Random Forests

Objectives

- Thoroughly explain the algorithm for constructing a random forest
- Explain why Random Forests are more accurate than a single decision tree, in terms of bias and variance.
- Compute the feature importance for a random forest model
- Explain out-of-bag error

Decision Trees – Quick Review

- Advantages
  - Model nonlinear relationships.
  - Easily deal with continuous or categorical data*
  - No feature scaling necessary
  - Easily handles missing values*
  - ***Highly interpretable***
- Disadvantages
  - Expensive to train
  - Often poor predictors

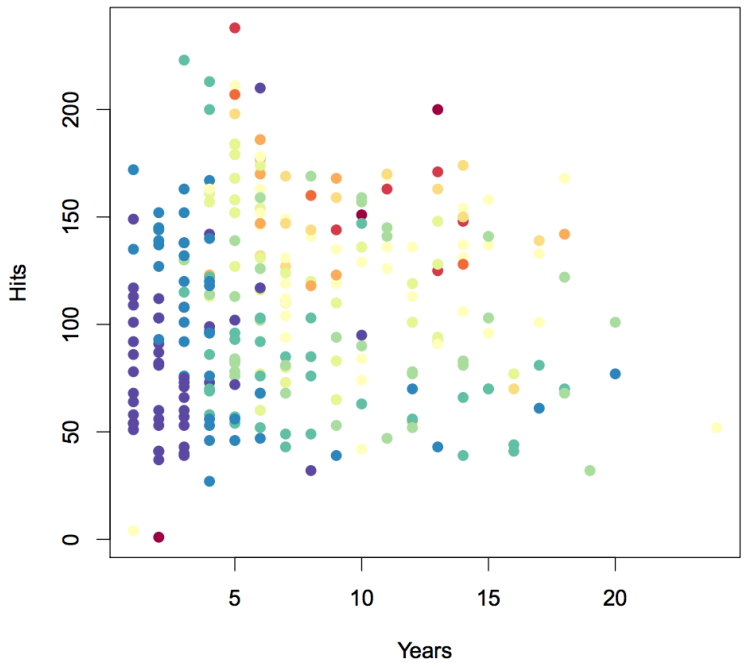
* Not in Python ☹

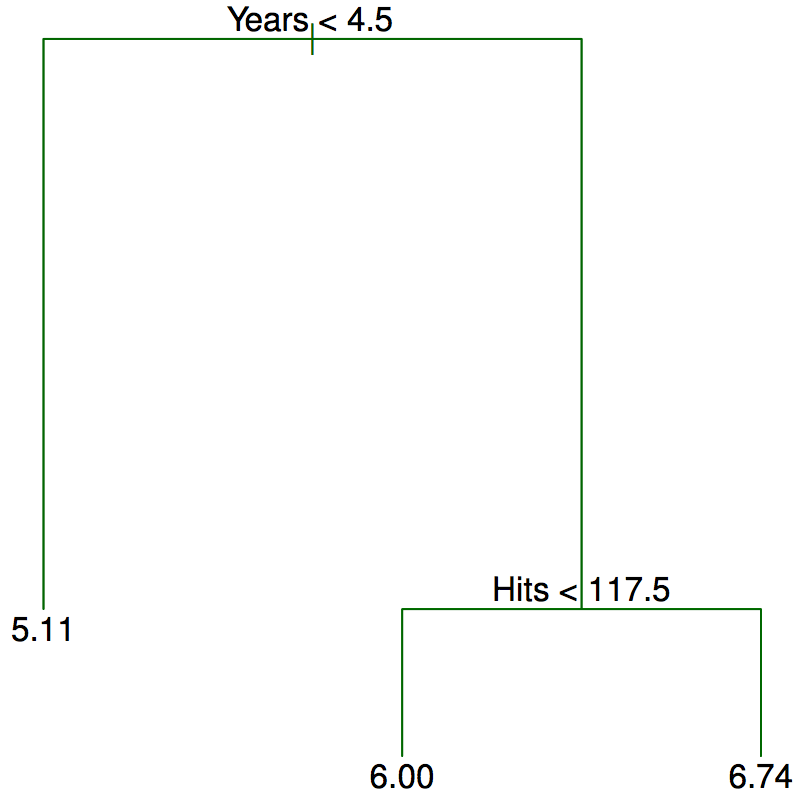
Decision Trees – Regression

Baseball salaries:

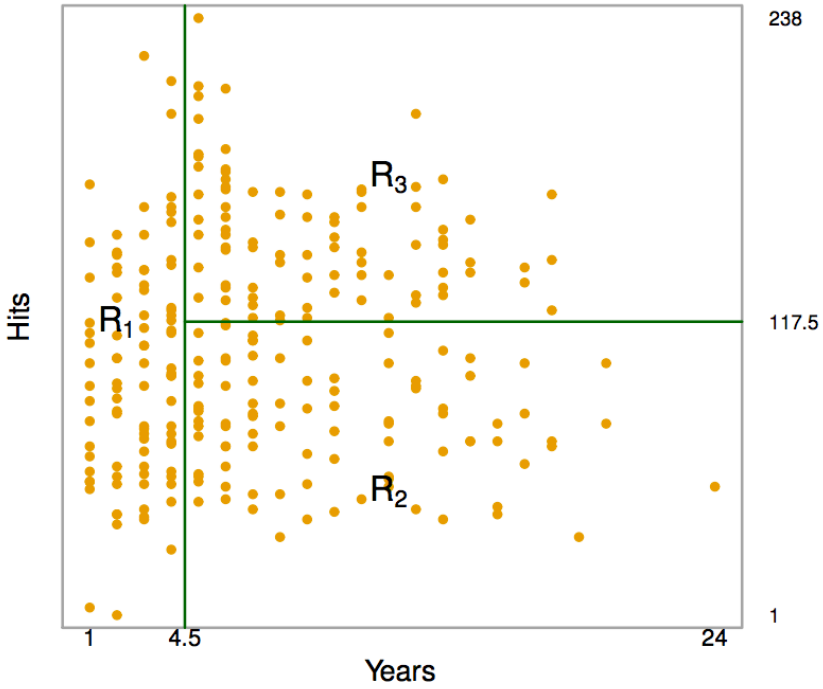
(Blue, Green) for low salaries

(Yellow, Red) for high salaries





Decision Trees – Regression

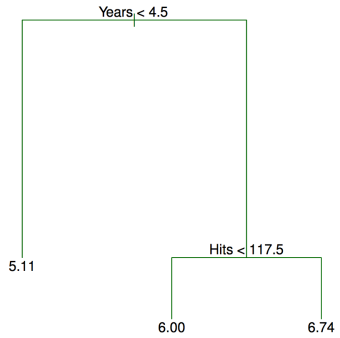


At each split, we aim to minimize:

$$\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2$$

Decision Trees – Classification

Making Predictions



At each terminal node (or rectangular region), predict

- Regression: Average
- Classification: Most commonly occurring class

How to split?

At each potential splitting node, minimize (in terms of information gain)

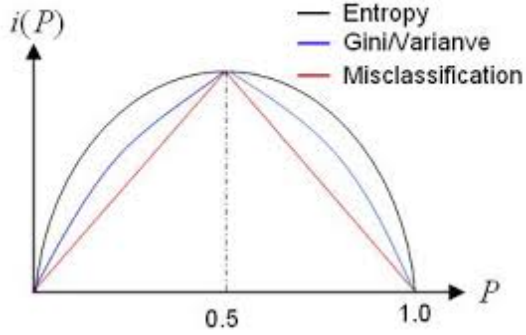
- Regression: RSS
- Classification:

Classification Error Rate $E = 1 - \max_k(\hat{p}_{mk})$

Gini index $G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$

Cross-entropy $D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$

$\hat{p}_{mk}$ is proportion in m-th region in k-th class



Bagging

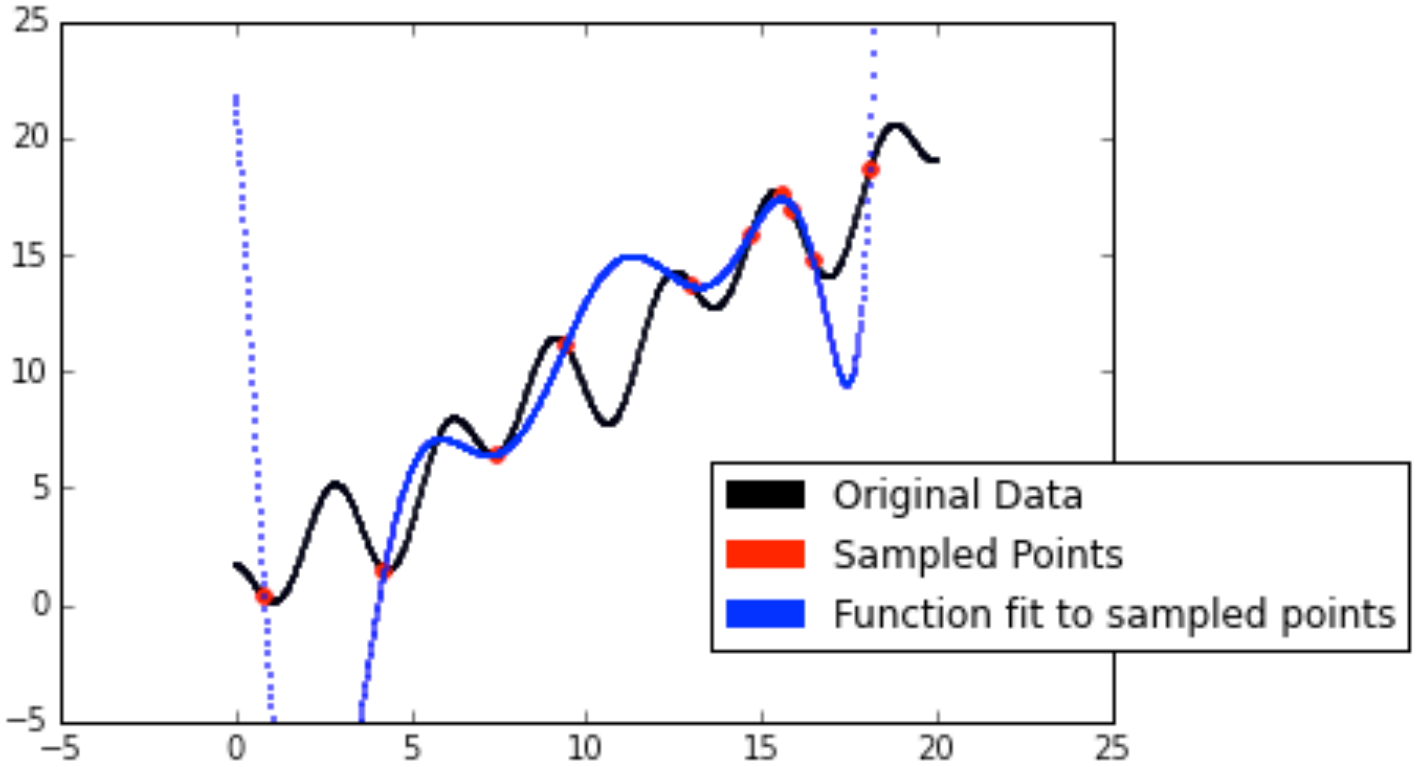
- Previously we looked at post-pruning our single decision tree to attack the variance

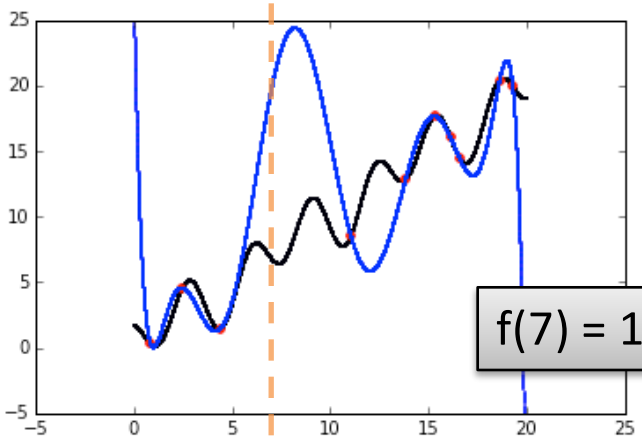
- Instead, we can just grow many large "bushy" trees and average away the variance (*central limit theorem*) by growing lots of trees (*bootstrapping*)!

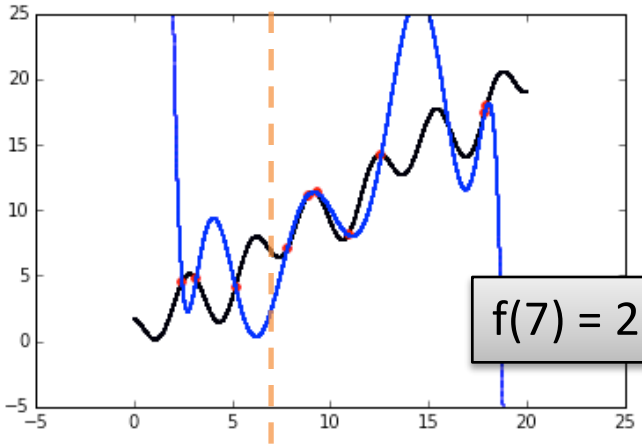
More Intuition

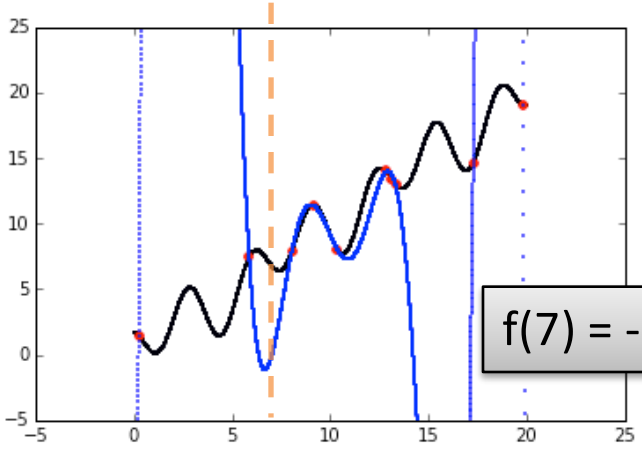
- Generate some sample data from a complex function (black points)
  - $f(x) = \sin(x) - 2\sin(2x - 1) + x$
- Pick 10 points at random from the sample and fit a degree 8 polynomial to the sample.

