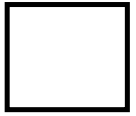
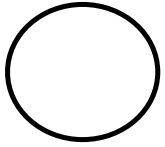
Idea

Define a decision tree in an optimal way for regression and classification

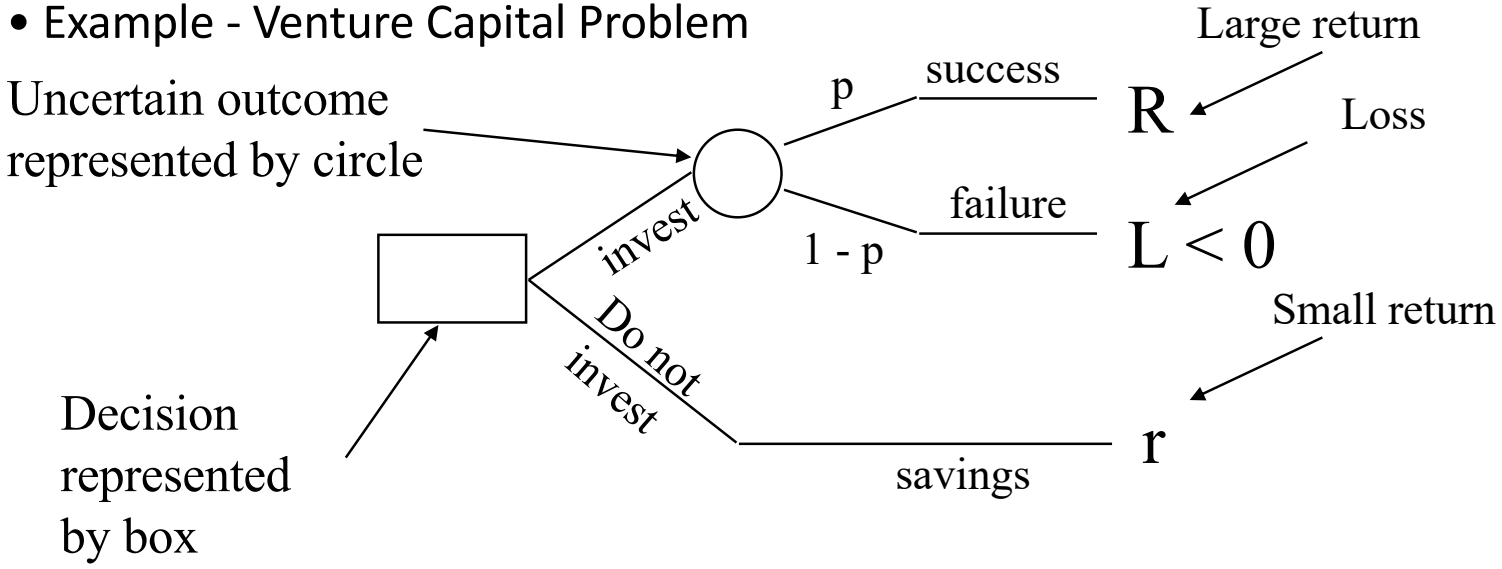
Decision trees

- Decision Node 
- Chance Node 
- Branch/sequence

Decision tree

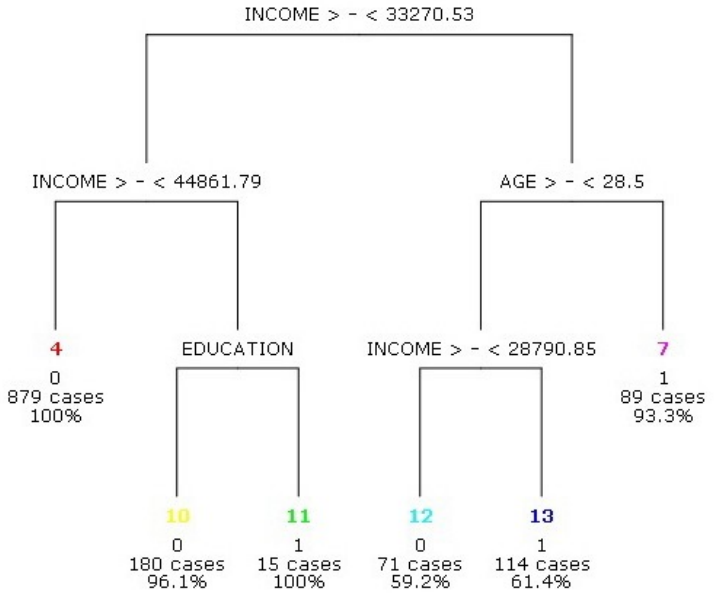
Decision trees

- Probabilities Account for uncertainty
  - Used to evaluate expected values
  - Example - Venture Capital Problem



