BAGGING AND RANDOM FORESTS

Jelena Jovanovic

Email: jeljov@gmail.com

Web: http://jelenajovanovic.net

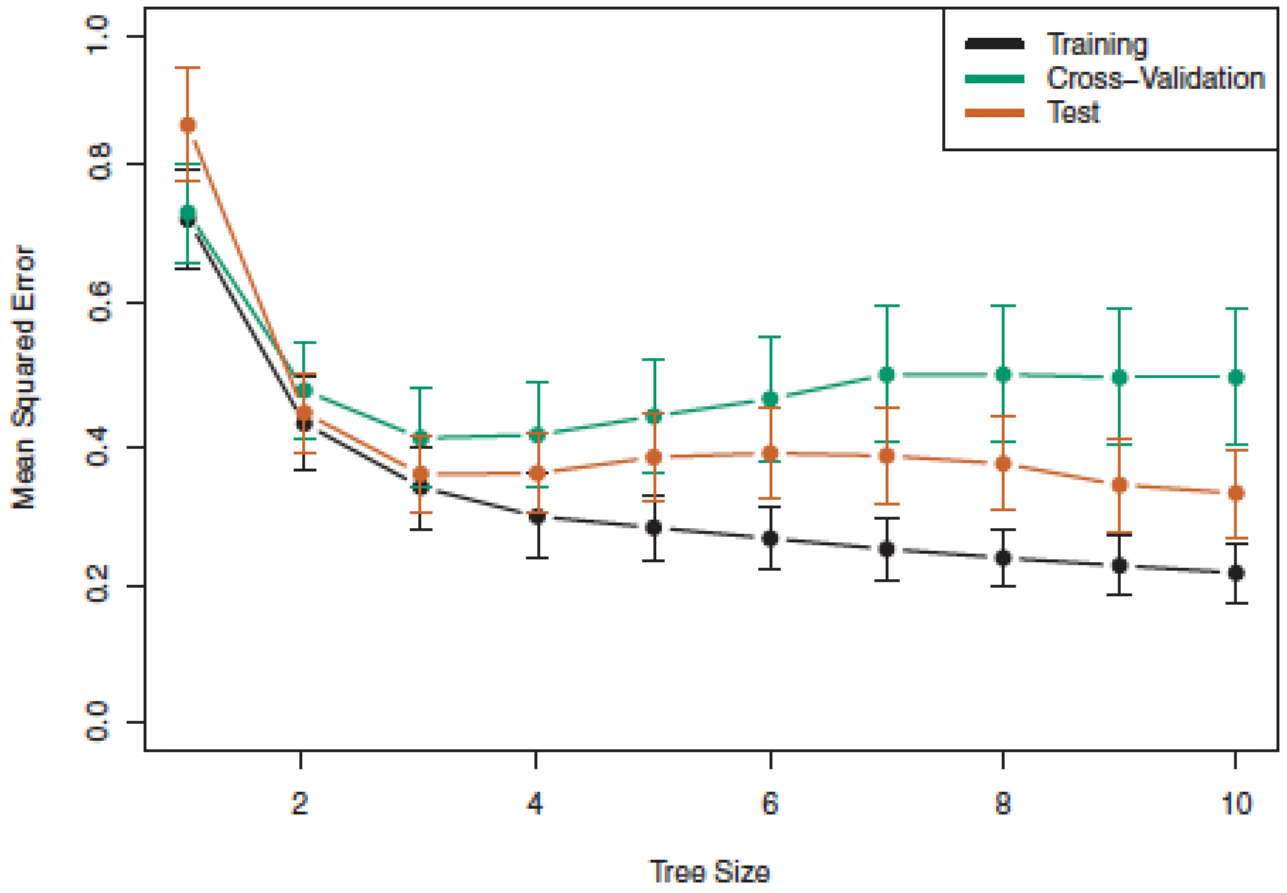
Outline

- Bagging
  - Bootstrapping
  - Bagging for Regression Trees
  - Bagging for Classification Trees
  - Out-of-Bag Error Estimation
  - Variable Importance: Relative Influence Plots
- Random Forests

