Nathalie Villa-Vialaneix
Année 2015/2016

M1 in Economics and Economics and Statistics
Applied multivariate Analysis - Big data analytics
Final exam

General recommandations

The project must be returned as a PDF file with a separate R script (`.R` file). Examples of expected answers are provided at http://www.nathalievilla.org/doc/pdf/ex_answers_M1SE.pdf (PDF file) and http://www.nathalievilla.org/doc/R/ex_answers_M1SE.R (R script).

Exercice 1 Example exercise

This section is an example of the kind of answers expected from you. The typical solution is provided at http://www.nathalievilla.org/doc/pdf/ex_answers_M1SE.pdf (PDF file) and http://www.nathalievilla.org/doc/R/ex_answers_M1SE.R (R script). Do not include it in your own essay!

Using the public iris dataset (available with `data(iris)`), sample at random 50 observations and make a barplot to represent the distribution of the different Species in this dataset. Use the random seed `set.seed(1609)` at the begining of your script. The final document must contain a title, your names and a short description of the dataset. The answer to your question must be commented and you must provide the R script which allowed to obtain it.

Exercice 2 Presentation of the dataset

The final exam is based on the data described at http://archive.ics.uci.edu/ml/datasets/Online+News+Popularity. Download the zip file in "Data Folder" and uncompress it. It contains two files, one with the data set and the other file describind it. Write an introduction with a short description of the data.

Answer: The data set used in this final exam comes from the UCI Machine Learning Repository and is available at http://archive.ics.uci.edu/ml/datasets/Online+News+Popularity. It contains a description of 39,645 articles published by Mashable (http://www.mashable.com) and is issued from an analysis published as K. Fernandes, P. Vinagre and P. Cortez. A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News. Proceedings of the *17th EPIA 2015 - Portuguese Conference on Artificial Intelligence*, September, Coimbra, Portugal.

61 variables describing the articles which, among them, a target variable (to predict), which is the number of shares of this article in social networks (which is used as a measure of the popularity of the data set). The first two variables of the data set are non informative to explain this target variable because they are, respectively, the article URL and the number of day between the article publication and the data set acquisition. The other 58 variables are different descriptors of the article, which are all numeric.