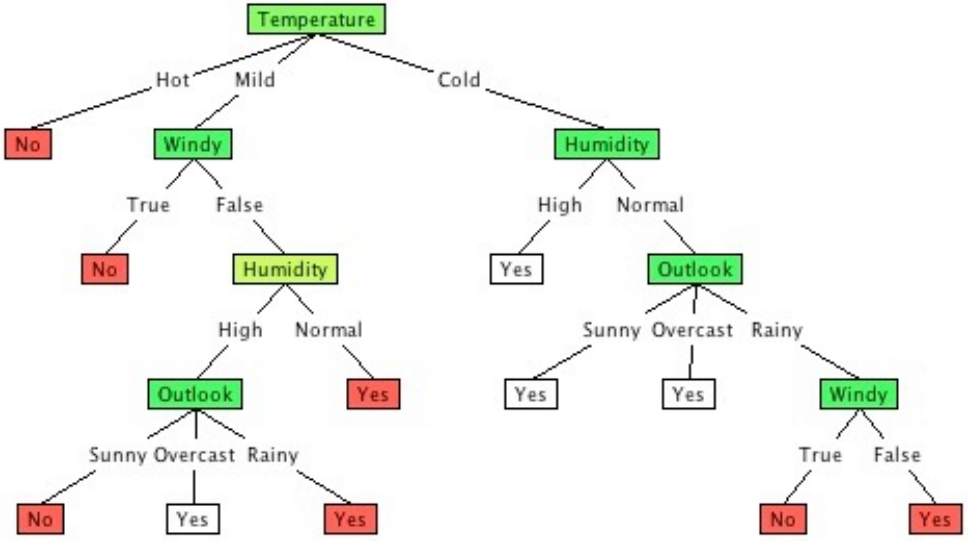
Decision trees

Tree consists of the root node, decision node and terminal node



Decision trees

Tree consists of the root node, decision node and terminal node



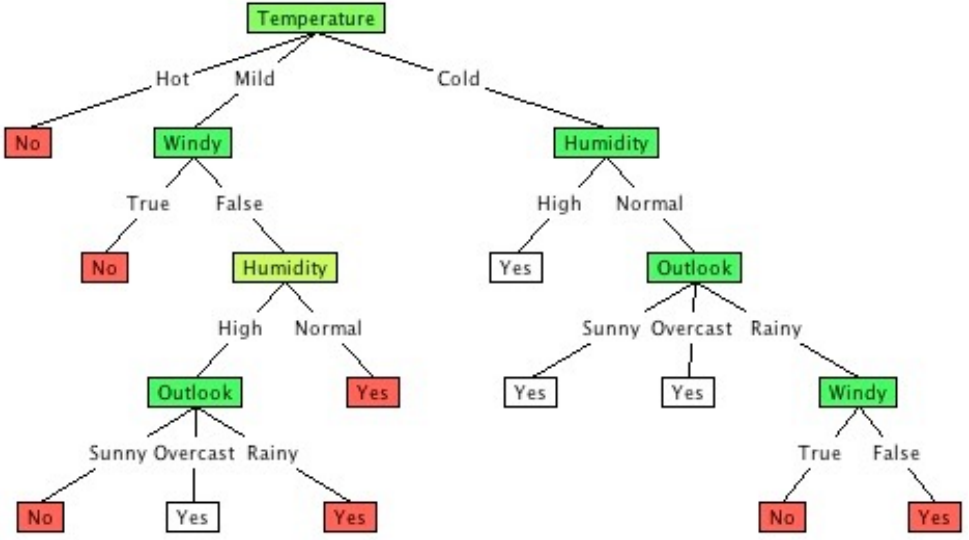
Decision tree

Decision trees

Structure

A decision tree consists of

- **Nodes**
  Tests for variables
- **Branches**
  Results of tests
- **Leaves**
  Classification



Decision trees

When and how do we split branches?