BAGGING

High variance problem

Decision trees tend to *overfit* and suffer from *high variance*



Bagging

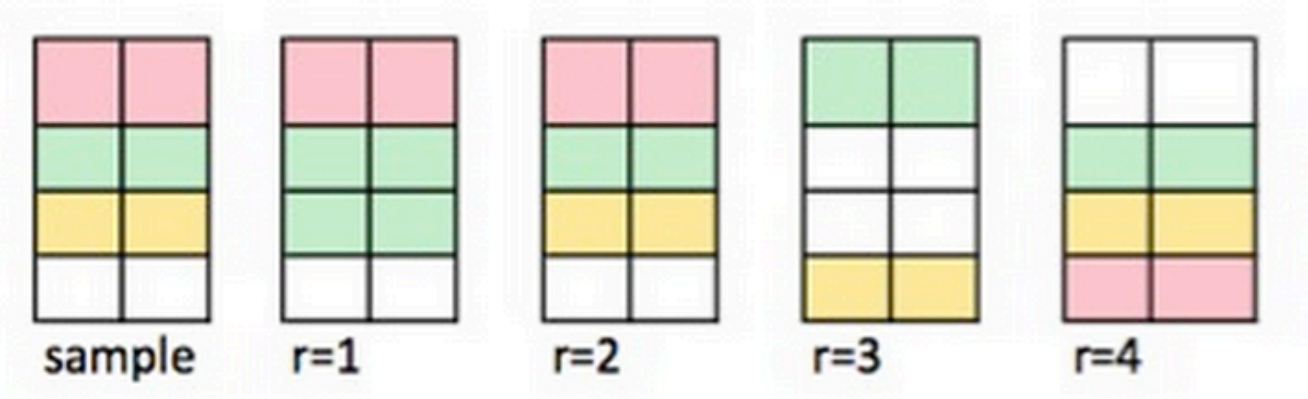
- It is desirable to have models with *low variance* as these yield similar results when applied to distinct data sets

- One way to solve this high variance problem is to use ***bagging*** – ***b***ootstrap ***agg***regat***ing***

What is bootstrapping?

It consists of resampling the given dataset through random sampling with replacement

- i.e., each new dataset is obtained by random sampling with replacement from the original dataset
- the new datasets are of the same size as the original dataset



What is bagging?

- Bagging is a general-purpose procedure for reducing the variance of a machine learning method
- It is based on two key things:
  - Bootstrapping: provides a plenty of training datasets
  - Averaging: leads to a reduction in variance
- Why does averaging reduce variance?
  - Given a set of *n* independent observations $Z_1, \ldots, Z_n$, each with variance σ^2, the variance of the mean $\bar{Z}$ of the observations is given by σ^2/n

How does bagging work?

- Generate B different bootstrapped training datasets
- Train the chosen machine learning method on each of the B training datasets
- Make prediction:
  - Regression: average all predictions from all B models
  - Classification: majority vote among all B models

Bagging for regression trees

Procedure:

- Create B bootstrapped training datasets
- Use the training sets to construct B regression trees and generate predictions with each one
- Average the resulting predictions

Important:

- The produced trees are not pruned, so each individual tree has high variance but low bias
- Averaging the predictions produced by these trees reduces the variance, thus leading to low variance and bias

Example 1: Housing Data

```
Console ~/R Studio Projects/Intro to Statistical Learning/
> data(Boston)
> str(Boston)
'data.frame':   506 obs. of  14 variables:
 $ crim   : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
 $ zn     : num  18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...
 $ indus  : num  2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
 $ chas   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ nox    : num  0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...
 $ rm     : num  6.58 6.42 7.18 7 7.15 ...
 $ age    : num  65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
 $ dis    : num  4.09 4.97 4.97 6.06 6.06 ...
 $ rad    : int  1 2 2 3 3 3 5 5 5 5 ...
 $ tax    : num  296 242 242 222 222 222 311 311 311 311 ...
 $ ptratio: num  15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
 $ black  : num  397 397 393 395 397 ...
 $ lstat  : num  4.98 9.14 4.03 2.94 5.33 ...
 $ medv   : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
>
```

