

Tree Methods

Jiayu Zhou

¹Department of Computer Science and Engineering
Michigan State University
East Lansing, MI USA

March 22, 2016

Table of contents

- 1 Introduction
- 2 Tree Methods
 - Regression Tree
- 3 Random Forest
 - Procedure
 - Why Random Forest Works
 - Variable Importance
 - Compare to existing algorithms
- 4 Proximity Measures
 - Proximity and Visualization
 - Other Applications of Proximity

Introduction

Classification and Regression Trees

The Generic Tree Algorithm

- Grow a binary tree.
- At each node, “split” the data into two “daughter” nodes.
- Splits are chosen using a splitting criterion.
- Bottom nodes are “terminal” nodes.

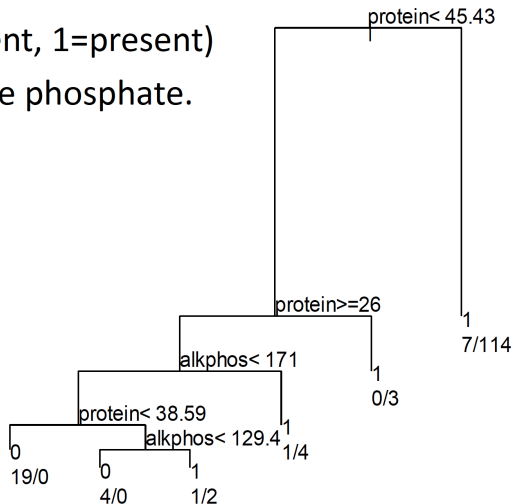
Remarks:

- For regression the predicted value at a node is the average response variable for all observations in the node.
- For classification the predicted class is the most common class in the node (majority vote).
- For classification trees, can also get estimated probability of membership in each of the classes

A Classification Tree

Predict hepatitis (0=absent, 1=present)
using protein and alkaline phosphate.

“Yes” goes left.



Splitting criteria

- **Regression:** residual sum of squares

$$\text{RSS} = \sum_{\text{left}} (y_i - y_L^*)^2 + \sum_{\text{right}} (y_i - y_R^*)^2$$

where:

- y_L^* = mean y -value for left node
- y_R^* = mean y -value for right node

- **Classification:** Gini criterion

$$\text{Gini} = N_L \sum_{k=1}^K p_{kL}(1 - p_{kL}) + N_R \sum_{k=1}^K p_{kR}(1 - p_{kR})$$

where:

- p_{kL} = proportion of class k in left node
- p_{kR} = proportion of class k in right node

Tree Methods

Regression: Prostate Cancer

Example: Prostate Cancer Prognosis

After tumor detection, there are many treatment options.

- Various chemo + radiation, surgical removal.

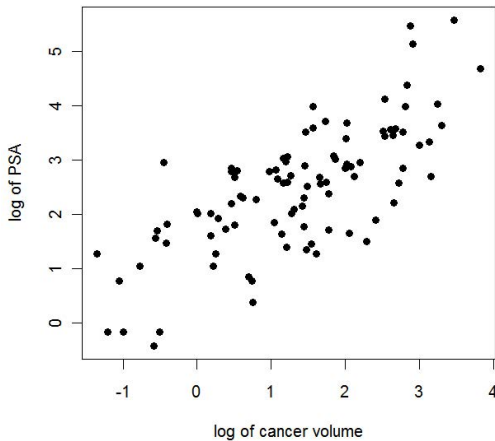
Biopsy information is useful to help in deciding treatment

- *Prostate Specific Antigen, in Log (lpsa)*: protein production. (target)
- *Log of Prostate Volume (lcavol)*: predictor
- *Log of Prostate Weight (lweight)*: predictor