- Done on different features (e.g. humidity)
- Metrics: Gini index $\text{Gini} = \sum_{i \neq j} p(i)p(j)$
  or cross-entropy

When do we stop?

- See next

Decision tree tend to be very complex and overfitted - need a way to reduce this complexity

Decision tree

Decision trees

What can a decision tree do?

- Classification (obvious)
- Non-linear functions → exponentially large trees

How can we produce smaller decision trees?

Smaller trees are better – avoid overfitting

How do we build small trees that accurately capture data?

Decision tree

Decision trees

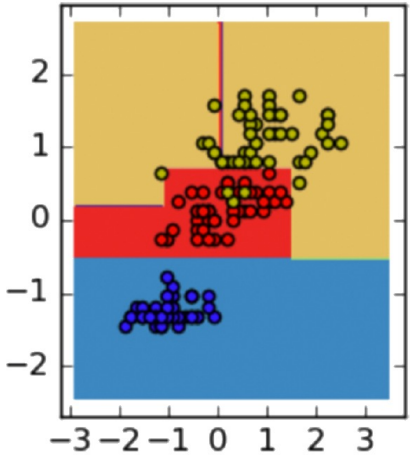
What can a decision tree do?

- Classification (obvious)
- Non-linear functions → exponentially large trees

How can we produce smaller decision trees?

Smaller trees are better – avoid overfitting

How do we build small trees that accurately capture data?



Feature selection

The quantification of information

Founded by Claude Shannon

Simple concepts:

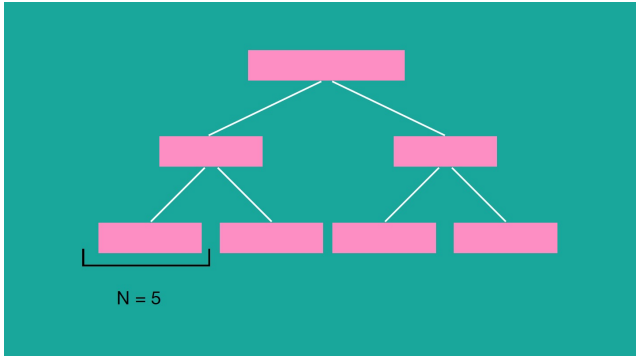
- Entropy: $H(X) = -\sum_x \mathbb{P}(X = x) \log \mathbb{P}(X = x)$
- Conditional Entropy: $H(Y|X) = \sum_x \mathbb{P}(X = x) H(Y|X = x)$
- Information Gain: $\mathrm{IG(Y|X)} = \mathrm{H(Y)} - \mathrm{H(Y|X)}$

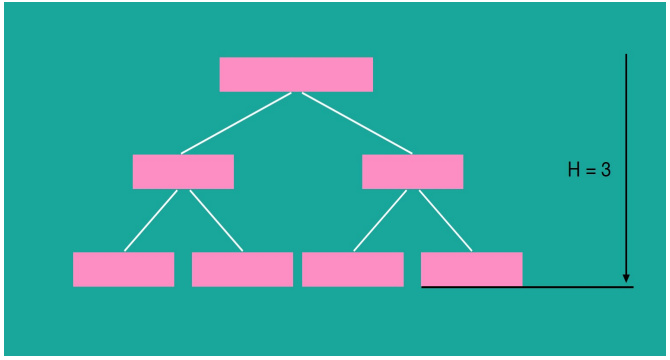
Select the variable with the highest information gain