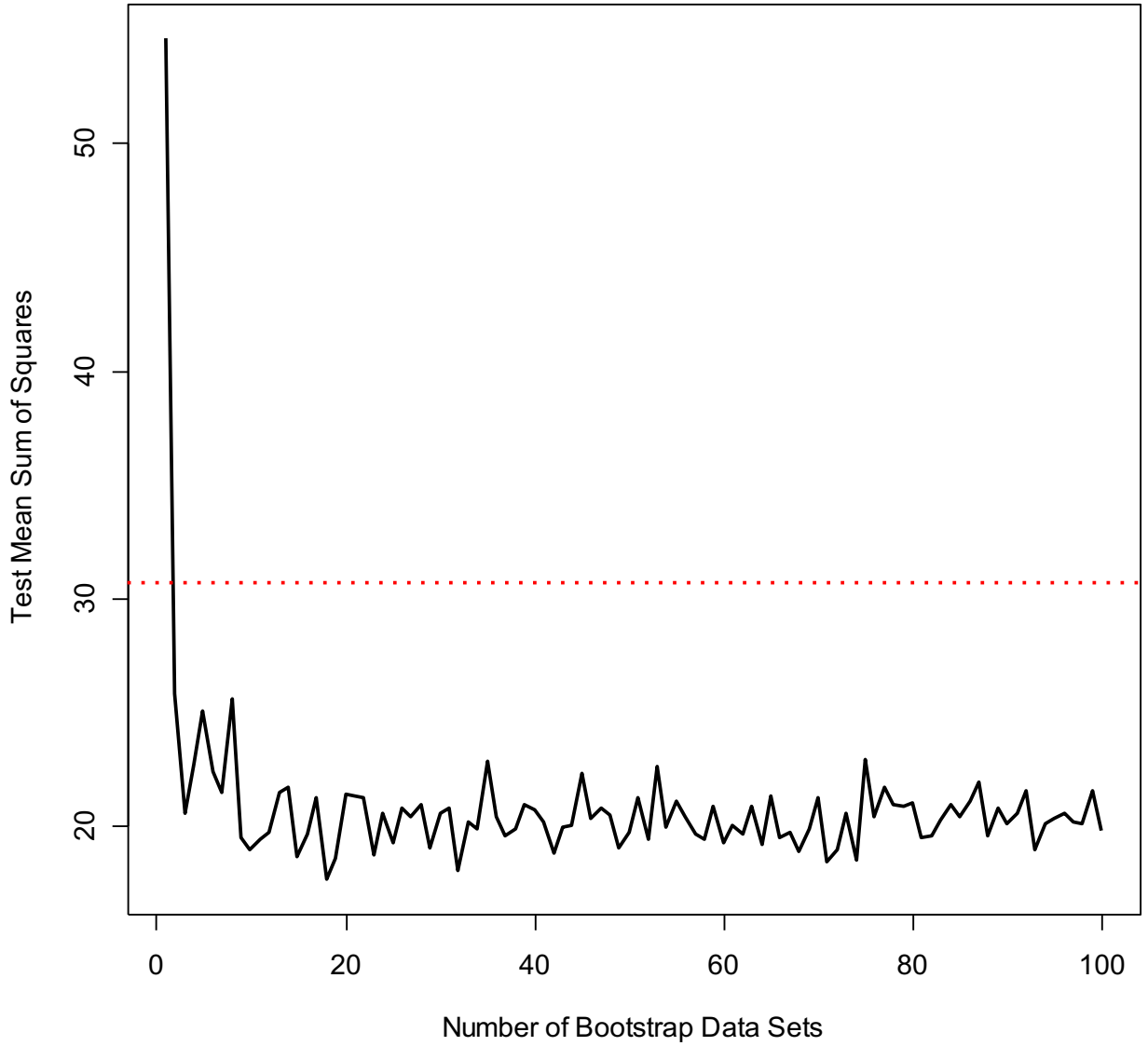
the value to be predicted

Source: http://lib.stat.cmu.edu/datasets/boston

Example 1: Housing Data

The red line represents the test mean square error using a single tree

The black line corresponds to the bagging error rate



Bagging for classification trees

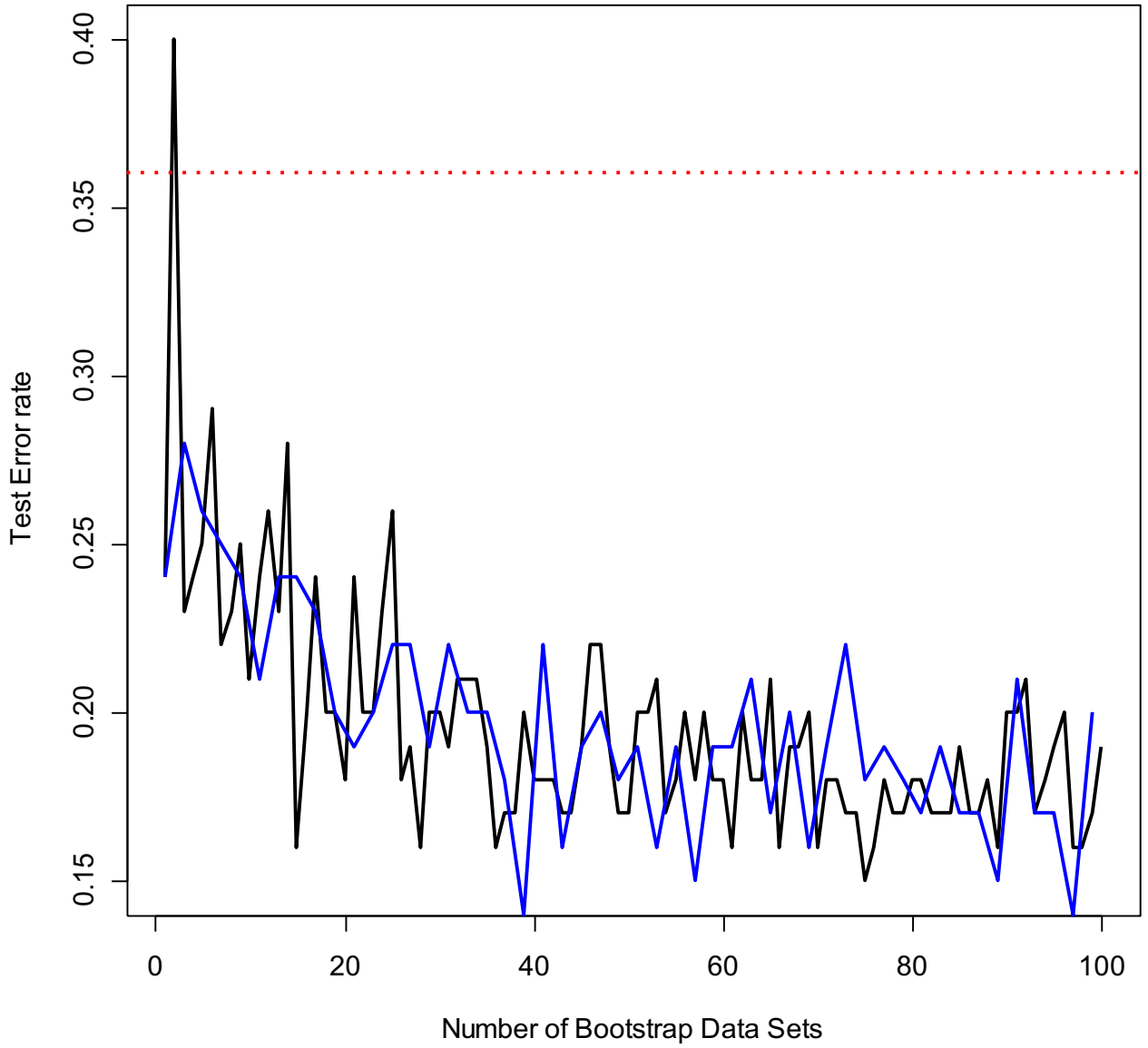
Procedure:

- Create B bootstrapped training datasets
- Use the datasets to construct B classification trees and generate predictions with each one
- Make the overall prediction using one of these two approaches (both tend to work well):
  - Majority vote – choose the class voted (i.e., predicted) by the majority of the bagged classification trees
  - If the bagged trees produce probability estimates, average the probabilities and then predict the class with the highest probability

Example 2: Car Seat Data

The red line represents the test error rate using a single tree

The black line corresponds to the bagging error rate using majority vote, while the blue line averages the probabilities.



Out-of-Bag Error Estimation

- Since bootstrapping involves random selection of subsets of observations to build a training data set, then the remaining non-selected part could be used as the testing data
- On average, each bagged tree makes use of around 2/3 of the observations, so we end up having 1/3 *Out-of-Bag* (OOB) observations that can be used for testing
- This approach is particularly convenient when performing bagging on large data sets for which cross-validation would be computationally very demanding

