Statistical Machine Learning
Part 8
Random Forest

Horacio Gómez-Acevedo
Department of Biomedical Informatics
University of Arkansas for Medical Sciences

March 11, 2021

UAMS

Bagging

We have seen that regression (or classification) trees are very useful but particularly unstable. That is, they suffer from high variance when presented with new data.
Fortunately, there is a general purpose procedures to reduce variance called *Bootstrap aggregation* or *bagging*.
Key Idea: Given a set of m independent (and identically distributed) observations $X_1, \ldots, X_m$ each with variance σ^2, the variance of the mean $\overline{X}$ is given by

$$\hat{\sigma}^2(\overline{X}) = \frac{1}{m}\sigma^2$$

Averaging a set of observations reduces variance

Bagging (cont)

It seems reasonable that for a given model $Y = f(X) + \varepsilon$ to get an estimate response at the point $X = x$ by averaging $\hat{f}^1(x), \ldots, \hat{f}^B(x)$ using B separate training sets, that is

$$\hat{f}_{\text{avg}}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^b(x)$$

where $\hat{f}^k$ is a tree regression estimate (without pruning) based on the observations $\{x_{k1}, \ldots, x_{kB}\}$.
Since we don't have multiple training sets, then we exploit **bootstrap!**. More precisely, for a bootstrapped training set b, we obtain the estimate $\hat{f}^{*b}(x)$, and the corresponding **bagging estimate**

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^{*b}(x)$$

Bagging for Regression Trees

There are few steps that we need to take for regression trees

- Construct B regression trees using B bootstrapped training sets (without pruning)
- Average the resulting predictions

Keep in mind that each (bootstrapped) tree has low bias but high variance. The bagging procedure will reduce the variance at the expense of a "modest" increase in bias.

Bagging for Classification Trees

The steps are similar for classification trees

- Construct B regression trees using B bootstrapped training sets (without pruning)
- The prediction will be based on the majority vote. That is, the class most commonly occurring will be selected.

Out-of-Bag Observations

The main idea of the bootstrap is that from m observations, we select a sample with replacement m observations.
What is the probability of **not** selecting sample 1 ?
The probability of picking sample different from 1 would be $(1 - \frac{1}{m})$. Since we are repeating the experiment with replacement, the probability that a bootstrap sample does not contain sample 1 is

$$\left(1 - \frac{1}{m}\right) \cdot \left(1 - \frac{1}{m}\right) \cdots \left(1 - \frac{1}{m}\right) = \left(1 - \frac{1}{m}\right)^m$$

A little bit of calculus shows that

$$\lim_{m \to \infty} \left(1 - \frac{1}{m}\right)^m = \exp(-1) \approx 36.79\%$$

Thus, bootstrapping will not touch about $1/3$ of the observations! and those observations are referred to as **Out-of-Bag (OOB)**.

OOB Error Estimation

We can exploit the OOB observations to estimate the test error in the bagging process without the need of cross-validation or even a split of the data in training and testing.
Once we have obtained our $\hat{f}_{bag}(x)$, we can use the OOB observations (i.e., observations not used for the bagging estimation) to determine predictions.
More precisely, we obtain $\hat{f}^i_{\text{oob}}(x)$ based on the OOB observations $\{x_{i1}, \ldots, x_{iK}\}$, where $K \approx B/3$ for B big enough.