Decision tree

Avoiding overfitting

1. Early stopping based on various criteria



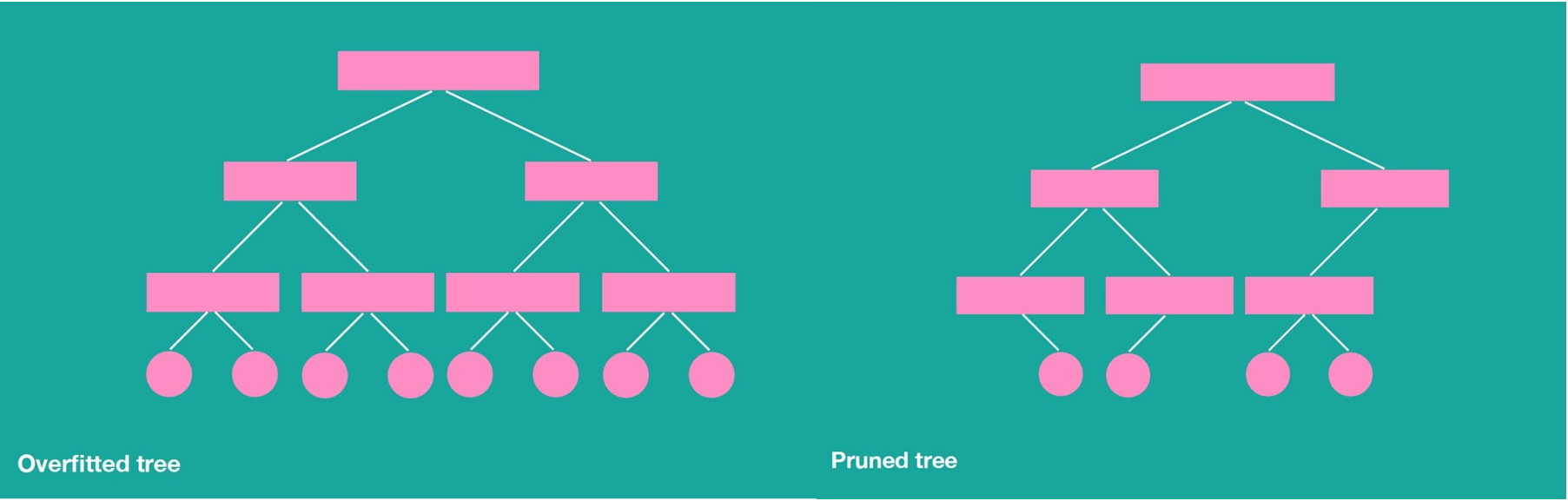


2. Tree pruning

Decision tree

Avoiding overfitting

2. Tree pruning



Decision tree

Problem with trees

With small trees: **large variance**

i.e. two realizations can be very different

Single one can have **large bias**

How do we solve that?