- **High Variance:** This model is highly sensitive---a slightly different dataset (choice of 10 points) will yield a very different model
- **Low Bias:** The model is complex enough---it just needs to see more data to make better predictions.



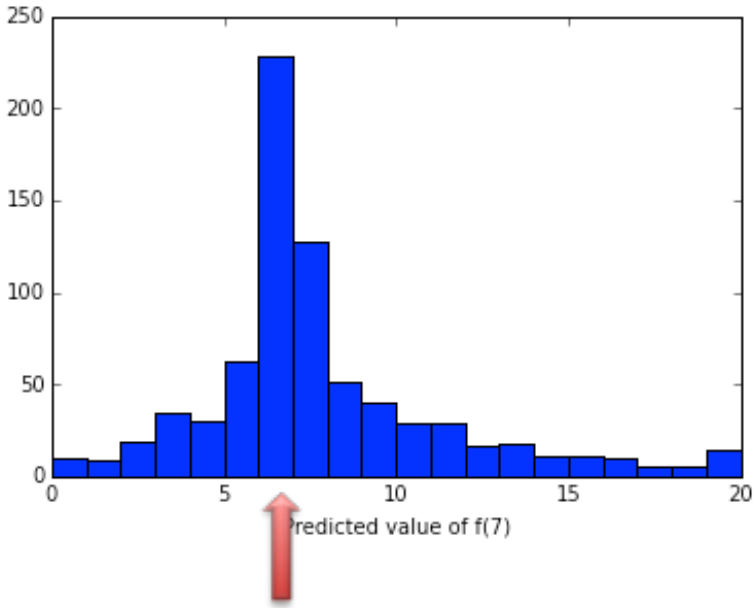






Repeat this process three times to get 3 very different functions (think *high variance*) with 3 very different predictions for f(7)

Repeat the process 1000 times and *on average* the prediction for f(7) is good (think *low bias)*.



Actual value: f(7) = 6.8

Bagging

Training

- Take B bootstrap samples from your data
- Build a decision tree on each sample (technically, any weak classifier)

Prediction

- Regression: Average prediction of B trees

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

- Classification: Majority vote among B trees

Error Estimation

- Since bootstrapped, each tree only uses about 2/3 of observations → remaining 1/3 can be used to estimate OOB (out-of-bag) error. Like Test-error!

Bias-Variance "Tradeoff"

Bias

- Deep tree → Relatively low bias
- Expectation of average of B trees same as expectation of any one of the trees

Variance

- Where we really win!
- Average of *B* i.d. (identically distributed) random variables, with pairwise correlation ρ, has variance…

