

```
BIC = mclustBIC(X)
summary(BIC)
```

```
## Best BIC values:
##              VVV,3       VVE,3       EVE,4
## BIC      -4760.091 -4775.53693 -4793.26143
## BIC diff     0.000   -15.44628   -33.17079
```

```
plot(BIC) # Only BIC plot
```



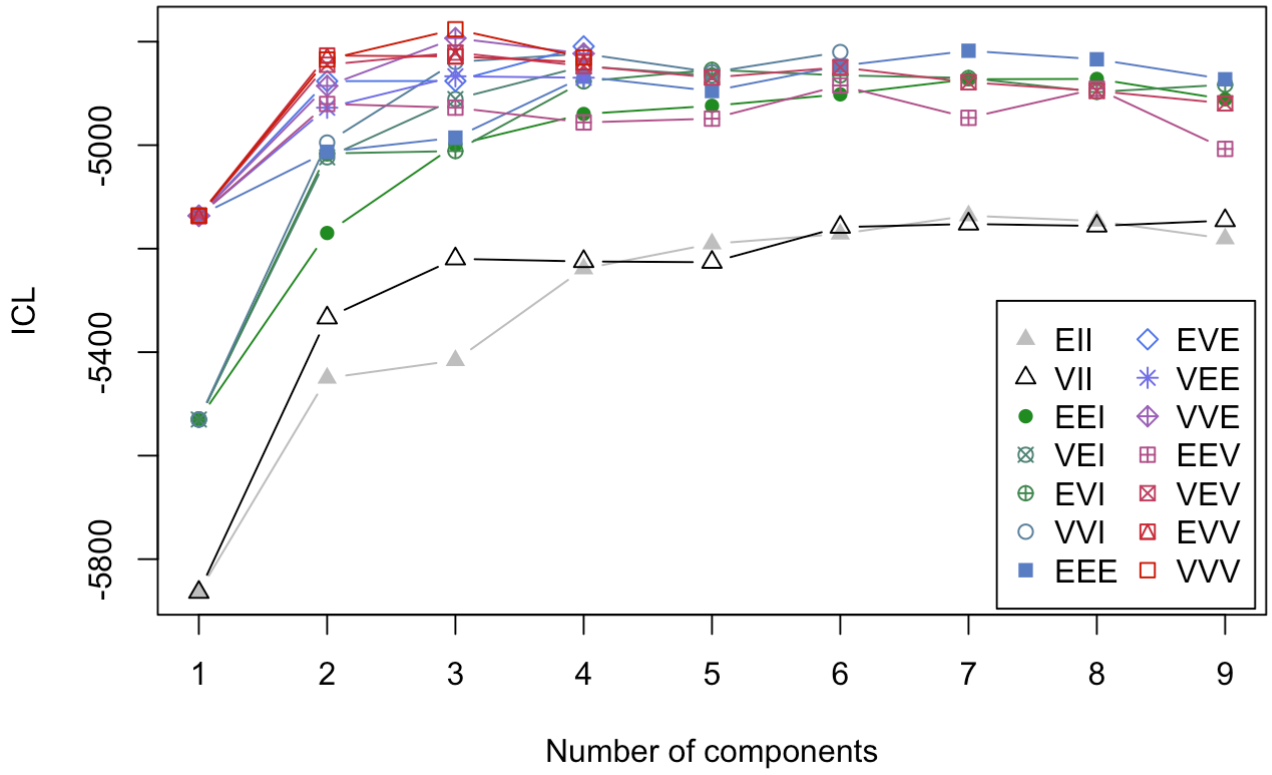

```
# plot(BIC,what = "BIC")
# The following just get BIC plot: plot(BIC,what = "classification"), plot(BIC,what =
 "uncertainty"), plot(BIC,what = "density")
```

mclustICL (http://finzi.psych.upenn.edu/R/library/mclust/html/mclustICL.html) from mclust uses ICL (Integrated Complete-data Likelihood) for parameterized Gaussian mixture models fitted by EM algorithm initialized by model-based hierarchical clustering.

```
ICL = mclustICL(X)
summary(ICL)
```

```
## Best ICL values:
##              VVV,3       VVE,3       EVE,4
## ICL      -4776.086 -4793.27143 -4809.16868
## ICL diff     0.000   -17.18553   -33.08278
```

```
plot(ICL)  # Only ICL plot
```



Expectation-maximization in R by hand

Suppose you have two coins. The probability of observing a head with the first coin is p and the second coin is q. We don't observe the flips, only the resulting sequence of heads or tails. Let $X_i \in 0, 1$ be the result of heads or tails at flip i.

Let $C_i \in 0, 1$ represent which coin was used. That is, $C = 0$ the first coin whith probability p was used and $C = 1$ the second coin whith probability q was used.

Since either the first or second coin was used for flip i, the probability of the result at flip i, is:

$$P(X_i) = P(X_i|C_i = 0)P(C_i = 0) + P(X_i|C_i = 1)P(C_i = 1)$$

Note that if the probability of using coin one doesn't change during the sequence of flips then $P(C_i = 0)$ is just $(C = 0)$ and the probability of using coin two is $1 - P(C = 0)$.