Combination of different decision trees show much better results

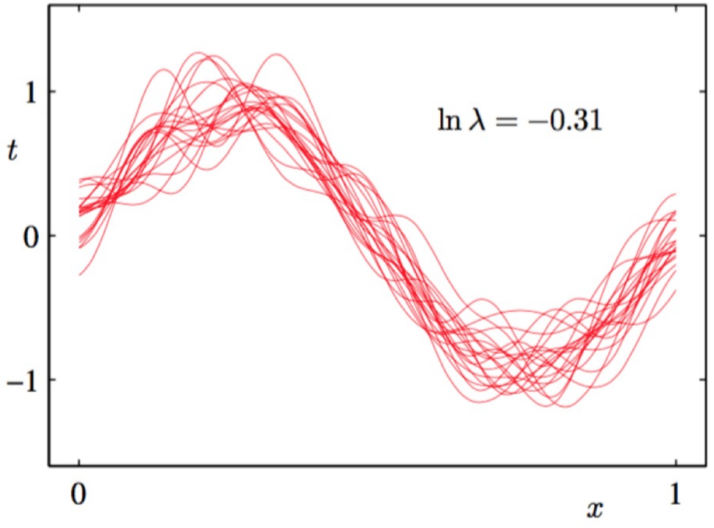
Constructing trees with low biases

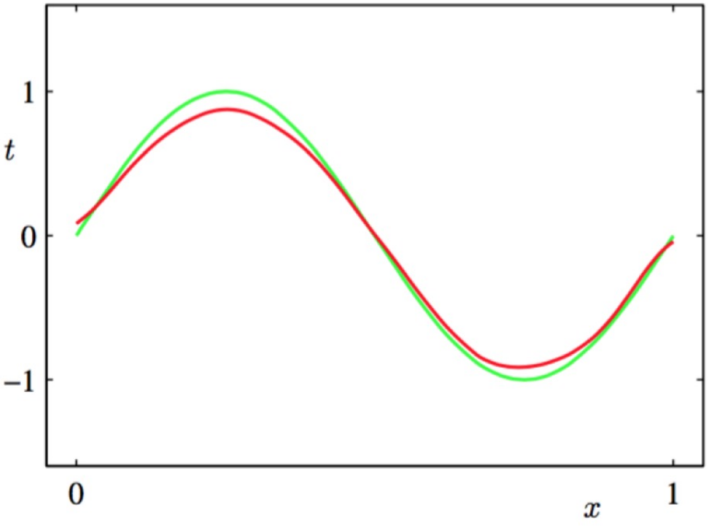
Decision trees are characterized by low bias but high variance even with a small changes in training sample.

$$
\begin{aligned}
E\{(y-\hat{y})^2\} &= E\{(\epsilon + f(\mathbf{x}) - \hat{y})^2\} \\
&= E\{(\epsilon + f(\mathbf{x}) - E(\hat{y}) + -\hat{y} + E(\hat{y}))^2\} \\
&= \sigma^2 + (f(\mathbf{x}) - E(\hat{y}))^2 + \text{var}(\hat{y}) \\
&= \text{Irreducible error} + \text{bias}^2 + \text{variance}
\end{aligned}
$$

Constructing trees with low biases

Example of various trees – low bias but inter-tree large variance





Decision tree

Model averaging

Decision trees can be simple, but often produce noisy (bushy) or weak (stunted) classifiers.

- **Bagging** (Breiman, 1996): Fit many large trees to bootstrap-resampled versions of the training data, and classify by majority vote.
- **Boosting** (Freund & Shapire, 1996): Fit many large or small trees to reweighted versions of the training data. Classify by weighted majority vote.
- **Random Forests** (Breiman 1999): Fancier version of bagging. In general Boosting>Random Forests>Bagging>Single Tree.

Bagging

Model averaging

Bagging (Breiman, 1996):

•Motivation: Average a given procedure over many samples to reduce the variance.

•To bag C, we draw **bootstrap** samples S1, ..., SB each of size N from the training data. Then define tree

•Bagging can dramatically reduce the variance of unstable procedures (like trees), leading to improved prediction.

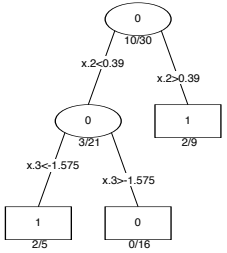
•All simple structures in a tree are lost.

Bagging

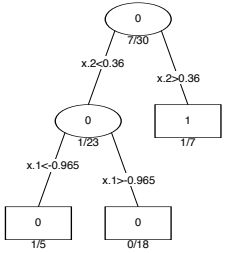
Model averaging

Bagging (Breiman, 1996):