$$\hat{f}_{OOB}(x) = \frac{1}{K} \sum_{i=1}^{K} \hat{f}^i_{\text{oob}}(x)$$

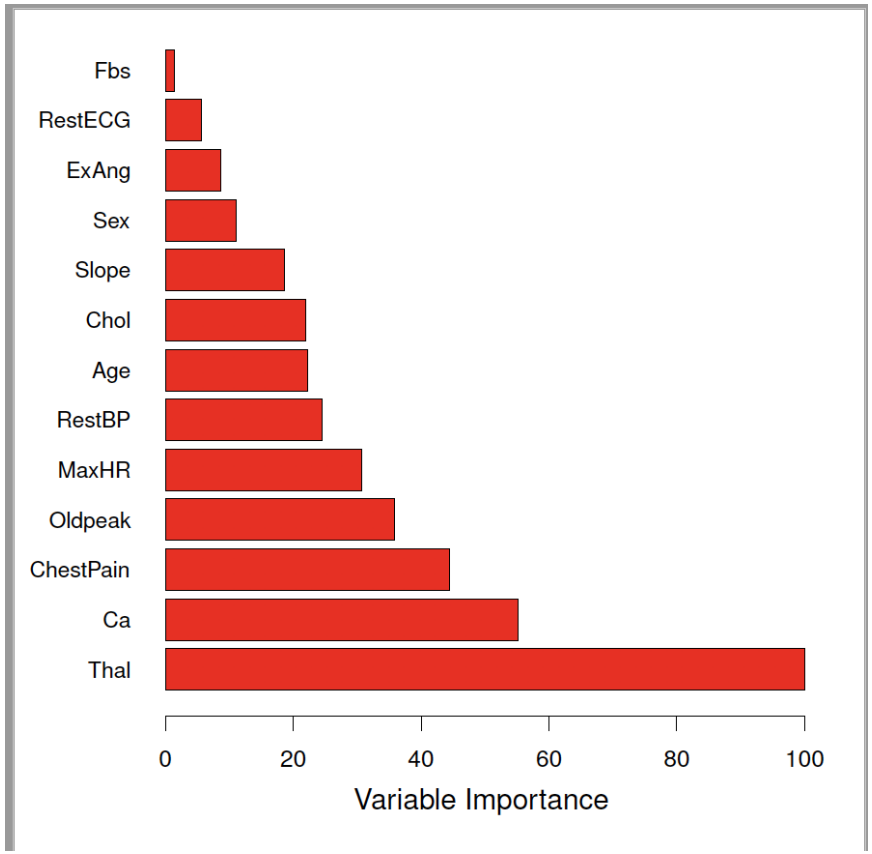
This procedure leads to the calculation of the test MSE that is a valid estimate since the response is derived from trees that were not involved in the bagged model.
A similar expression is valid for classification, but instead of the average we can use the majority vote and purity metrics instead of MSE or RSS.

Variable Importance Measures

We know that bagging improves the accuracy in our predictions, at the expense of making our models harder to interpret.
For bagging regression trees, we use the **Variable Importance Measure (VIM)** that is defined as the total amount that the RSS is decreased due to splits over the given predictor, averaged over all B trees. The larger the VIM, the more "relevant" is that predictor.
For bagging classification trees, we can define VIM as the total amount that the Gini index (or cross-entropy) is decreased by splits over a given predictor, averaged over all B trees.

VIM example

The `Heart` data set VIM plot with a mean decrease of GIni index and normalized VIM is shown below.

