40.016 The Analytics Edge

Bagging and Random Forests

Stefano Galelli

SUTD

Term 6, 2020

Outline

- Bootstrapping
- Bagging
- Random Forests

Bootstrapping

- The **Bootstrap** is a resampling method used to quantify the uncertainty associated with a statistical learning method (or a given estimator)
- (The other resampling method introduced in this course is **cross-validation**)
- The term originates from the expression *"to pull oneself up by one's bootstraps"*

Bootstrapping

Example

- We want to invest a given amount of money in two financial assets that yield returns of X and Y, respectively, where X and Y are random quantities
- We will invest a fraction α of our money in X, and will invest the remaining $1 - \alpha$ in Y
- We wish to choose the value of α that minimizes the total risk, measured with the variance of our investment, $Var(\alpha X + (1 - \alpha)Y)$

Bootstrapping

Example

- We want to invest a given amount of money in two financial assets that yield returns of X and Y, respectively, where X and Y are random quantities
- We will invest a fraction α of our money in X, and will invest the remaining $1 - \alpha$ in Y
- We wish to choose the value of α that minimizes the total risk, measured with the variance of our investment, $Var(\alpha X + (1 - \alpha)Y)$
- The value of α that minimizes the risk is

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$$

Bootstrapping

Example

- The values of σ_X^2, σ_Y^2, and σ_{XY} are unknown
- We can calculate estimates ($\hat{\sigma}_X^2$, $\hat{\sigma}_Y^2$, and $\hat{\sigma}_{XY}$) using a dataset with observations of X and Y
- And then estimate $\hat{\alpha}$

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_{XY}}{\hat{\sigma}_X^2 + \hat{\sigma}_Y^2 - 2\hat{\sigma}_{XY}}$$

Bootstrapping

Example

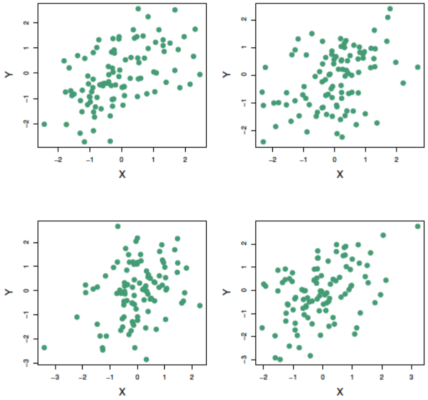

Figure 1: 100 simulated returns for investments X and Y (Source: James et al., 2014).

Bootstrapping

Example

We repeat this process 1,000 times, so as to get a better estimate of α:

- We set $\sigma_X^2 = 1$, $\sigma_Y^2 = 1.25$, and $\sigma_{X,Y} = 0.50$, so we know that the true value of α is 0.60
- The mean over all 1,000 estimates for α is

  $$\bar{\alpha} = \frac{1}{1000} \sum_{r=1}^{1000} \hat{\alpha}_r = 0.5996$$

  which is close to 0.60. The standard deviation of the estimates is:

  $$\sqrt{\frac{1}{1000-1} \sum_{r=1}^{1000} (\hat{\alpha}_r - \bar{\alpha})^2} = 0.083.$$

Bootstrapping

Back to the real world . . .

- The example above cannot be used, because we cannot generate new samples from the original population
- **Bootstrapping** mimics this process: we obtain distinct datasets by repeatedly sampling observations from the original data set *with replacement*
- Each of these bootstrap datasets is created by sampling with replacement, and is the same size as our original dataset

Bootstrapping

Example (with three observations)

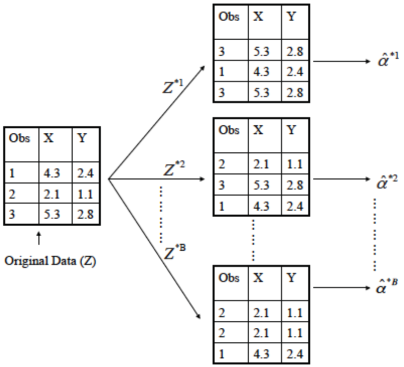

Figure 2: A graphical illustration of the bootstrap approach on a small sample containing $n = 3$ observations. (Source: James et al., 2014).

Bootstrapping

A few more points

- Denoting the first bootstrap data set by Z^{*1}, we use $Z^{*1}$to produce a new bootstrap estimate for α which we call $\hat{\alpha}^{*1}$
- The procedure is repeated B times (for example, 100 or 1,000), in order to produce B different bootstrap datasets (Z^{*1}, Z^{*2}, ..., Z^{*B}), and B corresponding estimates $\hat{\alpha}^{*1}$, $\hat{\alpha}^{*2}$, ..., $\hat{\alpha}^{*B}$
- We estimate the standard error of these bootstrap estimates with the formula

$$SE_B(\hat{\alpha}) = \sqrt{\frac{1}{B-1}\sum_{r=1}^{B}(\hat{\alpha}^{*r} - \bar{\hat{\alpha}}^{*})^2}.$$

Bootstrapping

A general picture for the bootstrap

Bootstrapping

What is the difference between cross-validation and Bootstrapping?