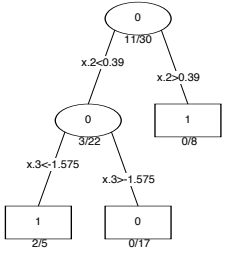
Original Tree



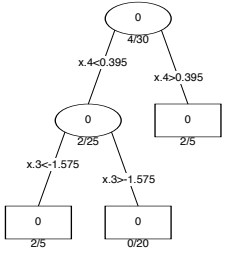
Bootstrap Tree 1



Bootstrap Tree 2



Bootstrap Tree 3



Random forest

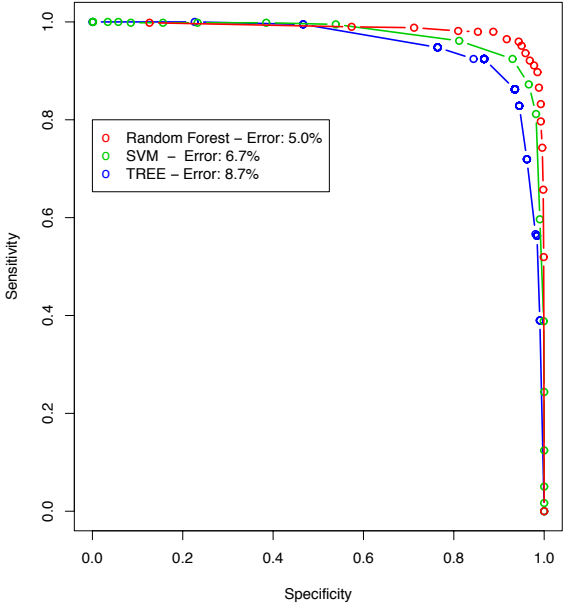
Model averaging

Random Forests (Breiman 1999):

- **Bagging features and samples** simultaneously:
- At each tree split, **a random sample of m features is drawn, and only those m features are considered for splitting**. Typically m = √d or log2 d, where d is the number of features
- For each tree grown on a bootstrap sample, the error rate for observations left out of the bootstrap sample is monitored. This is called the "out-of-bag" error rate.
- random forests tries to improve on bagging by "de-correlating" the trees. Each tree has the same expectation.

Model averaging

Random Forests (Breiman 1999):

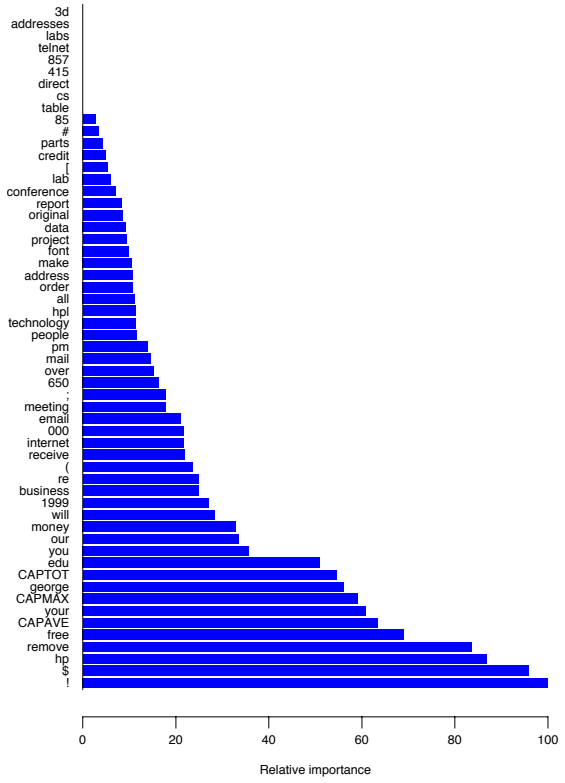


Random forest

Variable importance

•The bootstrap roughly covers 1/3 samples for each time.

•Do permutations over variables to check how important they are.



Boosting

Model averaging

- **Boosting** is an **ensemble method**, meaning it's a way of combining predictions from several models into one. It does that by taking each predictor sequentially and modelling it based on its predecessor's error (giving more weight to predictors that perform better):
- Fit a first model using the original data
- Fit a second model using the **residuals** of the first model
- Create a final/third model using the sum of models 1 and 2