Variable importance measure

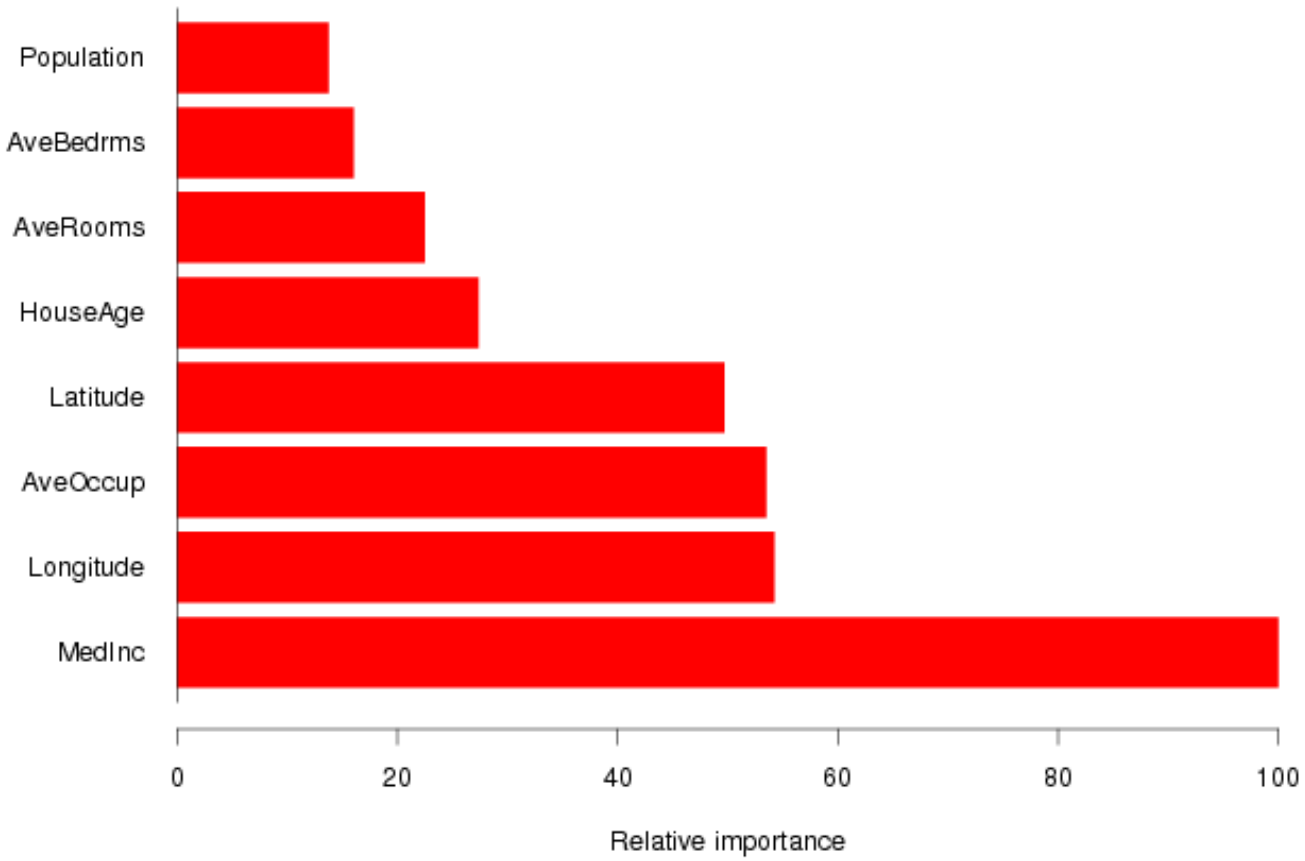
- Bagging typically improves the accuracy over prediction using a single tree, but this comes at the expense of having a model that is hard to interpret
- We have hundreds of trees, and it is no longer clear which variables are most important to the procedure, as it is impossible to plot the model
- But, we can still get an overall summary of the importance of each predictor using Relative Influence Plots

Relative Influence Plots

- Relative influence plots allow for identifying variables that are the most useful in predicting the response
- These plots give a score for each variable
- The larger the score the more influence the variable has
  - a number close to zero indicates the variable is not important and could be dropped

Example:
Relative Influence Plot for Housing Data

- Median Income is by far the most important variable
- Longitude, Average occupancy, and Latitude are the next most important



Relative Influence Plots

- The scores represent
  - the total decrease in MSE that is due to the splits on a particular variable (predictor), in case of regression problem
  - the total decrease in the Gini index or Cross-entropy, in the classification case

