MixAll: Learning mixture models

Serge Iovleff
University Lille 1

Abstract

The MixALL package can also be used in order to learn mixture models when the labels class are known. This short vignette assume that you have already read the vignette "Clustering With MixAll" (Iovleff (2016)).

Keywords: R, C++, STK++, Learning, missing values.

1. Introduction

It is possible to perform supervised learning with MixAll when the labels of the individuals are known. Let us recall the notations defined in the Iovleff (2016) vignette. $\mathcal{X}$ denote an arbitrary measurable space, $\mathcal{Z} = \{1, \ldots, K\}$ is the label set and $(\mathbf{x}, \mathbf{z}) = \{(\mathbf{x}_1, \mathbf{z}_1), \ldots, (\mathbf{x}_n, \mathbf{z}_n)\}$ represents $n$ independent vectors in $\mathcal{X} \times \mathcal{Z}$ such that each $\Pr(\mathbf{z}_i = k) = p_k$ and such that conditionnaly to $\mathbf{z}_i = k$, $\mathbf{x}_i$ arises from a probability distribution with density

$$h(\mathbf{x}_i | \lambda_k, \alpha)$$

parameterized by $\lambda_k$ and $\alpha$.

Given the matrix of obervation $\mathbf{x}$ and the vector of labels $\mathbf{z}$, the learning methods will estimate the unknown parameters $\lambda_k$ and $\alpha$. The learning R functions will also return a S4 class instance containing the posterior probabilities membership $t_{ik}$ and the predicted class membership $\mathbf{z}_i$ of each individuals ($\mathbf{ziFit}$ data member class).

2. Learning with MixAll

Learning analysis can be performed with the functions

1. `learnDiagGaussian` for diagonal Gaussian mixture models,
2. `learnCategorical` for Categorical mixture models,
3. `learnPoisson` for Poisson mixture models,
4. `learnGamma` for gamma mixture models,
5. `learnMixedData` for MixedData mixture models.

These functions have a common set of parameters with default values given in the table 1.
MixAll: Learning data

<table>
<thead>
<tr>
<th>Input Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>A matrix (or a list of matrix for mixed data) with the data to learn.</td>
</tr>
<tr>
<td>labels</td>
<td>A vector with the classes of each individuals. Values must be between 1 and K.</td>
</tr>
<tr>
<td>models</td>
<td>A vector with the models to adjust to each data set in case of mixed data, or a set of models to try to adjust. Default is <code>cluster*Names()</code> where '*' stands for DiagGaussian, Poisson, Gamma or Categorical.</td>
</tr>
<tr>
<td>prop</td>
<td>A vector of size K with the proportions of each class. If <code>prop</code> is NULL then the proportions are computed using the empirical distribution of the <code>labels</code>.</td>
</tr>
<tr>
<td>algo</td>
<td>A string defining the algorithm to use for the missing values. Possible values &quot;impute&quot;, &quot;simul&quot;.</td>
</tr>
<tr>
<td>nbIter</td>
<td>maximal number of iteration to perform. Default value is 100. Note that if there is no missing values, it should be 1.</td>
</tr>
<tr>
<td>epsilon</td>
<td>threshold to use in order to stop the iterations (not used by the &quot;simul&quot; algorithm). Default value 1e-08.</td>
</tr>
<tr>
<td>criterion</td>
<td>A string defining the model selection criterion to use. The best model is the one with the lowest criterion value. Possible values: &quot;AIC&quot;, &quot;BIC&quot;, &quot;ICL&quot;. Default is &quot;ICL&quot;.</td>
</tr>
<tr>
<td>nbCore</td>
<td>An integer defining the number of processor to use. Default is 1, 0 for all cores.</td>
</tr>
</tbody>
</table>

Table 1: List of common parameters of the learning functions.

2.1. Learning Multivariate (diagonal) Gaussian Mixture Models

Multivariate Gaussian mixture models (without correlations) can be learned using the `learnDiagGaussian` function. We illustrate this function with the well known geyser data set (Azzalini and Bowman (1990), Härdle (1991)).