. . . . . / 2

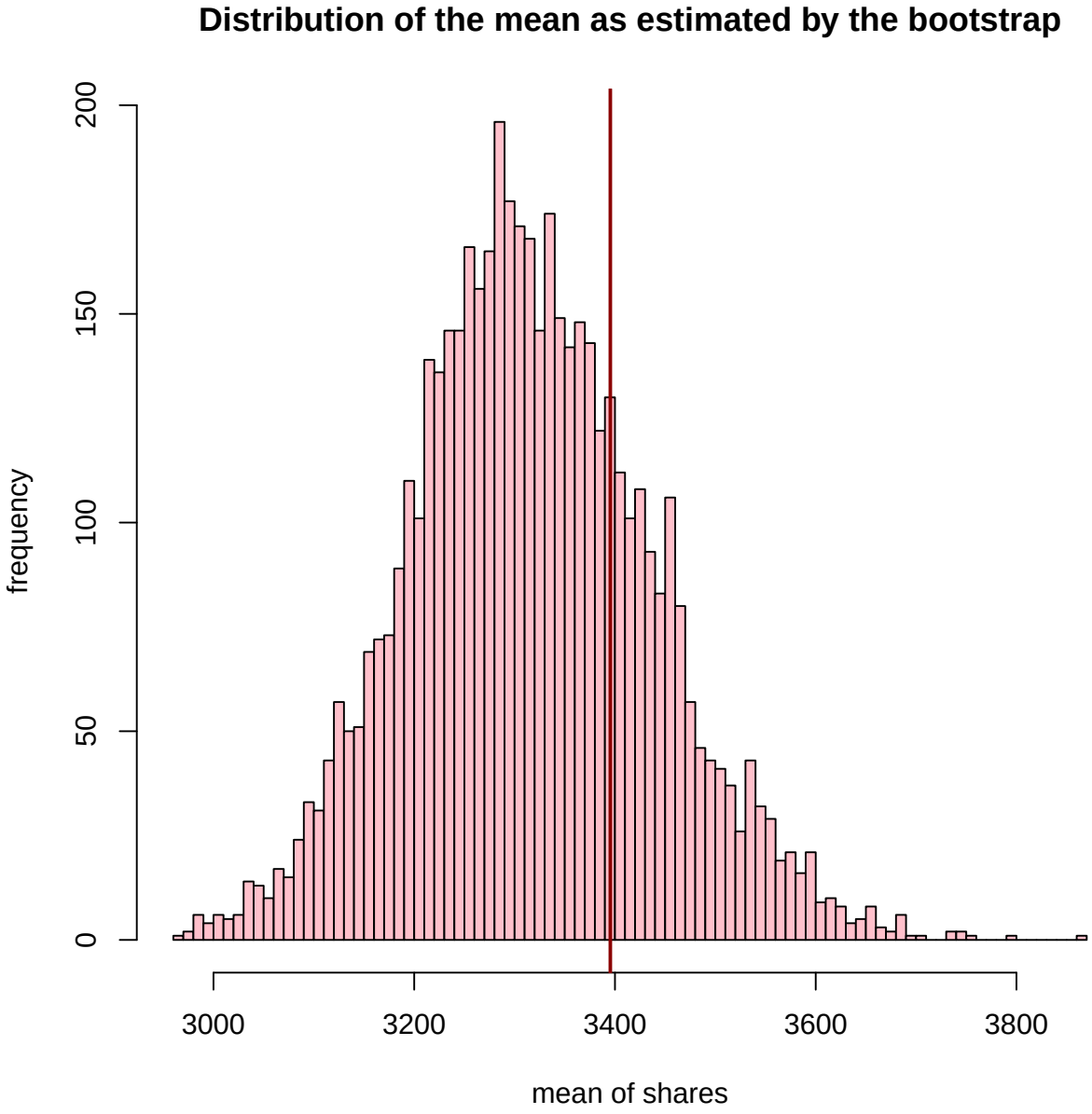
Exercice 3 Bootstrap

This section aims at giving an estimate of the mean of the number of shares using only a small subsample of the whole data set.

First import the data set in R and find the true mean of the number of shares. Then, take a sample of 5000 articles and use a bootstrap approach to provide a 95% confidence interval of the mean of the number of shares based on this small subsample. Finally, print a histogram of the mean distribution as estimated by the bootstrap approach with the true mean highlighted by a vertical line. Use 5000 bootstrap samples and start your script by setting the random seed by: `set.seed(1700)`.

Answer:

```r
set.seed(1700)
# 0. import data
df <- read.table("OnlineNewsPopularity/OnlineNewsPopularity.csv", sep=",",
                 quote="", stringsAsFactor = FALSE, header = TRUE)

# 1. mean of shares
mean_shares <- mean(df$shares)

# 2. extract a subsample with 5000 articles
sel_obs <- sample(1:nrow(df), 5000, replace = FALSE)
df_subsample <- df[sel_obs, ]

# 3. bootstrap IC for shares
## function to obtain mean from a bootstrap sample
bootstrap_mean <- function(a_sample, sub_indexes) {
  return(mean(a_sample[sub_indexes]))
}
## use it with boot
library(boot)
boot_mean <- boot(df_subsample$shares, bootstrap_mean, 5000)
## 95% IC
IC <- quantile(boot_mean$t, probs = c(0.025, 0.975))
IC

##     2.5%    97.5%
## 3091.287 3567.753

# 3. histogram of mean distribution
hist(boot_mean$t, xlab = "mean of shares", ylab = "frequency", breaks = 100,
     main = "Distribution of the mean as estimated by the bootstrap",
     col = "pink")
abline(v = mean_shares, col = "darkred", lwd = 2)
```



The true mean is equal to 3395.38 shares and a 95% confidence interval is [3091.29, 3567.75].