Note also that if we wanted to model a sequence of observed heights using the Galton data directly we would use a mixture of two Gaussian distributions repsesenting the height distributions by gender and the probability of getting "coin one or two" (i.e. a male or female) would be roughly 50%.

```
#------------------------------ Expectation  ----------------------------

expectation <- function(sample,p,a,b)
  {
  p_expectation <- (p*dbinom(sample,1,a)) / ( p*dbinom(sample,1,a) + (1-p)*dbinom(sampl
e,1,b) )
  return(p_expectation)
}


#------------------------------ Maximization   ------------------------------

maximization <- function(sample,epart){

  # estimate p

  p_temp <- mean(epart)

  # estimate a and b

  a_temp <- sum(sample*epart) / sum(epart)
  b_temp <- sum(sample*(1-epart)) / sum(1-epart)

  list(p_temp,a_temp,b_temp)
}



#--------------------------- Expectation Maximization Algorithm  --------------------
----

EM <- function(sample,p_inits,a_inits,b_inits,maxit=1000,tol=1e-6)
  {
  # Estimation of parameter(Initial)
  flag <- 0
  p_cur <- p_inits; a_cur <- a_inits; b_cur <- b_inits

  # Iterate between expectation and maximization parts

  for(i in 1:maxit){
    cur <- c(p_cur,a_cur,b_cur)
    new <- maximization(sample,expectation(sample, p_cur, a_cur, b_cur))
    p_new <- new[[1]]; a_new <- new[[2]]; b_new <- new[[3]]
    new_step <- c(p_new,a_new,b_new)

    # Stop iteration if the difference between the current and new estimates is less th
an a tolerance level
    if( all(abs(cur - new_step) < tol) ){ flag <- 1; break}


    # Otherwise continue iteration
    p_cur <- p_new; a_cur <- a_new; b_cur <- b_new
  }
```

```
  if(!flag) warning("Didn't converge\n")

  list(p_cur, a_cur, b_cur)
}



#------------------------------ Calculating Information matrix -----------------------
----

Info.Mat.function <- function(sample, p.est, a.est, b.est){
  expectation.est <- expectation(sample,p.est, a.est, b.est)
  info.mat <- matrix(rep(0,9),3,3)
  info.mat[1,1] <- - sum(expectation.est)/(p.est^2)  - sum((1-expectation.est))/((1-p.e
st)^2)
  info.mat[2,2] <- - sum(expectation.est*sample)/(a.est^2) - sum(expectation.est*(1-sam
ple))/((1-a.est)^2)
  info.mat[3,3] <- - sum((1-expectation.est)*sample)/(b.est^2) - sum((1-expectation.est)*
(1-sample))/((1-b.est)^2)
  return(-info.mat)
}


#------------------ Now Generate sample data -------------------------------

n <- 10000
p_true <- 0.85 # prob of using first coin
a_true <-  0.50 # the first coin has P(heads) = 0.50
b_true <-  0.70 # the second coin has P(heads) = 0.70
true <- c(p_true,a_true,b_true)
u <- ifelse(runif(n)<p_true, rbinom(n,1,a_true),rbinom(n,1,b_true))


# Set parameter estimates
p_init = 0.70; a_init = 0.70; b_init = 0.60


#--------------------Return EM Algorithm function and calculate Confidence Interval----
-------------------------

output <- EM(u,p_init,a_init,b_init)


# Confidence Intervals
sd.out <- sqrt(diag(solve(Info.Mat.function(u,output[[1]],output[[2]],output[[3]]))))
data.frame("Truth" = true, "EM Estimate" = unlist(output), "Lower CI" = unlist(output)
- qnorm(.975)*sd.out, "Upper CI" = unlist(output) + qnorm(.975)*sd.out)
```

```
##   Truth EM.Estimate  Lower.CI  Upper.CI
## 1  0.85   0.6863229 0.6772290 0.6954169
## 2  0.50   0.5605037 0.5487615 0.5722460
## 3  0.70   0.4505061 0.4330945 0.4679177
```

Assingment

- Use the same dataset you use for M04 Lesson 02 for the partition (k-means,PAM) and hierarchical clustering from the he the UC Irvine Machine Learning Repository (https://archive.ics.uci.edu/ml/)
- Cluster some of your data using EM based clustering.

Answer the following questions:
* How did you choose a model for EM?
* Evaluate the model performance.

- Cluster some of your data using EM based clustering that you also used for k-means,PAM. and hierarchical clustering.

Answer the following questions:
How do the clustering appaoches compare on the same data?

Write up your report as an .Rmd file.

Resources

Data Mining Algorithms In R/Clustering/Expectation Maximization (EM)
[https://en.wikibooks.org/wiki/Data_Mining_Algorithms_In_R/Clustering/Expectation_Maximization_(EM)
(https://en.wikibooks.org/wiki/Data_Mining_Algorithms_In_R/Clustering/Expectation_Maximization_(EM))]

References

- C. Fraley, A. E. Raftery, T. B. Murphy and L. Scrucca (2012). mclust Version 4 for R: Normal Mixture Modeling for Model-Based Clustering, Classification, and Density Estimation. Technical Report No. 597, Department of Statistics, University of Washington.
- C. Fraley and A. E. Raftery (2002). Model-based clustering, discriminant analysis, and density estimation. Journal of the American Statistical Association 97:611:631.
- C. Fraley and A. E. Raftery (2005, revised 2009). Bayesian regularization for normal mixture estimation and model-based clustering. Technical Report, Department of Statistics, University of Washington.
- C. Fraley and A. E. Raftery (2007). Bayesian regularization for normal mixture estimation and model-based clustering. Journal of Classification 24:155-181.