Data: <http://www.biostat.jhsph.edu/~ririzarr/Teaching/649/prostate.html>

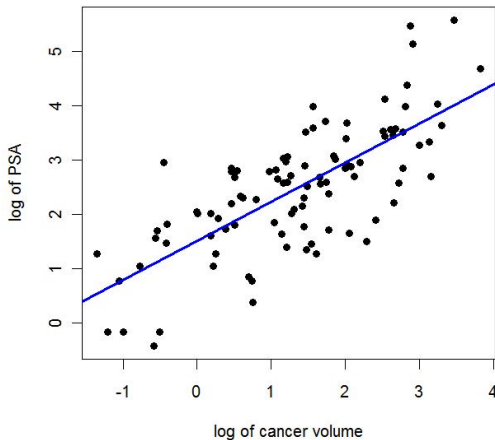
Regression: Prostate Cancer

Prostate Cancer Example: data



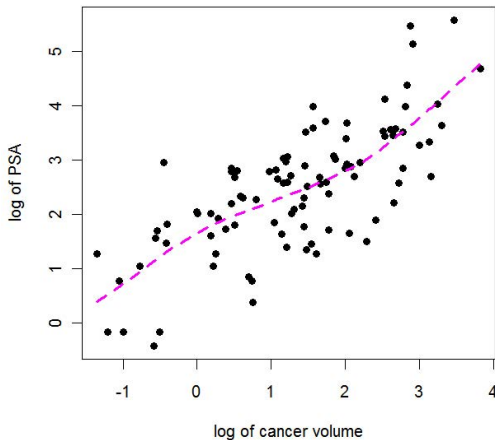
Regression: Prostate Cancer

Prostate Cancer Example: linear model

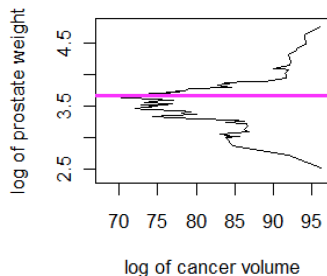
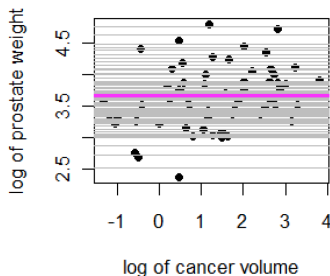


Regression: Prostate Cancer

Prostate Cancer Example: nonlinear model

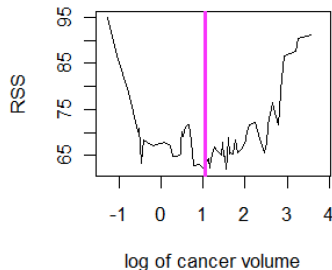
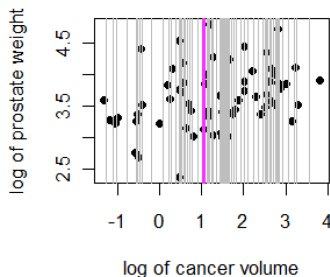


Choosing the best horizontal split



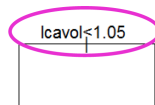
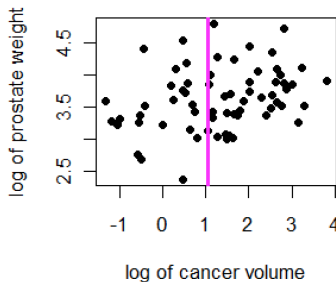
Best horizontal split is at 3.67 with $RSS = 68.09$.

Choosing the best vertical split



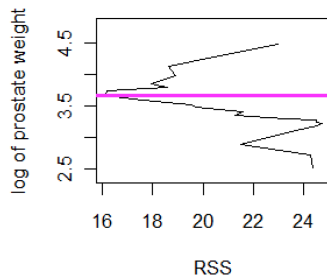
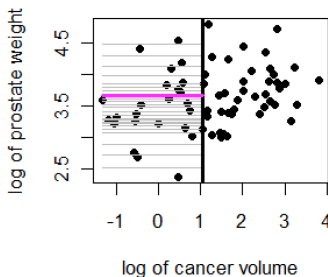
Best vertical split is at 1.05 with $RSS = 61.76$.