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

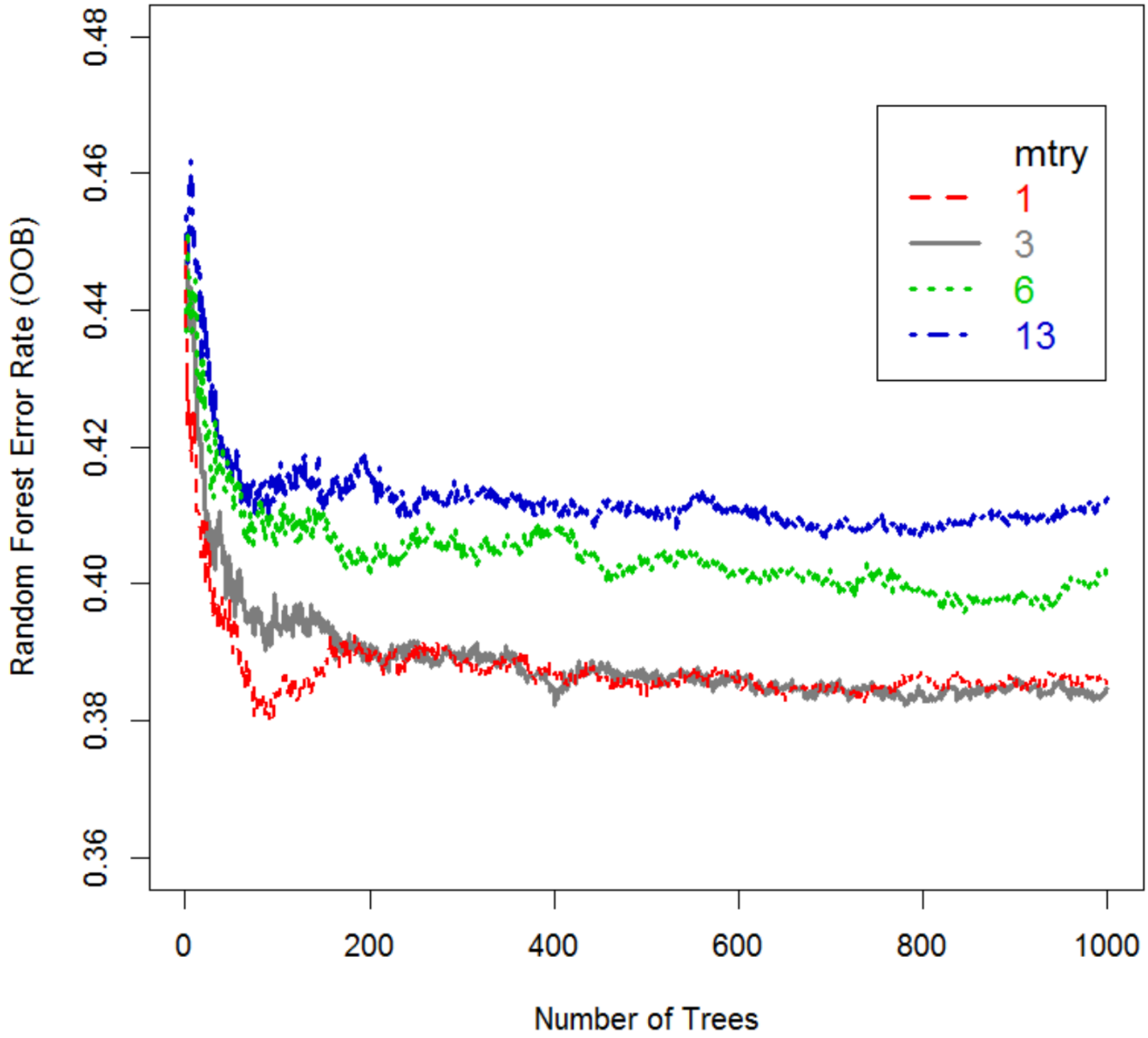
What happens as *B* increases?
What does *ρ* depend on?

Random Forests

Same idea except at each split considered choose a random selection of m predictors

Typically $m \approx \sqrt{p}$ so that if you have 100 predictors, you randomly 10 candidate features at **each** split point.

This "decorrelation" of the trees leads to improved performance over bagging.


Random Forest Error Rate (OOB)
0.36
0.38
0.40
0.42
0.44
0.46
0.48
0
200
400
600
800
1000
Number of Trees
mtry
1
3
6
13

Random Forest Parameters

- Total number of trees
- Number of features to use at each split
- Number of points to grab for each sample
- Individual decision tree parameters
  - Usually the individual estimators are grown to maximum depth. Remember, each estimator should have low bias, and the RF will deal with the variance.

In general, RF are fairly robust to the choice of parameters and over-fitting.

Tuning

Classification

$$m = \sqrt{p}$$

minimum node size = 1

`"max_features"`
`"min_samples_leaf"`

Regression

$$m = p/3$$

minimum node size = 5

Suggested defaults by the inventors.
But in practice you can tune these just like you did for λ in Lasso/Ridge!

