

# Bagging

From a course by [Davy Paindaveine](#) and [Nathalie Vialaneix](#)

Camille Mondon

## 1 Introduction

- Designed by Breiman (1996).
- The bootstrap has other uses than those described in the last chapter.
- In particular, it allows us to design **ensemble methods** in **statistical learning**.
- **Bagging (Bootstrap Aggregating)**, which is the most famous approach in this direction, can be applied to both **regression** and **classification**.
- Below, we mainly focus on **bagging of classification trees**, but it should be clear that bagging of regression trees can be performed similarly.

## 2 Reminders on classification

### 2.1 The classification problem

- In classification, one observes  $(X_i, Y_i)$ ,  $i = 1, \dots, n$ , where
  - $X_i$  collects the values of  $p$  predictors on individual  $i$ , and
  - $Y_i \in \{1, 2, \dots, K\}$  is the class to which individual  $i$  belongs.
- The problem is to classify a new observation for which we only see  $x$ , that is, to bet on the corresponding value  $y \in \{1, 2, \dots, K\}$ .
- A classifier is a mapping

$$\begin{aligned}\phi_{\mathcal{S}} : \mathcal{X} &\rightarrow \{1, 2, \dots, K\} \\ x &\mapsto \phi_{\mathcal{S}}(x),\end{aligned}$$

that is designed using the sample  $\mathcal{S} = \{(X_i, Y_i), i = 1, \dots, n\}$ .

### 2.2 Data set on a retirement centre

#### 2.2.1 Description

The channing data set has 462 observations and 4 variables.

This data frame contains the following columns:

- **sex**: A factor for the sex of each resident (“Male” or “Female”).
- **entry**: The residents age (in months) on entry to the centre
- **time**: The length of time (in months) that the resident spent at Channing House. (time=exit-entry)
- **cens**: The indicator of right censoring. 0 indicates that the resident died at Channing House, 1 indicates that they left the house prior to July 1, 1975 or that they were still alive and living in the centre at that date.

## 2.2.2 Classification goal

```
library(boot)
data(channing)
channing <- channing[, c("sex", "entry", "time", "cens")]
channing$cens <- as.factor(1 - channing$cens)
channing$sex <- as.factor(channing$sex)
n <- nrow(channing)
channing[sample(1:n, 4), ]
```

|     | sex    | entry | time | cens |
|-----|--------|-------|------|------|
| 289 | Female | 760   | 137  | 1    |
| 121 | Female | 1073  | 119  | 0    |
| 358 | Female | 857   | 137  | 1    |
| 429 | Female | 835   | 10   | 0    |

**Goal:** Predict  $\text{sex} \in \{\text{Male}, \text{Female}\}$  on the basis of two numerical predictors (entry, time) and a binary one (cens).

```
library(ggplot2)

ggplot(channing, aes(entry/12, time/12, color = sex)) +
  geom_point() +
  facet_wrap(vars(ifelse(cens == 1, "Censored", "Uncensored"))) +
  labs(x = "Age on entry (years)", y = "Time spent at the centre (years)")
```

