Random forests

From a course by Davy Paindaveine and Nathalie Vialaneix

Camille Mondon

1 How to improve bagging?

1.1 Averaging reduces variability...

- We argued that bagging of trees will work because **averaging** reduces variability: if $U_1, \ldots, U_n$ are uncorrelated with variance σ^2, then

$$\mathrm{Var}[\bar{U}] = \frac{\sigma^2}{n}\cdot$$

- However, the B trees that are 'averaged' in bagging are **not uncorrelated**! This will result into a smaller reduction of the variability.
- Random forests have the flavour of bagging-of-trees, but they incorporate a modification that aims at **decorrelating the trees**.

2 The procedure

2.1 Random forests

- Like bagging-of-trees, random forests predict via majority voting from classification trees trained on B bootstrap samples.
- **However, whenever a split is designed in each tree, the split is only allowed among** $m(<< p)$ **predictors randomly selected out of the** p **predictors**.
- **The rationale**: in a situation where there would be one strong predictor only, most bagged trees would use this predictor in the top split, which would result in highly correlated trees. The **tweak** above will prevent this, hence will lead to **less correlated trees**.

Using $m = p$ would simply provide bagging-of-trees. Using m small is appropriate when there are many correlated predictors. Common practice:

- For **classification** (where majority voting is used), $m \approx \sqrt{p}$.
- For **regression** (where tree predictions are averaged), $m \approx \frac{p}{3}$.

In both cases, results are actually not very sensitive to m.

Random forests

☒ are **nonparametric** and **efficient**

☒ can deal with **a large number of predictors** (high dimension)
☒ can cope with **both small and large sample sizes** (Big Data)

but they

□ rely on a rather **black box** model, and
□ are **not supported by strong theoretical results**

2.2 A simulation

Let us look at efficiency…

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

(1) Split the `channing` 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;
  (c) **do the same with a random forest classifier using the `randomForest` function in `R` with default parameters ($B = 500$ trees, $m \approx \sqrt{p}$).**

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

2.3 A simulation

3 Importance of each predictor

3.1 Measuring importance of each predictor

Because bagging-of-trees and random forests are poorly interpretable compared to classification trees, the following is useful.

The **importance, v_j say, of the jth predictor** is measured as follows.

For each tree (i.e., for any $b = 1, \ldots, B$),

- the prediction error on OOB observations is recorded, and
- the same is done after permuting randomly all values of the jth predictor (which essentially turns this predictor into noise), the **difference between both errors** is then averaged over $b = 1, \ldots, B$ (and normalized by the standard error—if it is positive), yielding v_j.

(A similar measure is used for regression, based on MSEs).