Afternoon

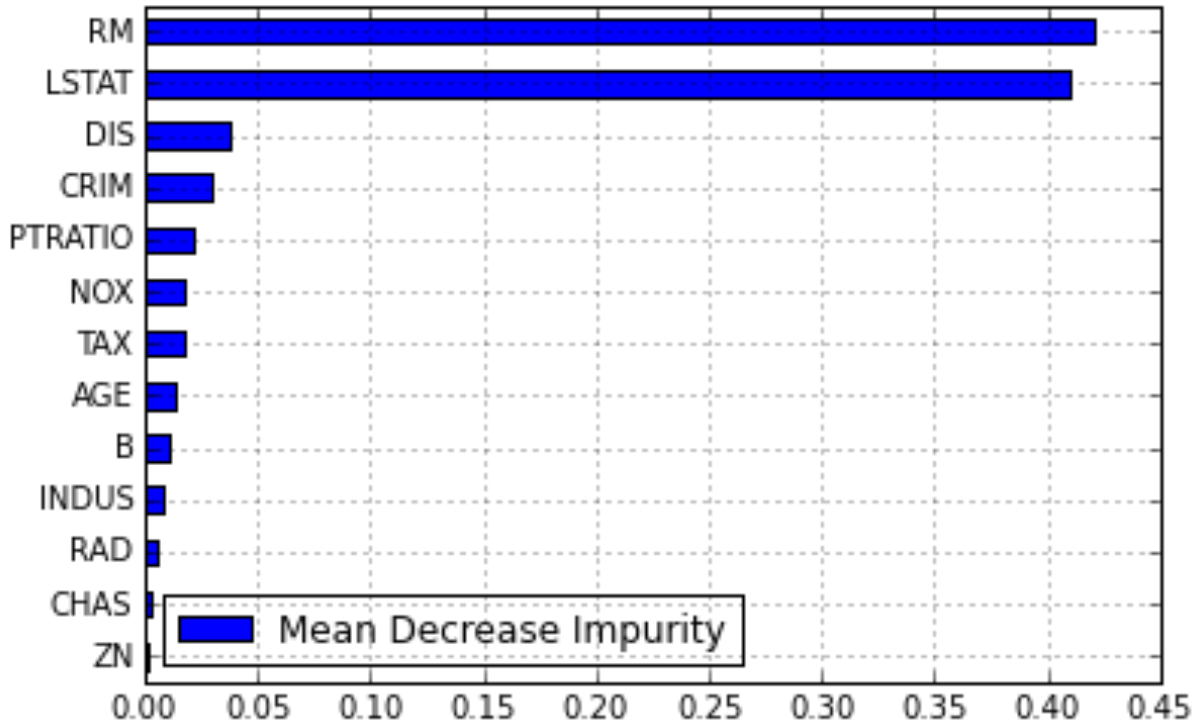
Model Interpretation

- Model Interpretability is the most commonly cited drawback of using Random Forests.
- The standard approach to interpretation is to measure the importance of each variable. Two methods exist:
  1. Mean Decrease Impurity
  2. Mean Decrease Accuracy
- Another approach is to analyze decision paths for individual data points.

Mean Decrease Impurity

- For each tree, each split is made in order to reduce the total impurity of the tree (Gini Impurity for classification, RSS for regression); we can record the magnitude of the reduction.
- Then the importance of a feature is the average decrease in impurity across trees in the forest, as a result of splits defined by that feature.
- **GOAL:** To determine which features have the largest impact on the model.

MDI - Boston Housing data



The biggest factors in predicting the value of a neighborhood are the average number of rooms and the class status of the neighborhood

Mean Decrease Accuracy

Alternative way to calculate variable importance

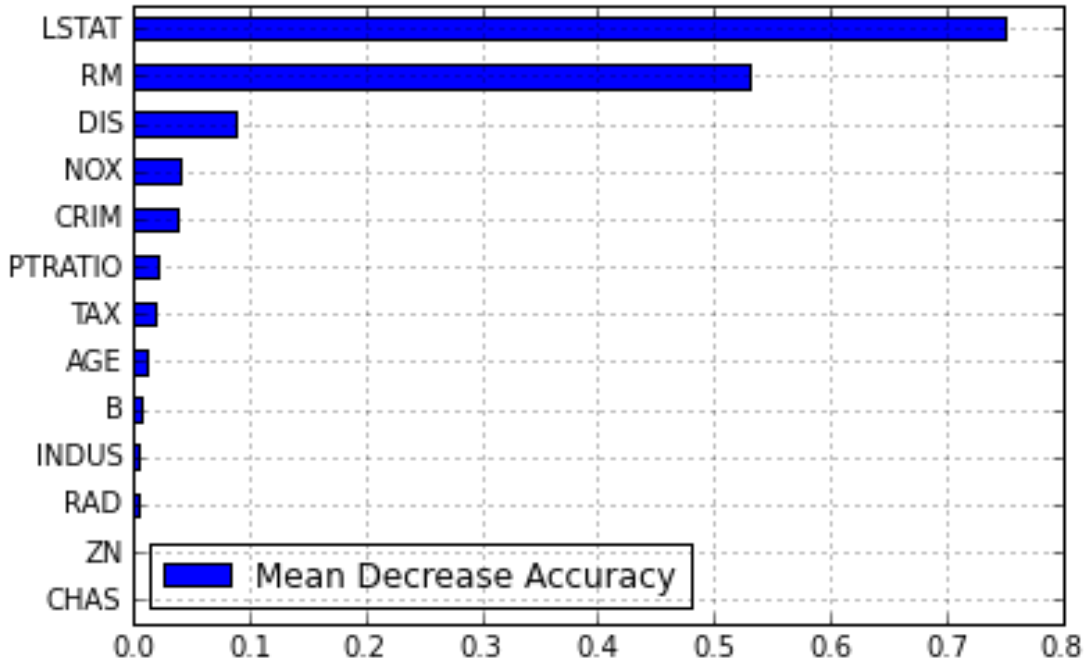
To evaluate importance of *j*th variable…