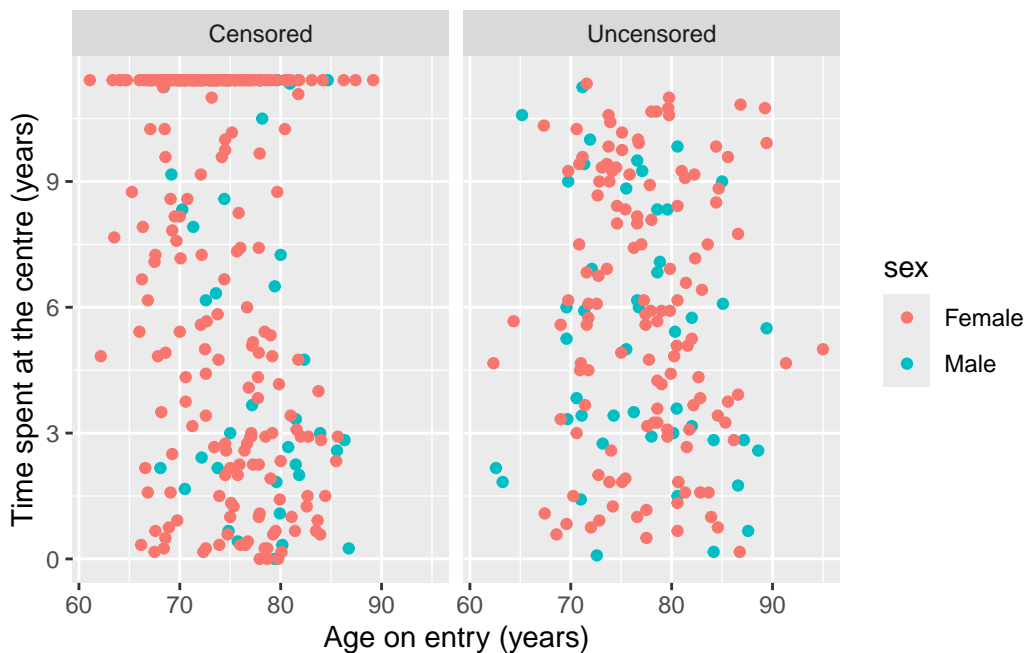
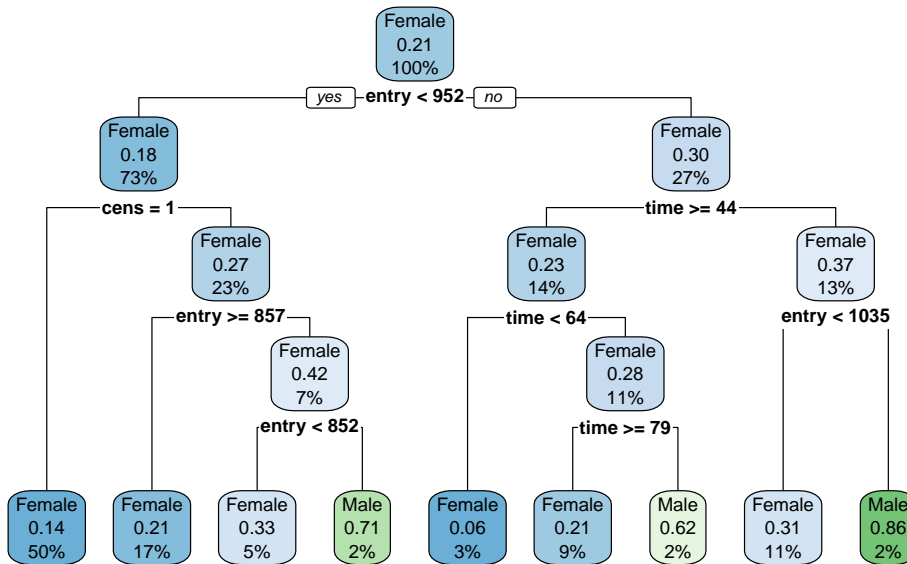


Figure 1: Scatter plot of the channing data set.

## 2.3 Classification trees

In Chapter 3 of this course, we learned about a special type of classifiers  $\phi_{\mathcal{S}}$ , namely classification trees (Breiman et al. 1984).

```
library(rpart)
library(rpart.plot)
fitted.tree <- rpart(sex ~ ., data = channing, method = "class")
rpart.plot(fitted.tree)
```



- (+) Interpretability
- (+) Flexibility
- (-) Stability
- (-) Performance

The process of **averaging** will reduce variability, hence, **improve stability**. Recall indeed that, if  $U_1, \dots, U_n$  are uncorrelated with variance  $\sigma^2$ , then

$$\text{Var}[\bar{U}] = \frac{\sigma^2}{n}.$$

Since unpruned trees have low bias (but high variance), this reduced variance will lead to a low value of

$$\text{MSE} = \text{Var} + (\text{Bias})^2$$

which will ensure a **good performance**.

How to perform this **averaging**?

### 3 Bagging of classification trees

#### 3.1 Bagging

Denote as  $\phi_{\mathcal{S}}(x)$  the predicted class for predictor value  $x$  returned by the classification tree associated with sample  $\mathcal{S} = \{(X_i, Y_i), i = 1, \dots, n\}$ .

**Bagging of this tree** considers predictions from  $B$  bootstrap samples

$$\begin{aligned} \mathcal{S}^{*1} &= ((X_1^{*1}, Y_1^{*1}), \dots, (X_n^{*1}, Y_n^{*1})) \rightsquigarrow \phi_{\mathcal{S}^{*1}}(x) \\ &\vdots \\ \mathcal{S}^{*b} &= ((X_1^{*b}, Y_1^{*b}), \dots, (X_n^{*b}, Y_n^{*b})) \rightsquigarrow \phi_{\mathcal{S}^{*b}}(x) \\ &\vdots \\ \mathcal{S}^{*B} &= ((X_1^{*B}, Y_1^{*B}), \dots, (X_n^{*B}, Y_n^{*B})) \rightsquigarrow \phi_{\mathcal{S}^{*B}}(x) \end{aligned}$$

then proceeds by **majority voting** (i.e., the most frequently predicted class wins):

$$\phi_{\mathcal{S}}^{\text{Bagging}}(x) = \underset{k \in \{1, \dots, K\}}{\operatorname{argmax}} \#\{b : \phi_{\mathcal{S}^{*b}}(x) = k\}$$