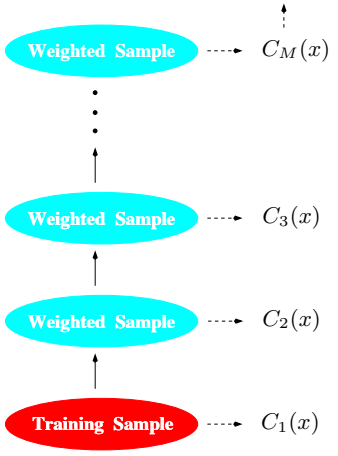
Boosting

Model averaging

Boosting (AdaBoost)

Combine weak learners into a single strong learner

- Average many trees, each grown to re-weighted versions of the training data (Iterative).
- Final Classifier is weighted average of classifiers:



Boosting

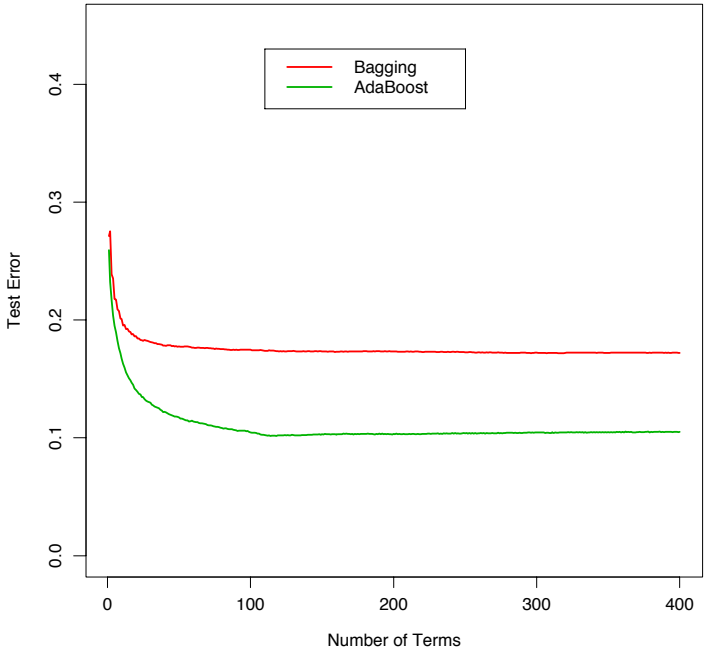
Model averaging

Boosting (AdaBoost)

- Simple classifier (decision stump) into 2 classes
- Error used for calculating weight given in final evaluation
- Intuitive sense – higher weight given for model with fewer errors
- Weights for each observations will be updated
- Weight will be increased for incorrectly classified observations: give more focused to next iteration, weights will be reduced for correctly classified observation