(1) When *b*th tree is grown, OOB samples passed down through tree → record accuracy

(2) Values of *j*th variable randomly permuted in OOB samples → compute new (lower) accuracy

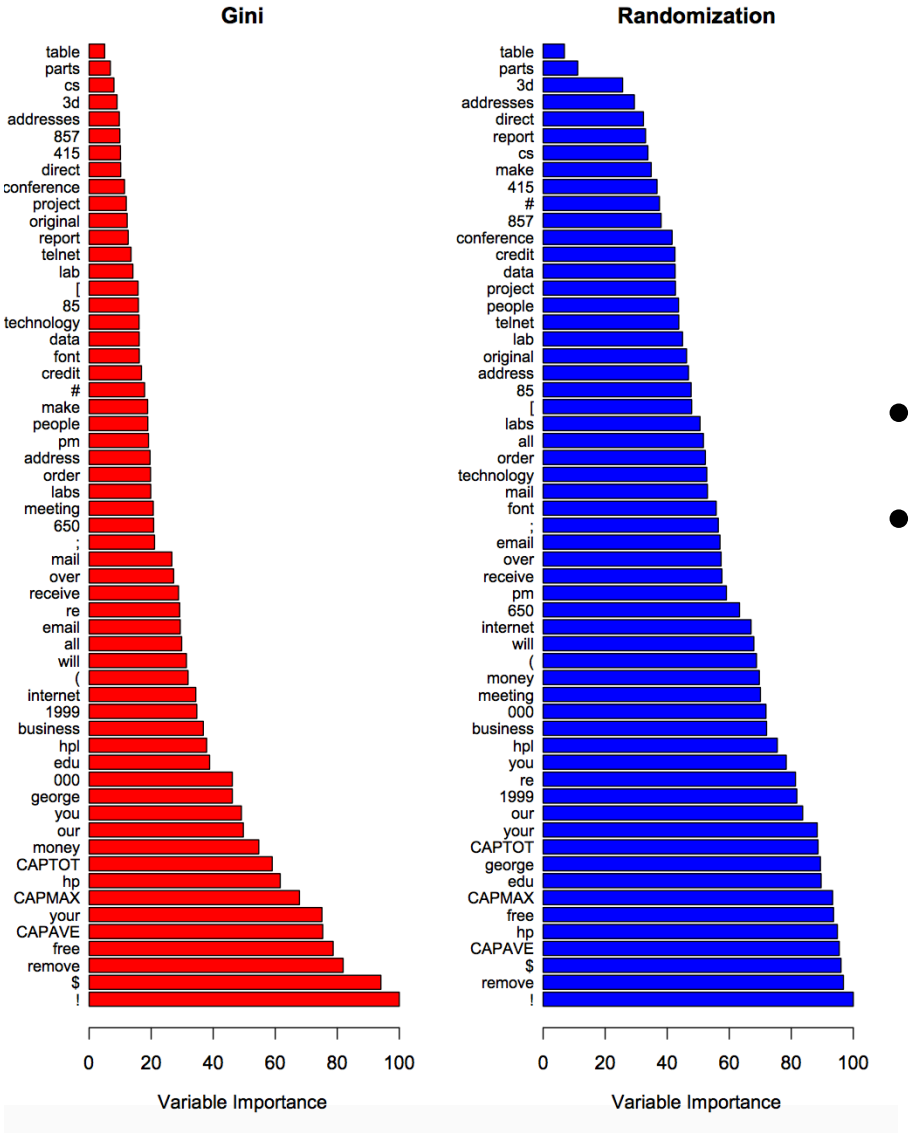
→ Average decrease in accuracy over all trees