Regression tree (prostate cancer)



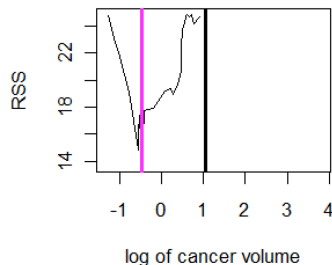
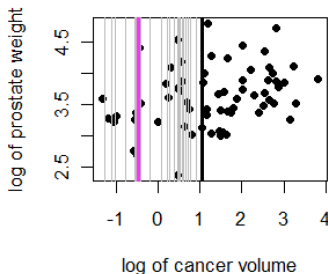
Use vertical as the first feature.

Choosing the best split in the left node



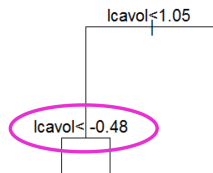
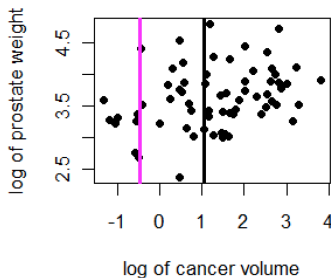
Best horizontal split is at 3.66 with $RSS = 16.11$.

Choosing the best split in the left node



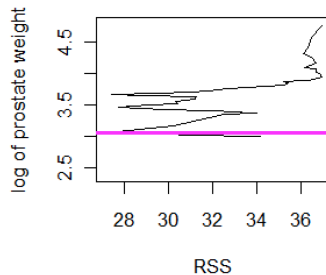
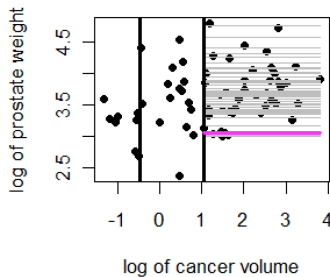
Best vertical split is at -0.48 with $RSS = 13.61$.

Regression tree (prostate cancer)



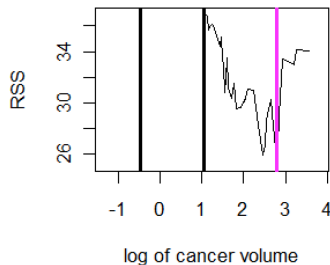
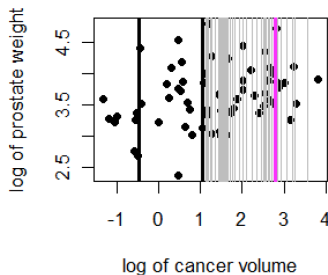
Use vertical as the feature for the left node.

Choosing the best split in the right node



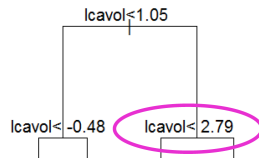
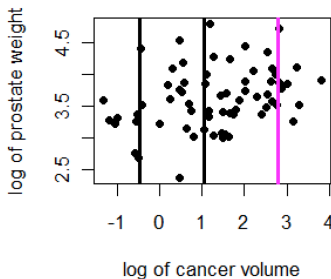
Best horizontal split is at 3.07 with $RSS = 27.15$.

Choosing the best split in the right node



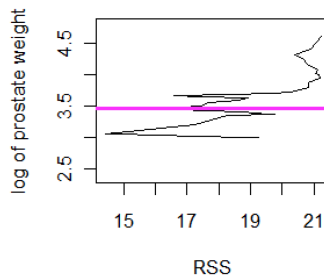
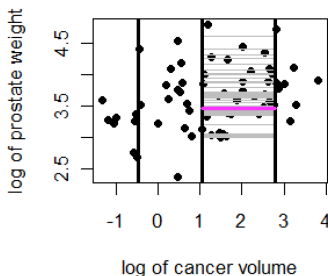
Best vertical split is at 2.79 with $RSS = 25.11$.