1. Bootstrap resamples with replacement, while cross-validation resamples without replacement
2. The main goal of cross-validation is to measure, or generalize, the performance of a model
3. Bootstrapping is used to establish empirical distribution functions for a widespread range of statistics

Bootstrapping

Can we use Bootstrapping to estimate the prediction error?

- We could think about using each bootstrap dataset as our training sample, and the original sample as our validation sample
- But each bootstrap sample has significant overlap with the original data –> the bootstrap will underestimate the true prediction error
- We can fix the problem by using the *out-of-bootstrap estimate* . . .

Bootstrapping

So, where can we use Bootstrapping?

It can be used to tackle the bias-variance trade-off of some statistical learning methods, such as . . . **Decision Trees**.

Bagging

Intuition: Decision Trees suffer from **high variance** –> if we split the training data into two parts at random, and fit a decision tree to both halves, the results that we get could be quite different.

Bagging (or **Bootstrap aggregation**) can be used to reduce the variance of a statistical learning method.

Bagging

How does it work?

- Given a set of n independent observations $Z_1, Z_2, \ldots, Z_n$, each with variance σ^2, the variance of the mean $\bar{Z}$ of observations is σ^2/n −> *averaging a set of observations reduces variance*
- When learning a statistical model, we can therefore:
  1. Take (Bootstrap) many training sets from the population
  2. Build a separate model using each training dataset
  3. Average the resulting predictions

Bagging

Let's write this (slightly) more formally:

1. Take repeated samples from the training dataset (if that's everything we have) –> we generate B different bootstrapped training datasets
2. We train model $f^{*b}(x)$ on the b-th bootstrapped training dataset (repeat for all B datasets)
3. We average all the predictions as follows

$$f_{bag}(x) = \frac{1}{B} \sum_{b=1}^{B} f^{*b}(x).$$

This entire process is called **Bagging**.

Bagging

How does it work for Classification Trees?

We only change Step 3: we record the class predicted by each of the B trees and take a **majority vote**.

Bagging

Out-of-Bag Error Estimation

- To test the error of a bagged model, we can use the out-of-bag (OOB) observations
- More specifically:
  - To obtain a single prediction for the *i*-th observation, we can average these predicted responses (for regression) or can take a majority vote (for classification). This leads to a single OOB prediction for the *i*-th observation
  - An OOB prediction can be obtained in this way for each of the n observations, from which we compute the overall OOB MSE or classification error
- When B is sufficiently large, we have a decent alternative to cross-validation

Back to R!

How can we implement Bagging in R? Options:

1. Write our own code
2. Use the function `bagging` (package `ipred`). But note this only works for Decision Trees.

Random Forests

Problem: When building Decision Trees with Bagging, trees tend to be *correlated*.

Example:

- Suppose the training dataset has a strong predictor and a few moderately strong predictors
- Then, most of the bagged trees will use the strong predictor in the top split(s) –> trees will be correlated, so we won't reduce variance much

Random Forests

Idea: When building the trees, we consider **a random sample of** m **predictors** from the full set of predictors p (at each split):

–> For each split, we consider only a subset of predictors

–> On average, $(p - m)$ predictors are not considered, so other predictors will have a chance

–> This process decorrelates the trees

Random Forests

Algorithm. Main steps:

1. Generate B boostrapped training datasets
2. For each dataset, train a Decision Tree. At each split, use a subset m of the p available predictors
3. Average the predictions from the B trees. (For classiciation, use majority voting.)

On the value of m

The fundamental difference between Bagging and Random Forests stands in the subset of predictors m:

- If $m = p$, then there is no difference between the two methods
- Recommended values of m:
  - Regression: $m = p/3$
  - Classification: $m = \sqrt{p}$
- Note that these values were found experimentally, so there is no theoretical guarantee they will provide the best performance on all datasets

Random Forests

Hyperparameters tuning: It is common practice to explore the effect of the hyperparameters value on the performance of Random Forests. To recap, we have the following parameters:

- Number of trees, B
- Number of predictors used at each split, $m \leq p$
- (number of points in each terminal leaf)

There are no optimization routines to find their values. We typically use *grid search*, or similar.

Back to R!

To learn a Random Forest, we will use the function `randomForest`, implemented in the package . . . `randomForest`:

```
randomForest(formula, data, ntree, mtry, ...)
```

Advantages and Disadvantages of Random Forests

Pros:

- Better bias-variance trade-off than CARTs
- Higher accuracy

Cons:

- Less interpretable
- Higher computational requirements

References

- James et al. (2014) *An Introduction to Statistical Learning with Applications in R*, Springer, 2014. Chapter 5.2 and 8.2.