Recommendation

For an excellent explanation of what bagging is, why it tends to produce good ML models, and the like, watch the tutorial by Alexander Ihler, available on YouTube:

https://www.youtube.com/watch?v=Rm6s6gmLTdg

RANDOM FORESTS

Random Forests

- A very efficient statistical learning method
- Builds on the idea of bagging, but includes a small tweak that de-correlates the trees, and leads to improved performance

Random Forests

- How does it work?
  - Create a number of bootstrapped training samples to be used for building a number of decision trees (e.g. 500)
  - When building these trees, each time a split in a tree is considered, instead of all *p* predictors, a random sample of *m* predictors is chosen as split candidates from the full set of *p* predictors (usually $m \approx \sqrt{p}$)

Random Forests

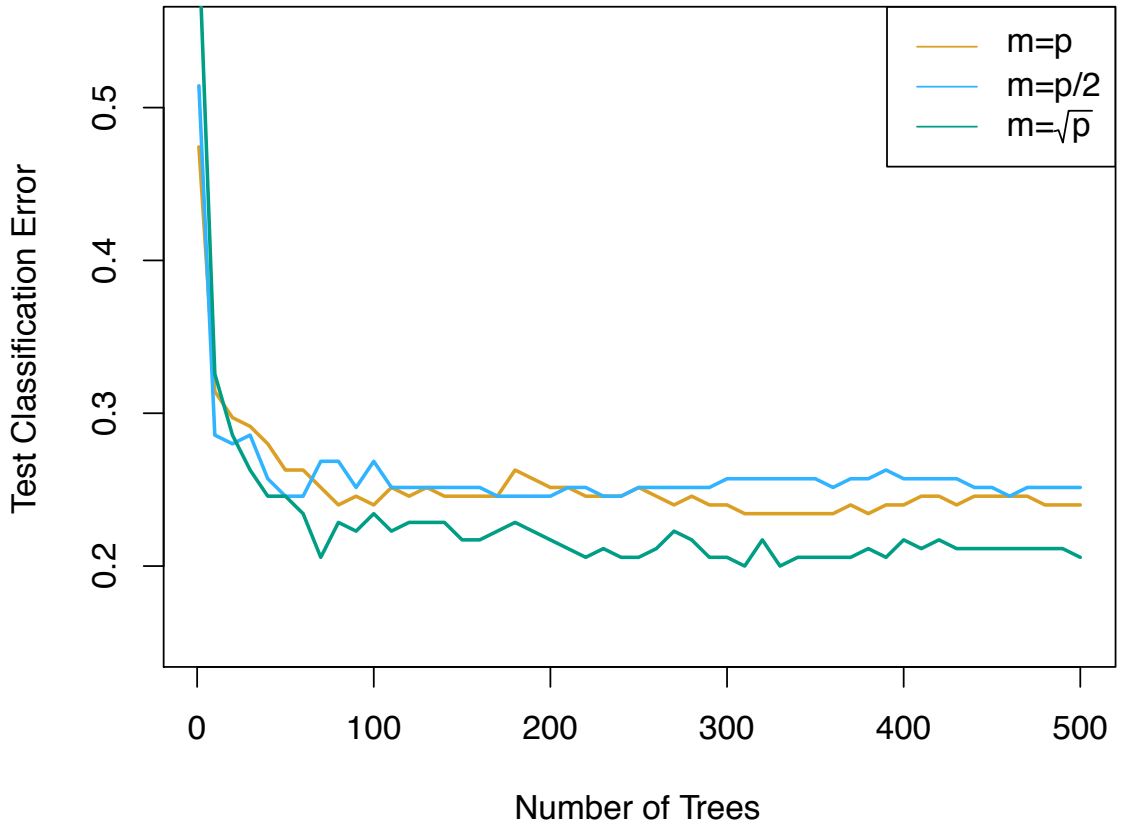
Why considering a random sample of predictors for each split?

- Suppose there is a very strong predictor in the data set along with a number of other moderately strong predictors;
- In that case, in the collection of bagged trees, most or all of them will use the very strong predictor for the first split
- All bagged trees will look similar => predictions from the bagged trees will be highly correlated
- Averaging many highly correlated quantities does not lead to a large variance reduction
- Therefore, random forests “de-correlates” the bagged trees leading to higher reduction in variance

Random Forests

Random Forest with different values of “m”

Notice when random forests are built using m = p, then this amounts simply to bagging.



Acknowledgement:
These slides are based on the slides prepared for the course “Applied Modern Statistical Learning Techniques” (link) and book “Introduction to Statistical Learning” (link)