The model selected by ICL criteria among the models with fixed proportion is Gaussian with the same standard deviations among the groups and the variables.

```r
> data(iris);
> x <- as.matrix(iris[,1:4]); z <- as.vector(iris[,5]); n <- nrow(x); p <- ncol(x);
> indexes <- matrix(c(round(runif(5,1,n)), round(runif(5,1,p))), ncol=2);
> cbind(indexes, x[indexes]) # store true values
     [,1] [,2] [,3]
[1,]   65   3  3.6
[2,]   32   2  3.4
[3,]   17   2  3.9
[4,]   13   3  1.4
[5,]   51   2  3.2
```
x[indexes] <- NA; # and set them as missing
model <- learnDiagGaussian( data=x, labels = z
+ , models = clusterDiagGaussianNames(prop = "equal"))
summary(model)

************************************************
* model name = gaussian_p_s
* nbSample = 150
* nbCluster = 3
* lnLikelihood = -1019.182
* nbFreeParameter= 70
* criterion name = ICL
* criterion value= 2396.021
************************************************

Estimated missing values and comparison between the true membership labels class and the fitted membership labels class are given below.

> # get estimated missing values
> missingValues(model)

    row col   value
 1     17  2 3.223319
 2     32  2 3.379443
 3     51  2 2.899486
 4     13  3 1.382116
 5     65  3 4.384769

> # compare predictions with true values
> table(model@zi,model@ziFit)

   0 1 2
0 50 0 0
1  0 47 3
2  0  4 46

> plot(model)
2.2. Learning Multivariate categorical Mixture Models

Categorical (nominal) data can be learned using the `learnCategorical` function. We illustrate this function with the birds data set.

```r
> data(birds)
> ## add 10 missing values
> x <- as.matrix(birds[,2:5]); z <- as.vector(birds[,1]); n <- nrow(x); p <- ncol(x);
> indexes <- matrix(c(round(runif(5,1,n)), round(runif(5,1,p))), ncol=2)
> cbind(indexes, x[indexes]) # print true values

[,1] [,2] [,3]
[1,] "49" "1" "pronounced"
[2,] "7" "2" "dotted"
[3,] "19" "2" "dotted"
[4,] "28" "3" "black & white"
[5,] "65" "1" "pronounced"

> x[indexes] <- NA;        # set them as missing
> model <- learnCategorical( data=x, labels=z)
```
+ models = clusterCategoricalNames(prop = "equal")
+ algo="simul", nbIter = 2
>

> summary(model)

*****************************************************************************
* model name = categorical_p_pk
*****************************************************************************
* nbModalities = 4
*****************************************************************************
* levels =
  [1] "none , poor pronounced, pronounced , very pronounced"
  [2] "dotted, none "
  [3] "black , black & white, white   "
  [4] "few , many, none"
*****************************************************************************
* nbSample = 69
* nbCluster = 2
* lnLikelihood = -461.5516
* nbFreeParameter= 23
* criterion name = ICL
* criterion value= 1109.044
*****************************************************************************

Estimated missing values and comparison between the true membership labels class and the fitted membership labels class are given below.

> # get estimated missing values
> missingValues(model)

    row col value
   1 49 1 3
   2 65 1 3
   3 7 2 2
   4 19 2 2
   5 28 3 3

> # compare predictions
> table(model@zi,model@ziFit)

          0 1
0 26 10
1 18 15

> plot(model)
2.3. Learning Multivariate Gamma Mixture Models

Gamma data can be learned using the `learnGamma` function. We illustrate this function with the iris data set.

```r
> data(iris)
> x <- as.matrix(iris[,1:4]); z <- as.vector(iris[,5]); n <- nrow(x); p <- ncol(x);
> indexes <- matrix(c(round(runif(5,1,n)), round(runif(5,1,p))), ncol=2);
> cbind(indexes, x[indexes]) # print true values

[,1] [,2] [,3]
[1,] 8 3 1.5
[2,] 38 2 3.6
[3,] 143 3 5.1
[4,] 70 3 3.9
[5,] 4 1 4.6

> x[indexes] <- NA; # set them as missing
> model <- learnGamma( data=x, labels= z
  + , models = clusterGammaNames(prop = "equal")
```

Hist of dim 1

---

**MixAll: Learning data**
> summary(model)

****************************************
* model name = gamma_p_ajk_b
* nbSample = 150
* nbCluster = 3
* lnLikelihood = -19303.11
* nbFreeParameter = 142
* criterion name = ICL
* criterion value = 39318.69
****************************************

> # get estimated missing values
> missingValues(model)

    row col value
1     4   1 4.196168
2    38   2 3.813789
3     8   3 2.099860
4    70   3 4.326391
5   143   3 4.029982

Estimated missing values and comparison between the true membership labels class and the fitted membership labels class are given below.