Regression tree (prostate cancer)



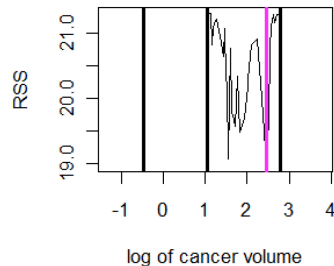
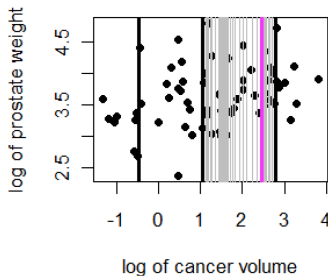
Use vertical as the feature for the right node.

Choosing the best split in the third node



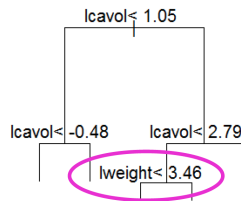
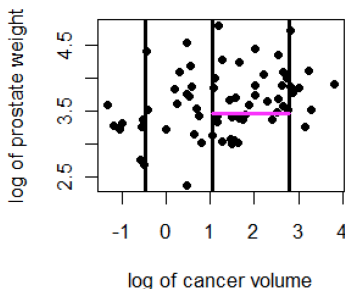
Best horizontal split is at 3.07 with $RSS = 14.42$, but this is too close to the edge. Use 3.46 with $RSS = 16.14$.

Choosing the best split in the third node



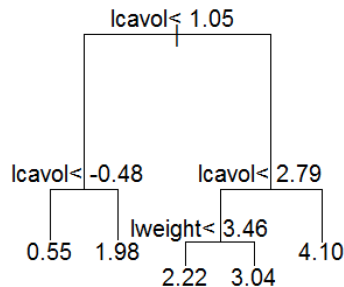
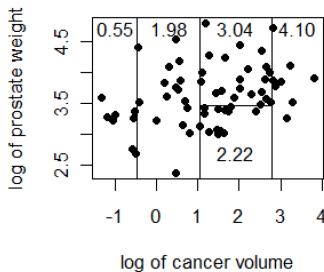
Best vertical split is at 2.46 with $RSS = 18.97$.

Regression tree (prostate cancer)

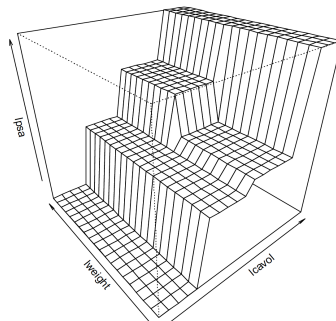
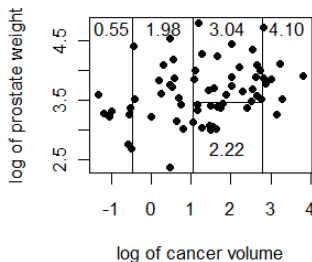


Use horizontal as the feature for the third node.

Regression tree (prostate cancer)



Regression tree (prostate cancer)



Classification and Regression Trees

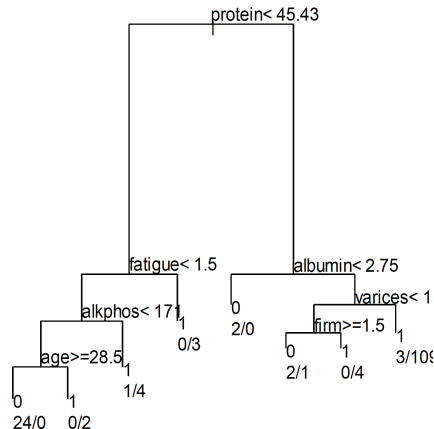
Advantages

- Applicable to both regression and classification problems.
- Handle categorical predictors naturally.
- Computationally simple and quick to fit, even for large problems.
- No formal distributional assumptions (non-parametric).
- Can handle highly non-linear interactions and classification boundaries.
- Automatic variable selection.
- Handle missing values through surrogate variables.
- Very easy to interpret if the tree is small.