#### 3.2 Toy illustration: bagging with $B = 3$ trees

##### 3.2.1 Bootstrap sample n°1

```
library(boot)
set.seed(20)
d <- sample(1:n, replace = TRUE)
fitted.tree <- rpart(sex ~ ., data = channing[d, ], method = "class")
rpart.plot(fitted.tree)
```

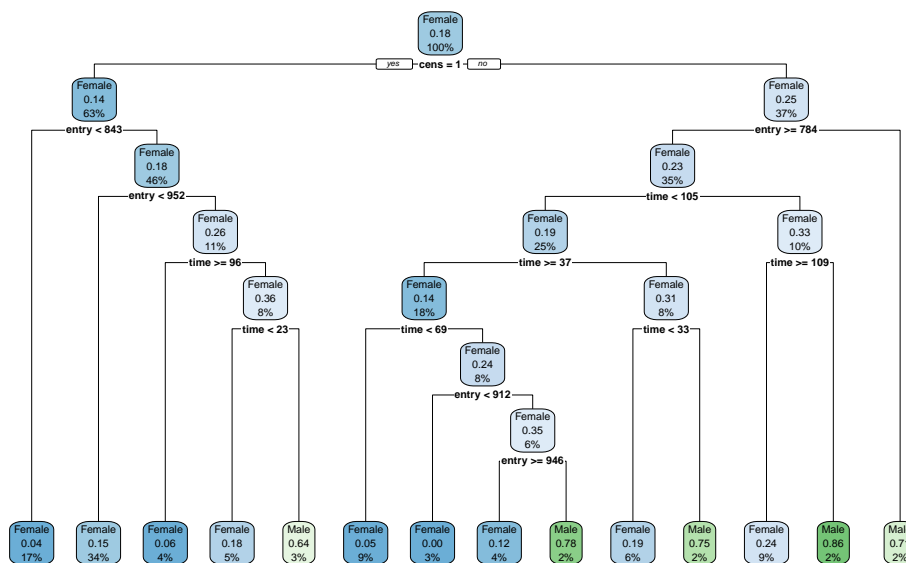


Figure 2: Classification tree from the first bootstrap sample.

```
channing[1, 2:4]
```

| entry | time | cens |
|-------|------|------|
| 782   | 127  | 0    |

```
predict(fitted.tree, channing[1, ],  
  type = "class"  
)
```

```
1  
Male  
Levels: Female Male
```

### 3.2.2 Bootstrap sample n°2

```
d <- sample(1:n, replace = TRUE)  
fitted.tree <- rpart(sex ~ ., data = channing[d, ], method = "class")  
rpart.plot(fitted.tree)
```

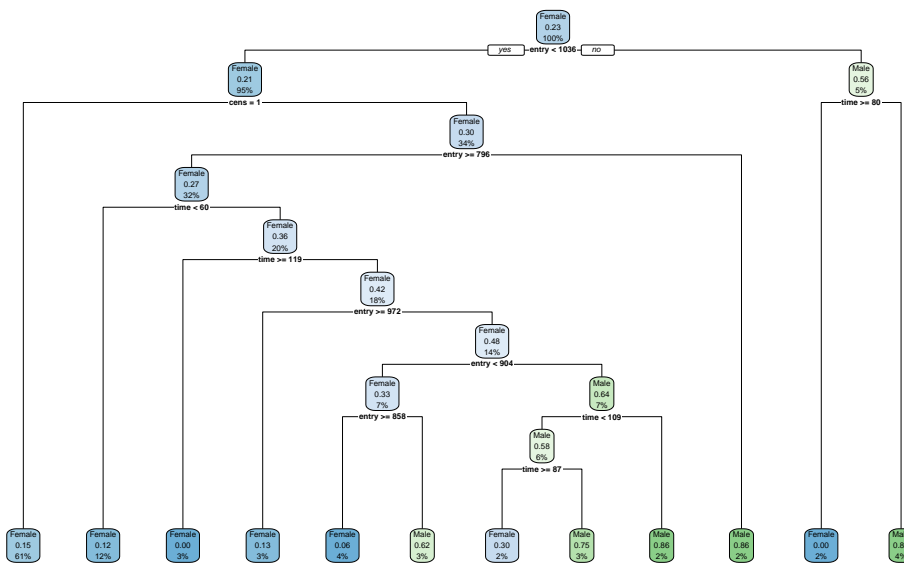


Figure 3: Classification tree from the second bootstrap sample.

```
channing[1, 2:4]
```

| entry | time | cens |
|-------|------|------|
| 782   | 127  | 0    |

```
predict(fitted.tree, channing[1, ],
  type = "class"
)
```

```
1
Male
Levels: Female Male
```