> # compare predictions
> table(model@zi,model@ziFit)

        0 1 2
0 50 0 0
1  0 48 2
2  0 46

> plot(model)
2.4. Learning Multivariate Poisson Models

Poisson data (count data) can be learned using the `learnPoisson` function. We illustrate this function with the debTrivedi data set.

```r
> data(DebTrivedi)
> x <- DebTrivedi[, c(1, 6, 8, 15)]; z <- DebTrivedi$medicaid; n <- nrow(x); p <- ncol(x);
> indexes <- matrix(c(round(runif(5,1,n)), round(runif(5,1,p))), ncol=2);
> cbind(indexes, x[indexes])  # print true values

     [,1] [,2] [,3]
[1,]  3585  1  0
[2,]  596  1 18
[3,] 1047  1  8
[4,]  920  3  1
[5,] 3308  4 12

> x[indexes] <- NA; # set them as missing
> model <- learnPoisson( data=x, labels=z
+                         , models = clusterPoissonNames(prop = "equal")
```

Hist of Sepal.Length

Hist of Sepal.Width

Hist of Petal.Length

Hist of Petal.Width

Hist of Sepal.Length

Hist of Sepal.Width

Hist of Petal.Length
+ algo="simul", nbIter = 2, epsilon = 1e-08 + )

> summary(model)

****************************************
* model name = poisson_p_ljk
* nbSample = 4406
* nbCluster = 2
* lnLikelihood = -161229.8
* nbFreeParameter= 17
* criterion name = ICL
* criterion value= 324538.1
****************************************

> # get estimated missing values
> missingValues(model)

<table>
<thead>
<tr>
<th>row</th>
<th>col</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>596</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>1047</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>3585</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>920</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>3308</td>
<td>12</td>
</tr>
</tbody>
</table>

Estimated missing values and comparison between the true membership labels class and the fitted membership labels class are given below.

> # compare predictions
> table(model@zi,model@ziFit)

<table>
<thead>
<tr>
<th>0 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>3730 274</td>
</tr>
<tr>
<td>280 122</td>
</tr>
</tbody>
</table>

> plot(model)
2.5. Learning Mixed data sets

Mixed data sets can be learned using the \texttt{learnMixedData} function. The original mixed data set has to be split in multiple homogeneous data sets and each one associated to a mixture model name.

We illustrate this function with the HeartDisease data set (Detrano \textit{et al.} (1989)).

\begin{verbatim}
> data(HeartDisease.cat)
> data(HeartDisease.cont)
> data(HeartDisease.target)
> ldata = list(HeartDisease.cat, HeartDisease.cont);
> models = c("categorical_pk_pjk","gaussian_pk_sjk")
> z<-HeartDisease.target[[1]];
> model <- learnMixedData(ldata, models, z, algo="simul", nbIter=2)
> summary(model)
\end{verbatim}

******************************************************************************
* model name = categorical_pk_pjk
* model name = gaussian_pk_sjk
* nbSample = 303
* nbCluster = 5
* lnLikelihood = -7525.496
* nbFreeParameter= 129
* criterion name = ICL
* criterion value= 16163.46

************************************************************************************

Estimated missing values and comparison between the true membership labels class and the fitted membership labels class are given below.

>`# get estimated missing values`
>`missingValues(model)`

```
[[1]]
row col value
[1,] 167  7  1
[2,] 193  7  1
[3,] 288  7  1
[4,] 303  7  1
[5,]  88  8  1
[6,] 267  8  3

[[2]]
row col value
```

>`# compare predictions`
>`table(model@zi,model@ziFit)`

```
   0  1  2  3  4
0 143 10  3  7  1
1  2 18  9  4  2
2  1  9 15  9  2
3  1  7  7 18  2
4  0  4  2  4  3
```

>`plot(model)`
MixAll: Learning data

References


Affiliation:

Serge Iovleff
Univ. Lille 1, CNRS U.M.R. 8524, Inria Lille Nord Europe
59655 Villeneuve d’Ascq Cedex, France
E-mail: Serge.Iovleff@stkpp.org
URL: http://www.stkpp.org