Classification and Regression Trees

Advantages (cont.)

- The picture of the tree can give valuable insights into which variables are important and where.
- The terminal nodes suggest a natural clustering of data into homogeneous groups.



Classification and Regression Trees

Disadvantages

- **Accuracy** - current methods, such as support vector machines and ensemble classifiers often have 30% lower error rates than CART.
- **Instability** - if we change the data a little, the tree picture can change a lot. So the interpretation is not as straightforward as it appears.

Classification and Regression Trees

Disadvantages

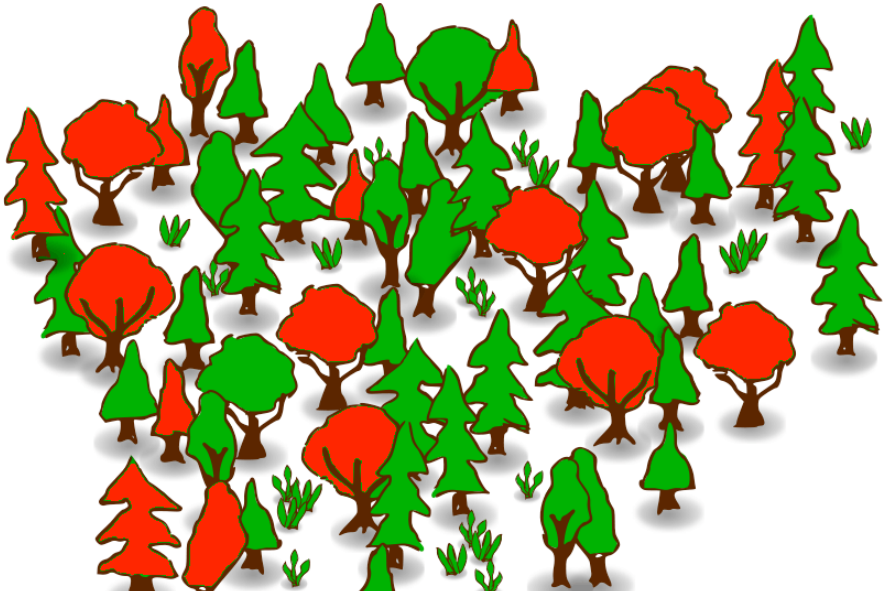
- **Accuracy** - current methods, such as support vector machines and ensemble classifiers often have 30% lower error rates than CART.
- **Instability** - if we change the data a little, the tree picture can change a lot. So the interpretation is not as straightforward as it appears.

Today, we can do better!

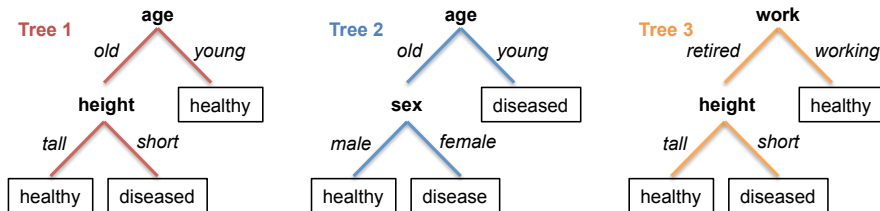
Random Forests

Random Forest

Random Forest

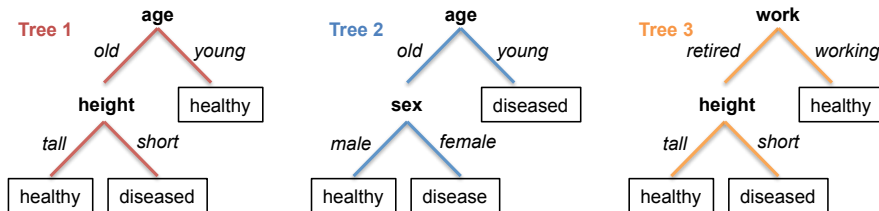


Example of Random Forest



- New sample: {old, retired, male, short}. Prediction?