Model averaging

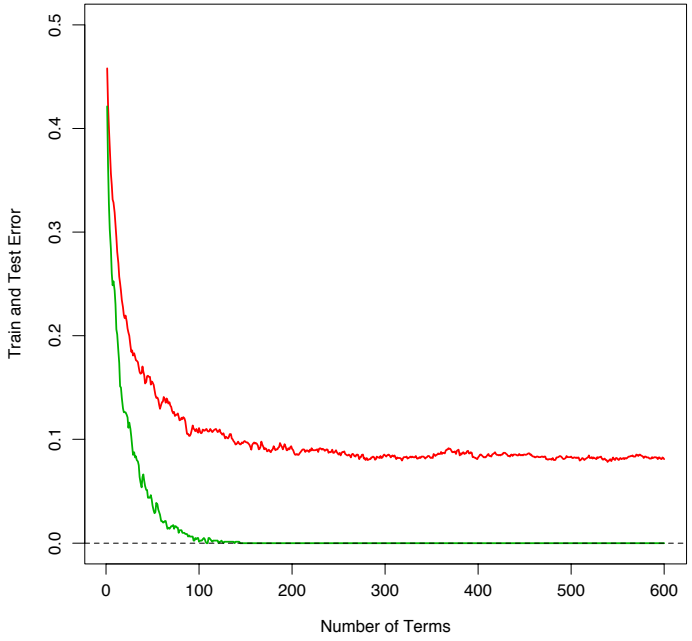
Boosting



Boosting

Model averaging

Boosting

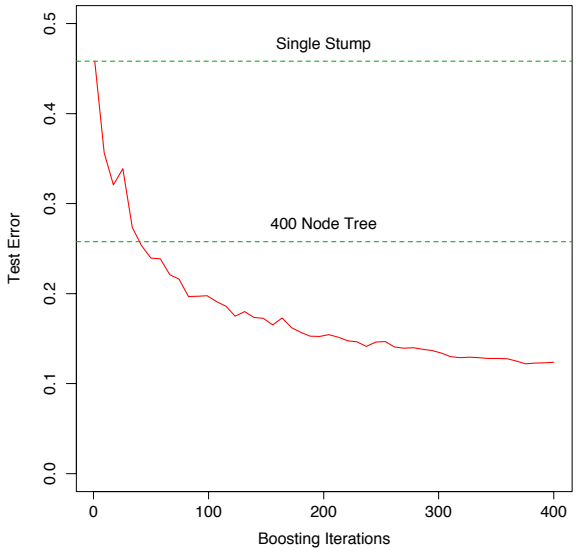


- More iterations continue to improve test error in many examples.
- The Adaboost is often observed to be robust to overfitting.

Boosting

Model averaging

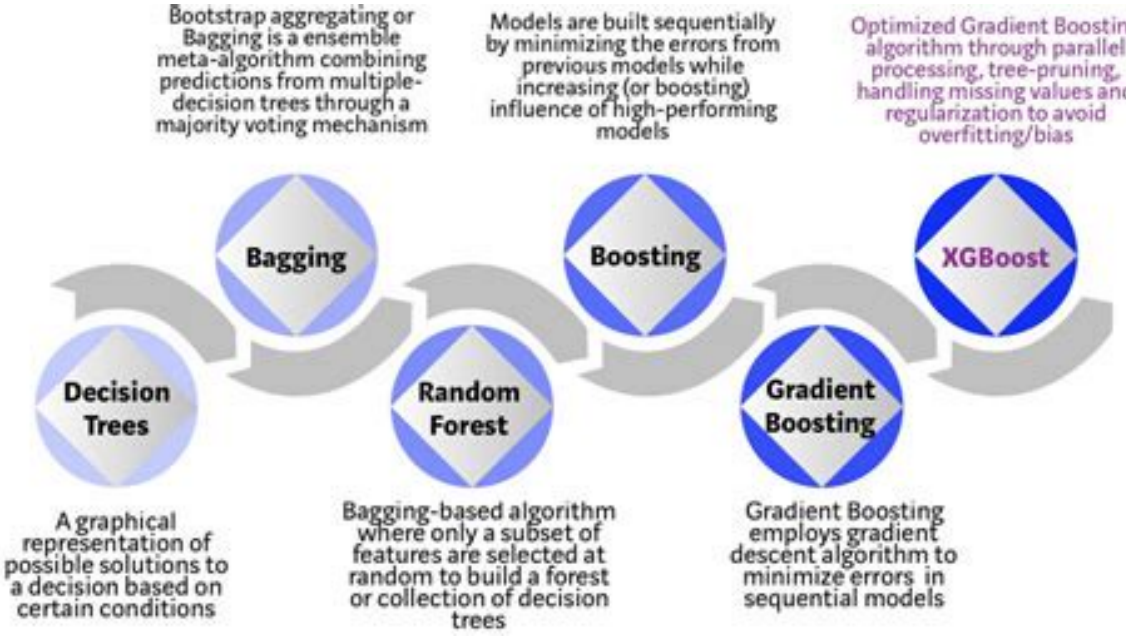
Boosting



The ensemble of classifiers is much more efficient than the simple combination of classifiers.

Extreme Gradient Boosting - XGBoost

Speed and performance – often among best performance ML algorithm



Boosting

Extreme Gradient Boosting - XGBoost

Speed and performance – often among best performance ML algorithm

Gradient boosting is a specific type of boosting, called like that because it minimizes the loss function using a **gradient descent algorithm**

→ Can use GPUs ☺

Trees summary

Powerful ML tool

But high variance

And memory intensive as need to save the trees!