Statistics and Classification on Genomic Data

julien.chiquet@gmail.com

Bioinformatics Summer School - Angers, 2016

http://julien.cremeriefamily.info/bioinfo_angers.html

This practical aims to provide an overview of a series of statistical methods now routinely used to process genomic data. The statistical tasks at play will concern both supervised and unsupervised classification problems. The classical Golub data set is used to illustrate these methods.

Note : you can form small and balanced groups of students to work.

1 Preliminaries

1.1 The Golub (Leukemia) data set

1.1.1 Description

Gene expression cancer data set (Golub et al.) of samples from human acute myeloid (AML) and acute lymphoblastic leukemias (ALL). 3571 expression values on 72 individuals.

- The binary response vector ($\mathbf{y}$) describes the type of leukemia (AML/ALL - 0/1).
- The matrix of predictors ($\mathbf{X}$) contains the log-transform gene expression levels of 3571 genes.

1.1.2 Source

Golub T.R et al. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531-537.

1.2 Loading the data

The data are part of the `spikeslab` package:

```
suppressMessages(library(spikeslab))
data(leukemia)
ls()
```

```
## [1] "leukemia"
```

You can also directly read it from the `.RData` file available on the web page.

```
load("data/leukemia.RData")
ls()
```

```
## [1] "expr"  "pheno"
```

1.3 Descriptive statistics

- Plot a normalized histogram + a density plot of the mean expression (respectively of the standard deviation) of the expression level of each individual. Comment.
- Have a look at the 'qqplot' vs. the normal distribution for the expression levels.
- Do a PCA analysis of the matrix of predictors. Plot the individual factor map for axes (1,2), (1,3), (2,3). Color the individual according to their status (AML/ALL). Comment

package: `FactoMineR`, with functions `PCA`, `plot.PCA`

1.4 Variable Screening by differential analysis

Depending of the performance of your computer, we may have to reduce the initial number of genes at play in order to perform your analysis comfortably (that is, by not waiting too much for `R`).

- For each gene, test the difference in the mean between the two groups (ALL/AML), with a t-test or a Wilcoxon test. You can also simply rank the genes by decreasing variance.
- Plot the histogram of the p-values before and after multiple testing correction (Benjamini-Hochberg, Bonferroni: function `p.adjust`).
- Only retain the genes with an adjusted p-value smaller than 0.1%.

If your computer cannot stand it, only retains the first thousand of genes.

2 Supervised classification of the samples

In this part, our goal is to select a small set of genes that do a reasonably good job for discriminating the AML/ALL samples. To this end, we rely on sparse logistic regression.

2.1 Main Analyses

- Why usual logistic regression would not run? (try it with the function `glm`).
- Randomly split your data in a training set and a test set. Use 2/3 of the sample for the training set.
- Adjust a logistic regression model regularized by a Lasso penalty ($\alpha = 1$)
- Adjust a logistic regression model regularized by an Elastic-net penalty. Use various values, say $\{.25, .5, .75, .95\}$ for the mixture parameter α.
- For each model (i.e. each value of α), choose the final estimator (that is, the final λ by cross-validation of the classification error.
- Compute the classification error on the test set and the rand index.
- Compare the genes selected by the differernt methods.
- Redo the all procedure by changin the training and test set. Comment.

package: `glmnet`, with functions `glmnet`, `cv.glmnet`, `predict.glmnet`, `plot.glmnet`, `plot.cv.glmnet`

2.2 Additional analyses

Use parallel computing to

- Estimate the classification error for the Lasso by resampling the training/test sets a hundred of times. Plot the boxplot of the estimated classification error.
- Select the more stable genes by subsampling the training set, a.k.a. perform stability selection.

package: `parallel`, with function `mclapply` - or `doMC`, with function `foreach`

Perform a simple unsupervised clustering *of the samples*

- Apply spectral clustering (see function of the web page) with 2 groups on the correlation matrix of the sample
- check adequation with the true classification by computing the adjusted Rand Index between the two classifications.

package: `mclust`, with function `adjustedRandIndex`

3 Unsupervised classification of the genes

In this part, we would like to regroup the predictors (i.e. the genes) together. The problem is that we do not have a clue on the classes we are looking for (contrary to the supervised problem, where there were two classes corresponding to AML/ALL).

Here, we do not even know the number of groups we are looking for. We rather speak of clusters than classes for unsupervised classification.

3.1 Main analysis

- Compute the correlation matrix between all the genes. Plot this matrix (into a `jpeg` file to save time !). You can use the image method of the package `Matrix`, with option `useRaster=TRUE`.
- Perform k-means clustering for a number of group of your choice (say, 15). Reorder the rows and columns of the correlation matrix accordingly. Redo the image plot. Comment.
- The BIC/ICL criterion for the k-means is defined by

  $$BIC(k) = -2 \log \ell(\mathbf{X}) + 2(kp + 1)\log(n),$$

  where k is the number of clusters and p the number of genes. The log-likelihood corresponds to a mixture of Gaussian distribution with spherical covariances, such that

  $$\log \ell(\mathbf{X}) = -\frac{1}{2}(np + np\log(WSS/p)) - n\log(k) + \frac{np}{2}\log(2\pi).$$

  The total within sum of squares WSS is sent back by the `kmeans` function. If you fail in implementing the BIC, you can use the function available on the website. Use this criterion to choose the number of cluster, by performing k-means clustering for a number of groups varying from 2 to 50. Redo the plot

3.2 Additional Analysis

Now that we have a clustering of the genes, we can take it into account in our original supervised classification problem:

- Create a matrix of expression for the "meta-genes" by computing the average expression in each cluster of gene for each sample. Use sparse logistic regression to select the most relevant clusters of genes.
- Apply the logistic group-Lasso (package `grpreg`) using the grouping obtained above on the genes. Compare the groups selected by this method with the meta genes selected by the Lasso.