Example of Random Forest



- New sample: {old, retired, male, short}. Prediction?
- Tree predictions: diseased, healthy, diseased
- Majority Rule: Diseased

The Random Forest Algorithm

- ① For $b = 1$ to B
 - ① Draw a **bootstrap sample** \mathbf{Z} of size N from the training data
 - ② Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{\min} is reached.
 - ① Select **m variables at random** from the p variables.
 - ② Pick the best variable/split-point among the m
 - ③ Split the node into two daughter nodes
- ② Output the ensemble of trees $\{T_b\}_1^B$

To make a prediction at a new point x :

- *Regression*: $\hat{f}_{\text{rf}}^B = \frac{1}{B} \sum_{b=1}^B T_b(x)$
- *Classification*: Let $\hat{C}_b(x)$ be the class prediction of the b -th random-forest tree. Then $\hat{C}_{\text{rf}}^B(x) = \text{majority vote}\{\hat{C}_b(x)\}_1^B$.

Differences to standard tree

- Train each tree on bootstrap resample of data
Bootstrap resample of data set with N samples: Make new data set by drawing with replacement N samples; i.e., some samples will probably occur multiple times in new data set
- For each split, consider only m randomly selected variables
Feature bagging
- Dont prune (why?)
- Fit a forest of *B trees* in such a way and use average or majority voting to aggregate results

Why Random Forest Works

- Mean Squared Error = Variance + Bias² + Noise
- If trees are sufficiently deep, they have very small bias (this is why we don't prune the trees)
- How could we improve the variance over that of a single tree?

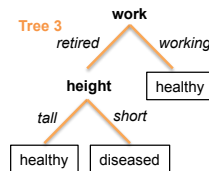
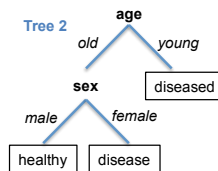
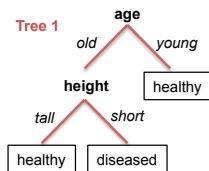
Why Random Forest Works (cont.)

$$\begin{aligned}
 \text{Var} \left(\frac{1}{B} \sum_{i=1}^B T_i(c) \right) &= \frac{1}{B^2} \sum_{i=1}^B \sum_{j=1}^B \text{Cov}(T_i(x), T_j(x)) \\
 &= \frac{1}{B^2} \sum_{i=1}^B \left(\sum_{j \neq i}^B \text{Cov}(T_i(x), T_j(x)) + \text{Var}(T_i(x)) \right) \\
 &= \frac{1}{B^2} \sum_{i=1}^B ((B-1)\sigma^2\rho + \sigma^2) = \frac{B(B-1)\rho\sigma^2 + B\sigma^2}{B^2} \\
 &= \frac{(B-1)\rho\sigma^2}{B} + \frac{\sigma^2}{B} = \rho\sigma^2 - \frac{\rho\sigma^2}{B} + \frac{\sigma^2}{B} \\
 &= \rho\sigma^2 + \sigma^2(1-\rho)/B
 \end{aligned}$$

Variance decreases if 1) **correlation ρ decreases** (i.e., if m decreases); 2) **number of trees B increases** (irrespective of ρ)

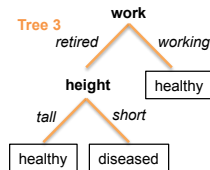
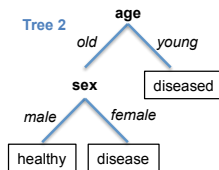
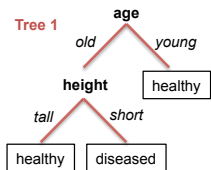
Variable Importance

- Advantage of tree methods is the interpretability.
- But we lose this property in random forest: what are the important variables in the forest?



Variable Importance

- Advantage of tree methods is the interpretability.
- But we lose this property in random forest: what are the important variables in the forest?