3.2 Measuring importance of each predictor

References

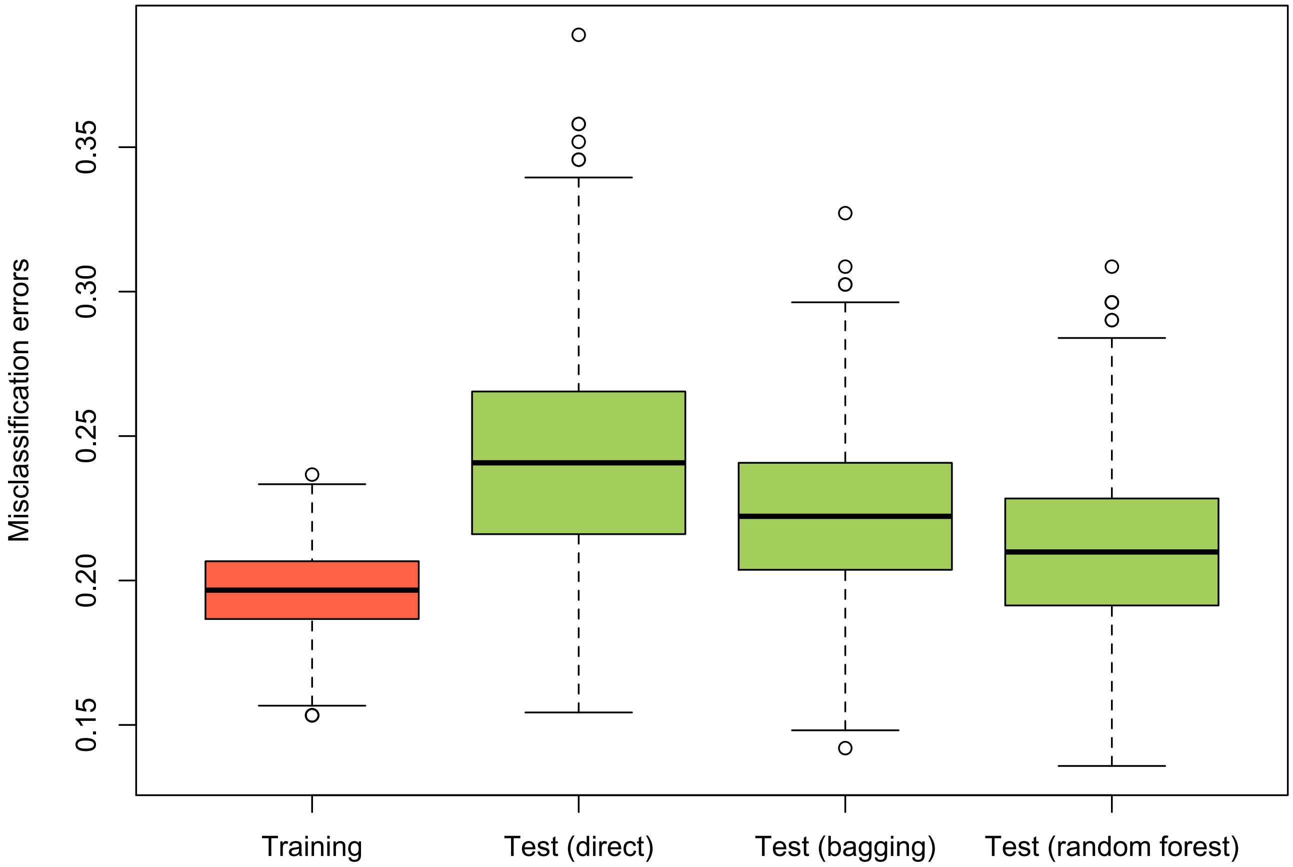

Figure 1: Results of the simulation (boxplots of misclassification errors).

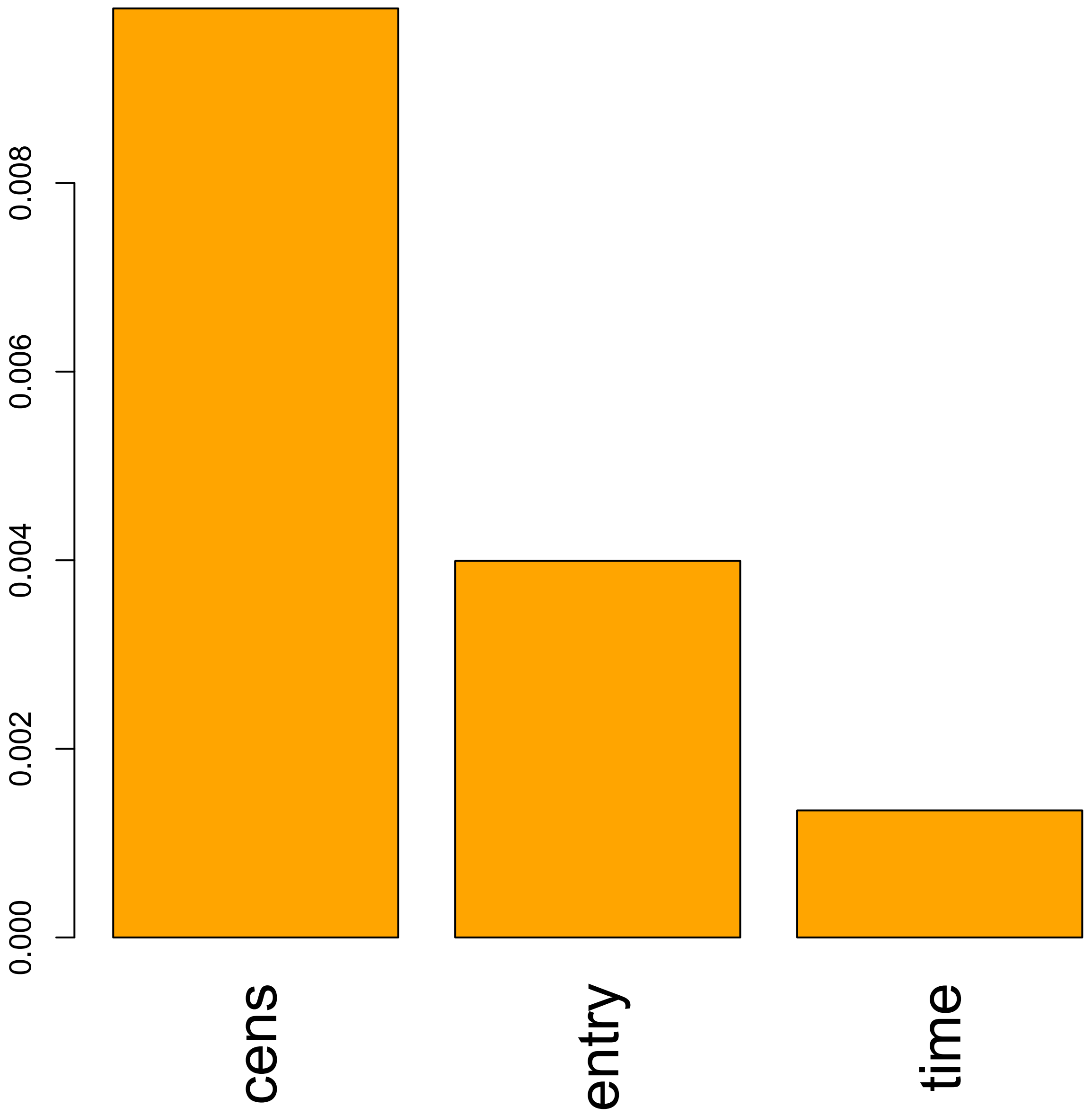

Figure 2: Importance of each predictor (decreasing order).