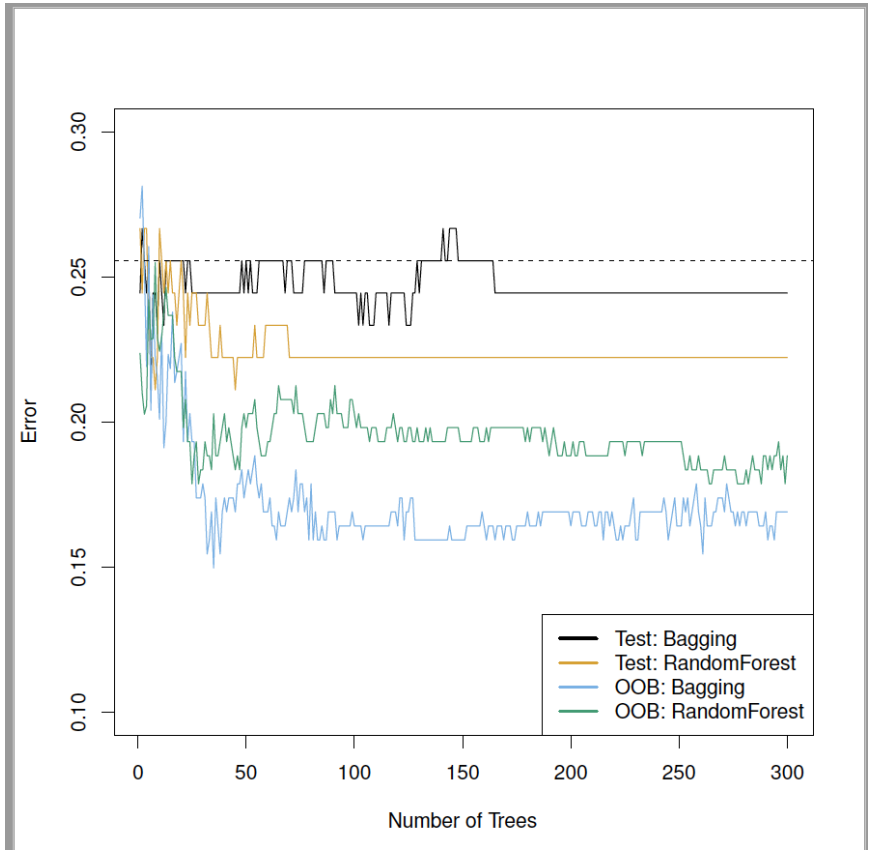


Random Forest

It follows similar rationale as in bagging but with an interesting random twist.
We build a number of decision trees on bootstrapped training samples. But when building these decision trees, each time a split in a tree is considered, a *random sample of m predictors* is chosen as split candidates from the full set of p predictors. We normally set $m \approx \sqrt{p}$.
What is the advantage of random forest over bagging?
When we have a strong predictor, bagging trees will consider that predictor frequently, thus bagging trees will look alike. By having a random choice on the predictors, we may generate "different" trees that otherwise we would not have explored. This process is referred to as *decorrelating trees*.

Random Forest vs Bagging

The test errors from the `Heart` data are depicted below



Boosting

Boosting is another general methodology to improve the predictions from a decision tree. In this case trees are grown sequentially as they gather information from previously generated trees.
Boosting does not require bootstrap sampling as each tree is fit on a modified version of the original data set.

Boosting Algorithm

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set.
2. for $b = 1, \ldots, B$ repeat:
   2.1 Fit a tree $\hat{f}^b$ with d splits to the training data (X, r).
   2.2 Update $\hat{f}$ by adding in a shrunken version of the new tree:

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$

   2.3 Update the residuals,

$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$$

3. Output the boosted model

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x) \tag{1}$$

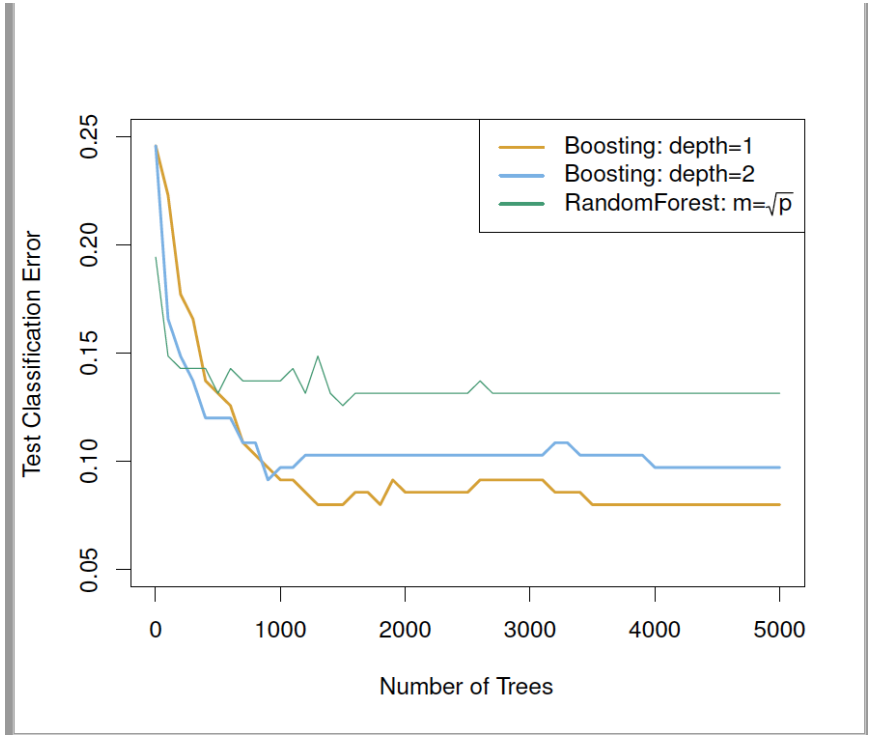
Boosting

Given a current model, we fit a decision tree to the residuals from the model rather than the outcome Y as the response. And we add these residuals to a new decision tree and update again the residuals.
Boosting has the following tuning parameters:

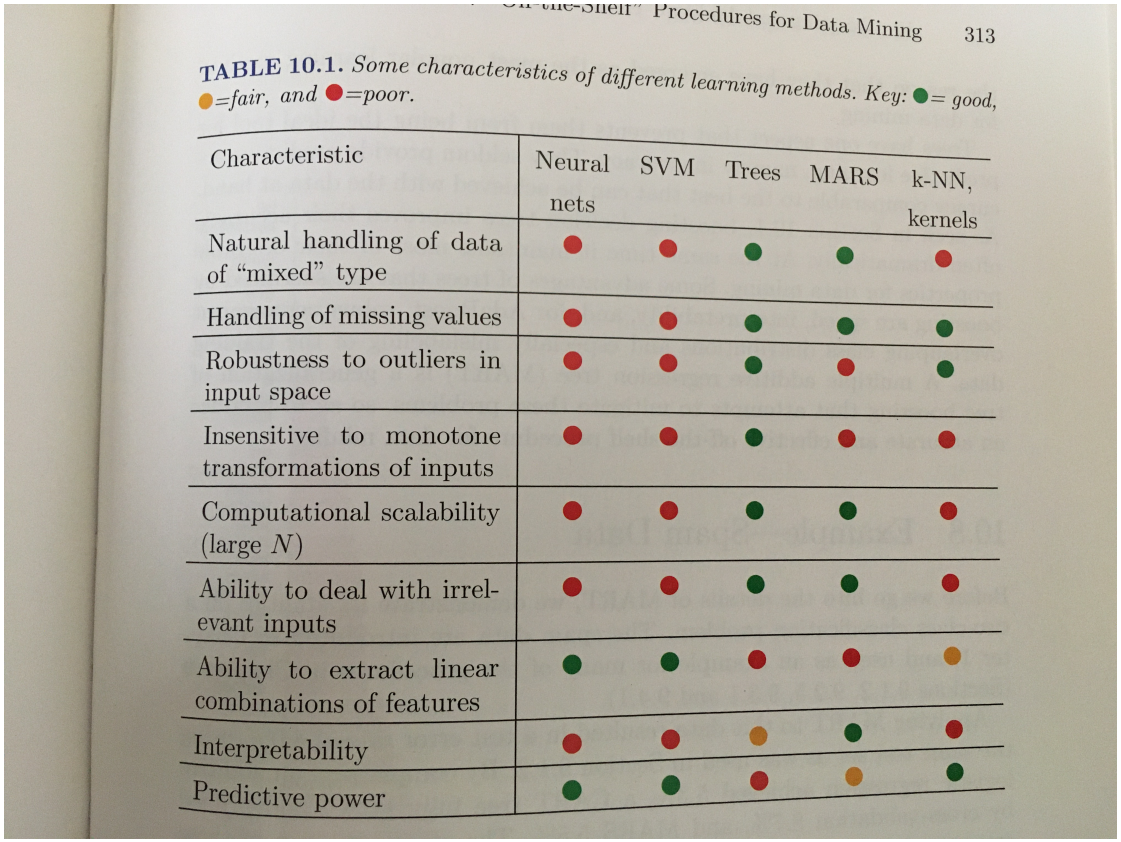
- B that represents the number of trees. Do not take very large values of B as boosting tends to overfit. We can use cross-validation to determine a good candidate for B.
- The shrinkage parameter λ controls the learning rate.
- The number of splits d controls the complexity of the boosted ensemble. Sometimes $d = 1$ works well.

Boosting

Boosting and random forest comparison in a 15-class gene expression data set to predict cancer.



Final Thoughts



...the-Shelf" Procedures for Data Mining 313

TABLE 10.1. *Some characteristics of different learning methods. Key:* ● *= good,* ● *=fair, and* ● *=poor.*

Characteristic	Neural nets	SVM	Trees	MARS	k-NN, kernels
Natural handling of data of "mixed" type	●	●	●	●	●
Handling of missing values	●	●	●	●	●
Robustness to outliers in input space	●	●	●	●	●
Insensitive to monotone transformations of inputs	●	●	●	●	●
Computational scalability (large N)	●	●	●	●	●
Ability to deal with irrelevant inputs	●	●	●	●	●
Ability to extract linear combinations of features	●	●	●	●	●
Interpretability	●	●	●	●	●
Predictive power	●	●	●	●	●

References

Materials and some of the pictures are from (1),(2), and (3).

1. Gareth James et al. *An Introduction to Statistical Learning with applications in R*. Springer (2015)
2. Trevor Hastie et al. *The Elements of Statistical Learning* Springer (2001).
3. Aurélien Géron. *Hands-on Machine Learning with Scikit-Learn & TensorFlow* O'Relly (2017)

I have used some of the graphs by hacking TiKz code from StakExchange, Inkscape for more aesthetic plots and other old tricks of TEX