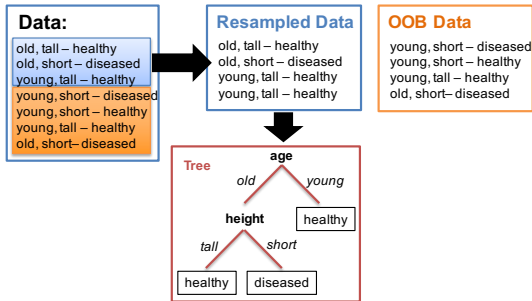


Good news. We can still identify important variables in the model.

Estimating Generalization Error

Out-of-Bag (OOB) Error

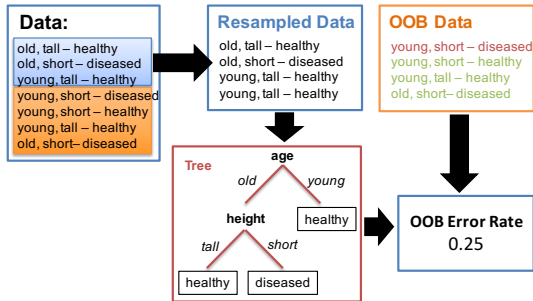
- Similar to leave-one-out cross-validation, but almost without any additional computational burden
- OOB error is a random number, since based on random resamples of the data



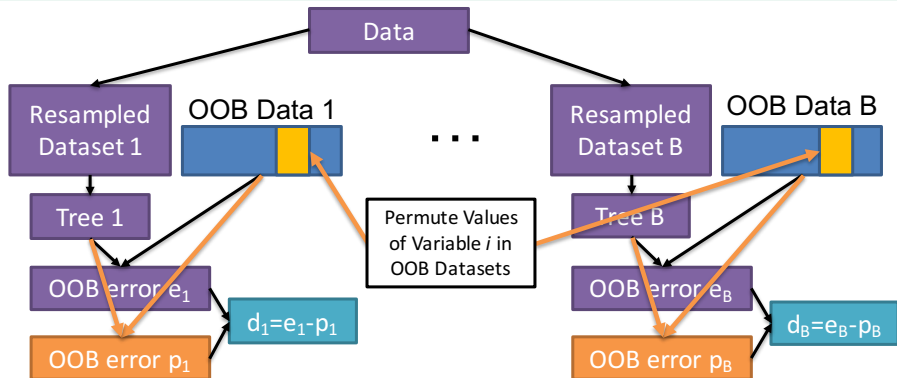
Estimating Generalization Error

Out-of-Bag (OOB) Error

- Similar to leave-one-out cross-validation, but almost without any additional computational burden
- OOB error is a random number, since based on random resamples of the data



Variable Importance for variable i using Permutations



- Average of d : $\bar{d} = \frac{1}{B} \sum_{j=1}^B d_j$
- Variance of d : $s_d^2 = \frac{1}{B-1} \sum_{j=1}^B (d_j - \bar{d})^2$
- Importance of feature i : $v_i = \bar{d}/s_d$

Tree and Random Forest

Trees

- + Trees yield insight into decision rules
- + Rather fast
- + Easy to tune parameters
- Prediction of trees tend to have a high variance

Random Forest

- + RF as smaller prediction variance and therefore usually a better general performance
- + Easy to tune parameters
- Rather slow
- “Black Box”: Rather hard to get insights into decision rules

Random Forest and LDA

Random Forest

- + Can model nonlinear class boundaries
- + OOB error “for free” (no CV needed)
- + Works on continuous and categorical responses (regression / classification)
- + Gives variable importance
- + Very good performance
- Black box
- Slow

LDA

- + Very fast
- + Discriminants for visualizing group separation
- + Can read off decision rule
- Can model only linear class boundaries
- Mediocre performance
- No variable selection
- Only on categorical response
- Needs CV for estimating prediction error