### 3.2.3 Bootstrap sample n°3

```
d <- sample(1:n, replace = TRUE)
fitted.tree <- rpart(sex ~ ., data = channing[d, ], method = "class")
rpart.plot(fitted.tree)
```

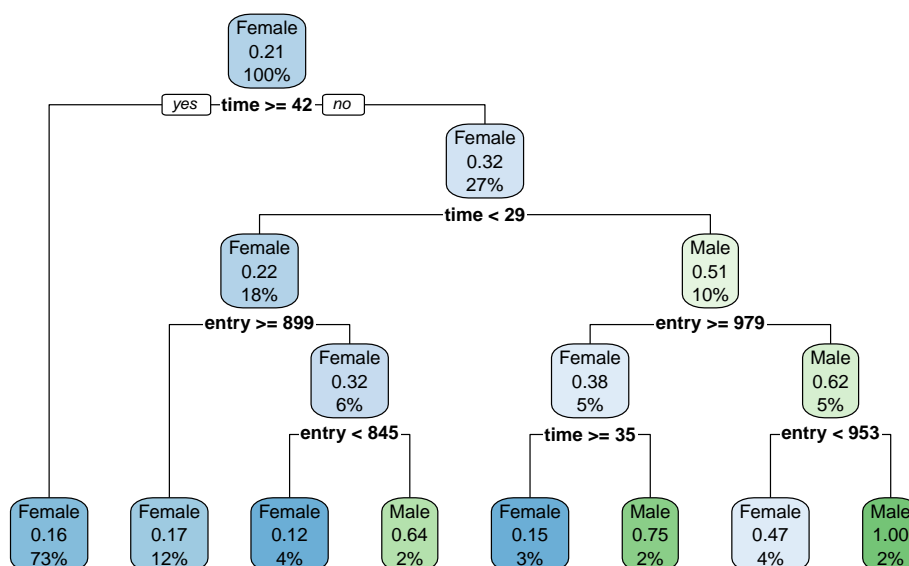


Figure 4: Classification tree from the third bootstrap sample.

```
channing[1, 2:4]
```

| entry | time | cens |
|-------|------|------|
| 782   | 127  | 0    |

```
predict(fitted.tree, channing[1, ],
  type = "class"
)
```

```
1
Female
Levels: Female Male
```

### 3.2.4 Aggregation of the classifiers

For  $x = (\text{entry}, \text{time}, \text{cens}) = (782, 127, 1)$ ,

- **two** (out of the  $B = 3$  trees) **voted for Male**
- **one** (out of the  $B = 3$  trees) **voted for Female**,

the bagging classifier will thus classify  $x$  into **Male**.

Of course,  $B$  is usually much larger ( $B = 500$ ?  $B = 1000$ ?), which requires automating the process (through, e.g., the boot function).

### 3.3 Bagging vs. CART

We repeat  $M = 1000$  times the following experiment:

- (1) Split the data set into a training set (of size 300) and a test set (of size 162);
- (2) (a) **train** a classification tree on the training set and evaluate its **test** error (i.e., misclassification rate) on the test set;  
(b) do the same with a bagging classifier using  $B = 500$  trees.

This provides  $M = 1000$  test errors for the **direct** (single-tree) approach, and  $M = 1000$  test errors for the **bagging** approach.