MDA - Boston Housing Data



Results are similar to MDI, but some of the relative magnitudes are different

Comparison of Feature Importances



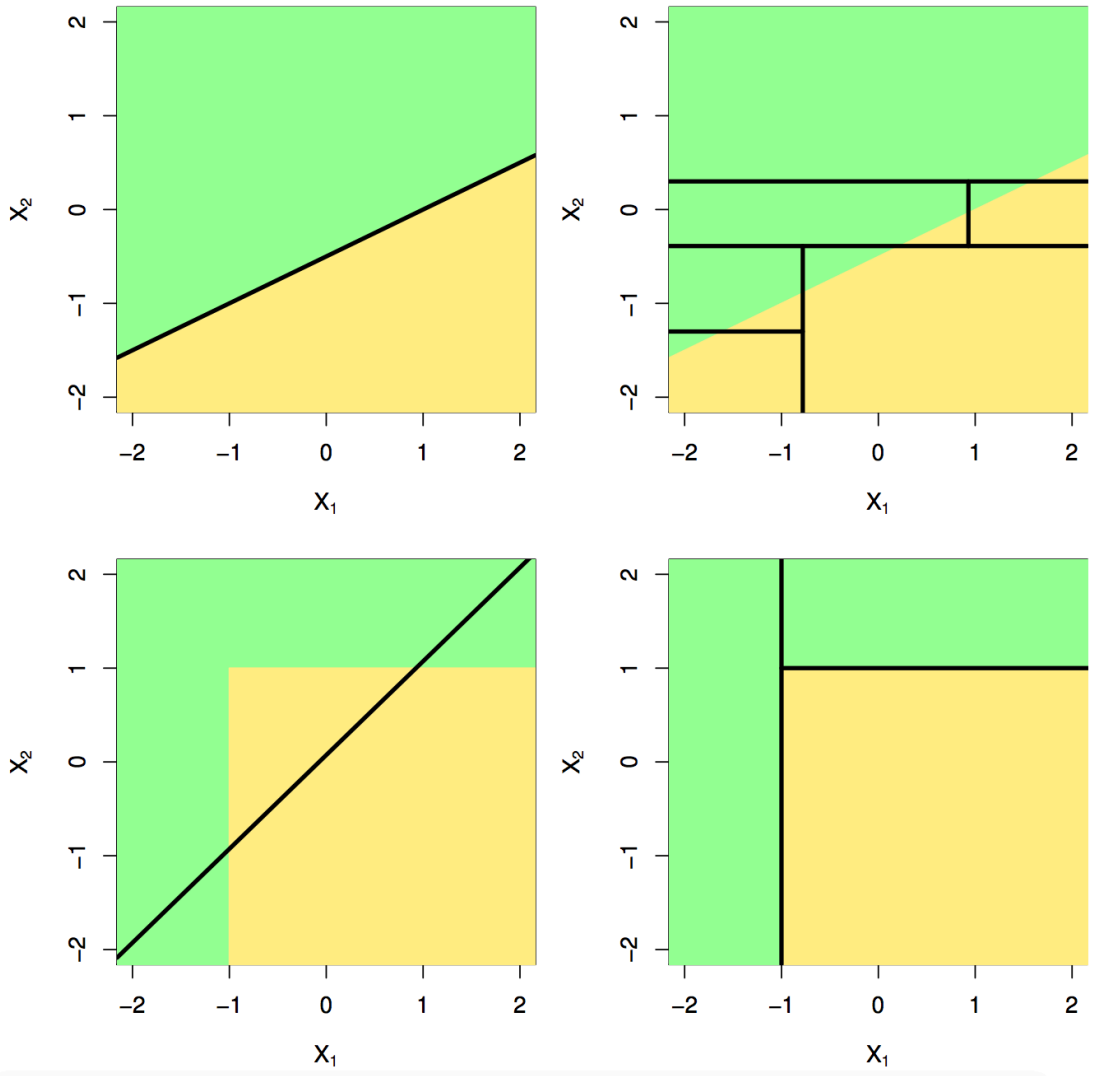
- Note similarity in rankings
- More even distribution for Randomization (2nd way)

Feature importance in sklearn

http://scikit-learn.org/stable/modules/ensemble.html#feature-importance-evaluation

- Basically, the higher in the tree the feature is, the more important it is in determining the result of a data point.
- The expected fraction of data points that reach a node is used as an estimate of that feature's importance for that tree.
- Finally, average those values across all trees to get the feature's importance.

Trees Revisited



Random Forest – Quick Review

- Advantages
  - Model nonlinear relationships.
  - Easily deal with continuous or categorical data*
  - No feature scaling necessary
  - Easily handles missing values*
  - ***Somewhat interpretable***
  - Prediction accuracy on par with cutting edge algorithms (Kinect uses RF for face detection)
- Disadvantages
  - Expensive to train

* Not in Python ☹

Questions

- Describe the random forest algorithm, step by step
  - How to build single tree?
  - How many to build?
  - How does final classification/regression estimate happen?
- What happens as number of trees, *B* increases, for bagging or random forest?
- Why does Random Forest outperform bagging?
- How does the Random Forest “win” at the Bias-Variance

Questions

- Describe the random forest algorithm, step by step
  - How to build single tree? Deep tree generally, since will average away variance
  - How many to build? Many as computationally reasonable
  - How does final classification/regression estimate happen? Majority/Average
- What happens as number of trees, *B* increases, for bagging or random forest? Variance decreases. Greater the B, the better, although there are diminishing returns after a certain point. Also incurring some perhaps undesirable computational cost
- Why does Random Forest outperform bagging?

  "Decorrelate" the trees through random selection of *m* at each node.

  $$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$
- How does the Random Forest "win" at the Bias-Variance

  Mostly wins through "averaging away" the variance. Again $\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$