Proximity Measures

Proximities

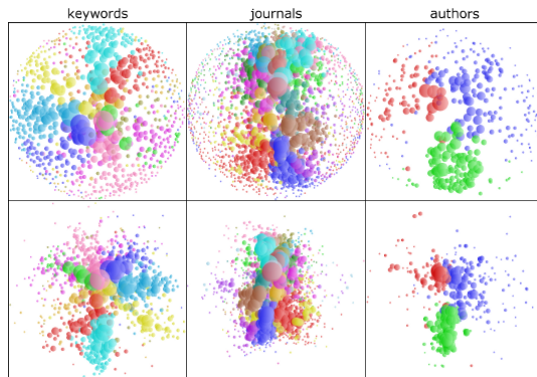
- Proximity in a tree : the proportion of the time that they end up in the same node.
- The proximities don't just measure similarity of the variables - they also take into account the importance of the variables.
 - Two cases that have quite **different** predictor variables might have **large** proximity if they differ only on variables that are **not important**.
 - Two cases that have quite **similar** values of the predictor variables might have **small** proximity if they differ on inputs that are **important**.
- Proximity in random forest

Proximities

- Proximity in a tree : the proportion of the time that they end up in the same node.
- The proximities don't just measure similarity of the variables - they also take into account the importance of the variables.
 - Two cases that have quite **different** predictor variables might have **large** proximity if they differ only on variables that are **not important**.
 - Two cases that have quite **similar** values of the predictor variables might have **small** proximity if they differ on inputs that are **important**.
- Proximity in random forest : After each tree is built, all of the data are run down the tree, and proximities are computed for each pair of cases. If two cases occupy the same terminal node, their proximity is increased by one. At the end of the run, the proximities are normalized by dividing by the number of trees.

Visualizing using Proximities

- To “look” at the data we use multidimensional scaling (MDS) to get a picture in 2-D or 3-D:



$$\min \text{Stress}_D(x_1, x_2, \dots, x_N) = \left(\sum_{i \neq j=1, \dots, N} (d_{ij} - \|x_i - x_j\|)^2 \right)^{1/2}$$

Visualizing using Proximities

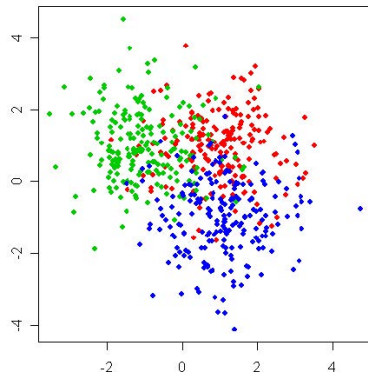
- at-a-glance information about which classes are overlapping, which classes differ
- find clusters within classes
- find easy/hard/unusual cases (data points)

With a good tool we can also

- identify characteristics of unusual points
- see which variables are locally important
- see how clusters or unusual points differ

Visualizing using Proximities

- Synthetic data, 600 cases
- 2 meaningful variables
- 48 “noise” variables
- 3 classes



Other Uses for Random Forests

- Missing data imputation.
- Feature selection.
In the NIPS competition 2003, several of the top entries used RF for feature selection.
- Unsupervised learning (cluster analysis).

Missing Data Imputation

Fast way: replace missing values for a given variable using the median of the non-missing values (or the most frequent, if categorical)

Better way (using proximities):

- ① Start with the fast way.
- ② Get proximities.
- ③ Replace missing values in case i by a weighted average of non-missing values, with weights proportional to the proximity between case i and the cases with the non-missing values.

Repeat steps 2 and 3 a few times (5 or 6).

Unsupervised Learning

Generating an ensemble of individual tree predictors, leads to a measure of natural dissimilarity between the observations

- Global histone modification patterns predict risk of prostate cancer recurrence
- Used RF clustering of 183 tissue microarrays to find two disease subgroups with distinct risks of tumor recurrence.
- <http://www.nature.com/nature/journal/v435/n7046/full/nature03672.html>