Import data set ..... / 1
Empirical mean ..... / 1
Subsample ..... / 1
Bootstrap to obtain IC ..... / 2
Histogram ..... / 1

Exercice 4 Bagging

1. Create a new variable which is a factor with two values: `"above"` for articles which have a number of shares equal to or above the median (1400 shares) and `"below"` for articles which have a number of shares below the median. Then remove from the dataset non informative variables (the first two columns) and columns related to the original target variable (`shares`). Finally, split the data set into a train data set that contains the 5000 observations sampled in the previous exercise and a training data set that contains the remaining observations.

Answer:

```r
# 1. create a new variable (factor), target
df$target <- rep("above", nrow(df))
df$target[df$shares < median(df$shares)] <- "below"
df$target <- as.factor(df$target)

# 2. remove non informative variables and the variable shares
df <- df[ ,-c(1,2,61)]

# 3. separate data into train / test
train_df <- df[sel_obs, ]
test_df <- df[-sel_obs, ]
```

Create new variable ..... / 1
Remove uninformative variables ..... / 0.5
Split data into train / test ..... / 0.5

2. Use the package **ipred** to obtain a bagging of regression trees which predicts the new variable (with values "above" and "below") from all the other informative variables. Use only the training test to train the model with $B = 100$ bootstrap samples. What is the OOB error of this model? What is its test error? Also report the computational time needed to train the model. Start your script by setting the random seed `set.seed(1213)`.

Answer:

```r
set.seed(1213)
# 1. bagging training
library(ipred)
comp_time <- system.time(
  bagging_ipred <- bagging(target ~ ., data = train_df, nbagg = 100,
                           coob = TRUE))

# 2. results
## computational time
comp_time

##    user  system elapsed
##  24.554   0.092  24.681

## OOB error
bagging_ipred$err

## [1] 0.3602

## test error
pred_test <- predict(bagging_ipred, test_df)
err_test <- mean(test_df$target != pred_test)
err_test

## [1] 0.3523843
```

The OOB error of the bagging approach is equal to 36% and the test error of the bagging approach is equal to 35.2%. The computational time needed to train the method is 25 seconds.

Bagging ..... / 2
Computational time ..... / 0.5
OOB error ..... / 0.5
Test error ..... / 1

Exercice 5 Random forest

Train a random forest on the same training data set and with 500 trees (5 times more than in the previous exercise). What is its OOB error and its test error (with the same test set as in the previous exercise)? What is the computational time needed to train this forest? Start your script by setting the random seed `set.seed(1459)`.

Answer:

```
set.seed(1459)
library(randomForest)
# 1. Train de forest
comp_time <- system.time({
  rf <- randomForest(target ~ ., data = train_df, ntree = 500)
})
# 2. computational time
comp_time

##    user  system elapsed
##  16.105   0.029  16.129

# 3. OOB and test errors
rf$err.rate[500,1]

##    OOB
## 0.3408

err_test <- mean(predict(rf, test_df) != test_df$target)
err_test

## [1] 0.340896
```

The OOB error of the bagging approach is equal to 34.1% and the test error of the bagging approach is equal to 34.1%. The computational time needed to train the method is 16 seconds. The method is thus faster and slightly more accurate than the bagging trees trained in the previous section.

Random forest ..... / 2
Computational time ..... / 0.5
OOB error ..... / 0.5
Test error ..... / 1

Exercice 6 Parallel computing

Train the previous forest in parallel and compare the computational times. What is the test error of this forest? (Do not set a random seed for this exercise.)

Answer:

```
library(doMC)
registerDoMC(cores = 2)
library(foreach)
comp_time <- system.time({
  rf_par <- foreach(ind = 1:10, .combine = combine) %dopar% {
    randomForest(target ~ ., data = train_df, ntree = 50)
  }
})
# 2. computational time
comp_time

##    user  system elapsed
##  10.333   0.309  10.657
```

```r
# 3. test error
err_test <- mean(predict(rf_par, test_df) != test_df$target)
err_test

## [1] 0.3387311
```

The test error between the two versions (parallel and sequential) is very similar, as expected. However, the computational time needed to train the forest in parallel is 11 seconds, which is smaller than the time needed to compute the entire forest sequentially.

Parallel computation ..... / 2
Computational time ..... / 0.5
Test error ..... / 0.5
Total : ..... / 21