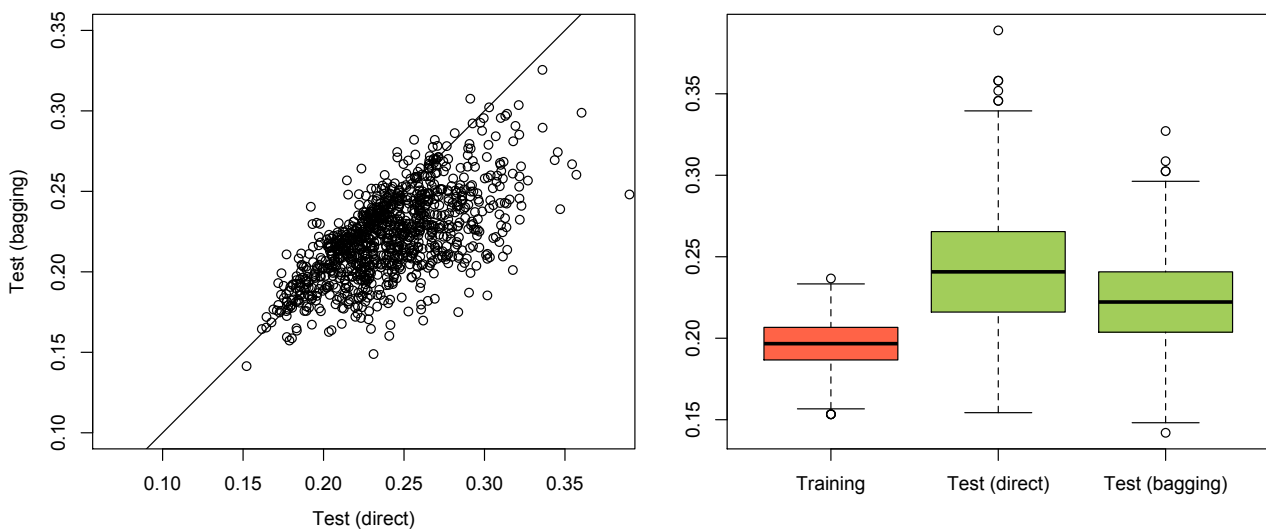


Figure 5: Results of the simulation (Q-Q plot and boxplot).

## 4 Estimating the prediction accuracy

Several strategies to estimate prediction accuracy of a classifier:

- (1) **Compute a test error** (as above): Partition the data set  $\mathcal{S}$  into a training set  $\mathcal{S}_{\text{train}}$  (to train the classifier) and a test set  $\mathcal{S}_{\text{test}}$  (on which to evaluate the misclassification rate  $e_{\text{test}}$ ).

**(2) Compute an  $L$ -fold cross-validation error:**

Partition the data set  $\mathcal{S}$  into  $L$  folds  $\mathcal{S}_\ell$ ,  $\ell = 1, \dots, L$ . For each  $\ell$ , evaluate the test error  $e_{\text{test},\ell}$  associated with training set  $\mathcal{S} \setminus \mathcal{S}_\ell$  and test set  $\mathcal{S}_\ell$ .

Table 5:  $L$ -fold cross-validation framework.

|              | Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5 |
|--------------|--------|--------|--------|--------|--------|
| Run $\ell=1$ | Test   | Train  | Train  | Train  | Train  |
| Run $\ell=2$ | Train  | Test   | Train  | Train  | Train  |
| Run $\ell=3$ | Train  | Train  | Test   | Train  | Train  |
| Run $\ell=4$ | Train  | Train  | Train  | Test   | Train  |
| Run $\ell=5$ | Train  | Train  | Train  | Train  | Test   |

Then the ( $L$ -fold) ‘cross-validation error’ is:

$$e_{\text{CV}} = \frac{1}{L} \sum_{\ell=1}^L e_{\text{test},\ell}$$

**(3) Compute the Out-Of-Bag (OOB) error<sup>1</sup>:**

For each observation  $X_i$  from  $\mathcal{S}$ , define the OOB prediction as

$$\phi_{\mathcal{S}}^{\text{OOB}}(X_i) = \underset{k \in \{1, \dots, K\}}{\text{argmax}} \#\{b : \phi_{\mathcal{S}^{*b}}(X_i) = k \text{ and } (X_i, Y_i) \notin \mathcal{S}^{*b}\}$$

This is a **majority voting** discarding, quite naturally, bootstrap samples that use  $(X_i, Y_i)$  to train the classification tree. The OOB error is then the corresponding misclassification rate

$$e_{\text{OOB}} = \frac{1}{n} \sum_{i=1}^n \mathbb{1}[\phi_{\mathcal{S}}^{\text{OOB}}(X_i) \neq Y_i]$$

## 5 Final remarks

- **Bagging of trees can also be used for regression.** The only difference is that majority voting is then replaced with an averaging of individual predicted responses.
- **Bagging is a general device that applies to other types of classifiers.** In particular, it can be applied to  $k$ -NN classifiers (we will illustrate this in the practical sessions).
- **Bagging affects interpretability of classification trees.** There are, however, solutions that intend to measure importance of the various predictors (see the next section).

## References

Breiman, Leo. 1996. “Bagging Predictors.” *Machine Learning* 24 (2): 123–40. <https://doi.org/10.1007/BF00058655>.  
 Breiman, Leo, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. 1984. *Classification And Regression Trees*. 1st ed. Routledge. <https://doi.org/10.1201/9781315139470>.

<sup>1</sup>This is for bagging procedures only.