

Ensemble Methods

Lecture for CHM696D

12 March 2003

Rick Higgs & Dave Cummins
Statistical & Information Sciences
Lilly Research Laboratories

Where We Left Off Last Time...

Recursive Partitioning:

Is unable to extrapolate beyond the range of observed responses.

Is less efficient than parametric models at revealing smooth trends.

Gives a non-parsimonious description of additive models.

Tends to require more observations than linear models.

- While reasonable RP models have been built on as few as 50 observations, an "interesting" model usually requires hundreds.

Interpretation can be misleading.

- A tree's simplistic form may fool a user into missing highly correlated but more relevant covariates.
- Different trees can often describe the same data.

Higher order interactions can be masked by nonsignificant main effects where two or more covariates perfectly counterbalance each other and so an initial split is not made.

Discrete, non-smooth nature of model (bigger problem for regression)

Unstable → high variance (slight change in data produces very different tree)

Ensemble Methods

- Basic idea:
 - Build many models (e.g. trees)
 - When making prediction, use an average from each model.
- These methods have had the most profound impact on prediction accuracy of any method in the last decade
- Theory is not all worked out but it's coming
 - Volumes of empirical data for these methods

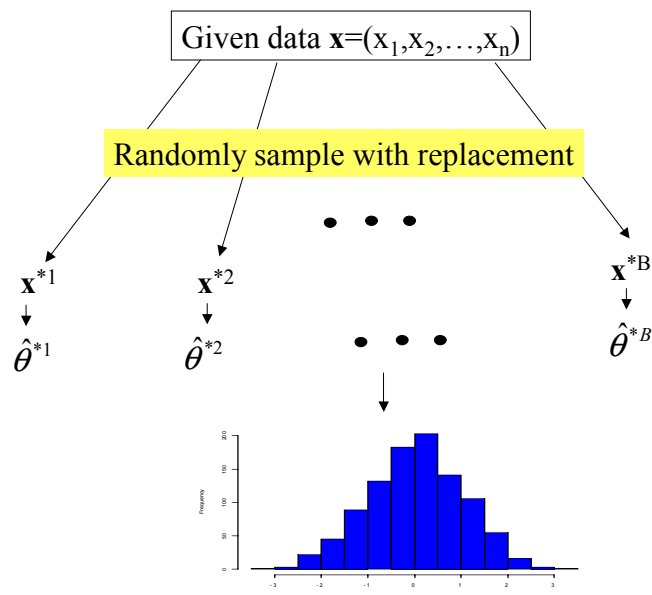
Two+ Ways to Form Ensembles

- Bagging (and it's variants)
 - Randomness introduced by sampling data
 - Works best with large trees
 - Framework for understanding: correlation and strength of ensemble members
 - Individual ensemble members have low bias, high variance
 - Averaging reduces the variance
- Boosting (and it's variants)
 - Small trees (even stumps) seem to work best
 - Successive trees constructed via iterative re-weighting
 - Essentially an additive model of elementary basis functs.

But First.... The Bootstrap

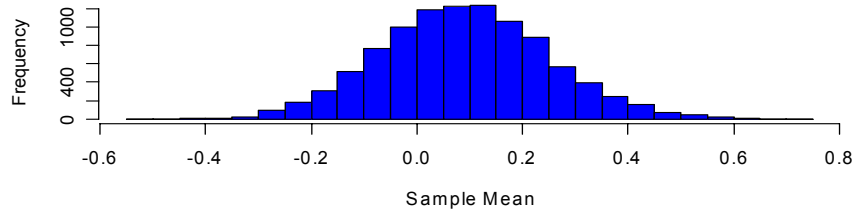
- Computational technique to obtain the distribution of a parameter from a sample of data
- Resample the sample data (with replacement) to create numerous bootstrap samples
 - Compute estimate of parameter of interest on each bootstrap sample
 - Estimate distribution of parameter by empirical distribution of parameter computed on bootstrap samples
- Works via the plug-in principle
 - If $\theta = t(F)$
 - Then plug-in estimate is $\hat{\theta} = t(\hat{F})$
 - Where \hat{F} is the empirical distribution derived from data

The Bootstrap

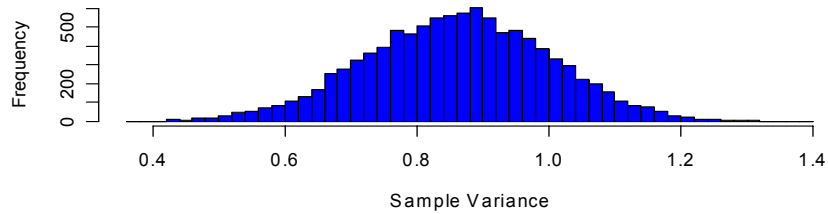


Simple Bootstrap Example

10000 Bootstrap Samples 30 N(0,1) iid



10000 Bootstrap Samples



Combining Models (Regression)

Given data: $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$

Set of models: $\hat{f}_1(\mathbf{x}), \hat{f}_2(\mathbf{x}), \dots, \hat{f}_k(\mathbf{x})$

Simple average: $\hat{f}(\mathbf{x}) = \frac{1}{k} \sum_{i=1}^k \hat{f}_i(\mathbf{x})$

Bias: $Bias(\mathbf{x}) = E_T[\hat{f}(\mathbf{x})] - E[y | \mathbf{x}]$

$$Bias(\mathbf{x}) = \frac{1}{k} \sum_{i=1}^k E_T[\hat{f}_i(\mathbf{x})] - E[y | \mathbf{x}]$$

If individual models are unbiased then so is their average

Combining Models (Regression)

$$\text{Variance: } \text{Var}(\mathbf{x}) = E_T[(\hat{f}(\mathbf{x}) - E_T[\hat{f}(\mathbf{x})])^2]$$

$$\text{Var}(\mathbf{x}) = E_T\left[\left(\frac{1}{k} \sum_{i=1}^k \hat{f}_i(\mathbf{x}) - \frac{1}{k} \sum_{i=1}^k E_T[\hat{f}_i(\mathbf{x})]\right)^2\right]$$

$$\text{Var}(\mathbf{x}) = \frac{1}{k^2} \sum_{i=1}^k \text{Var}[\hat{f}_i(\mathbf{x})] + \frac{2}{k} \sum_{i < j} \text{Cov}[\hat{f}_i(\mathbf{x}), \hat{f}_j(\mathbf{x})]$$

If individual models have: $\text{Var}[\hat{f}_i(\mathbf{x})] \approx \sigma_f^2$

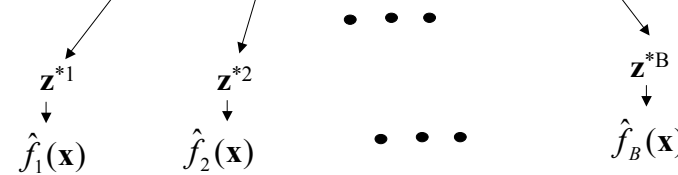
$$\text{Then: } \text{Var}(\mathbf{x}) \approx \frac{\sigma_f^2}{k} + \frac{2}{k} \sum_{i < j} \text{Cov}[\hat{f}_i(\mathbf{x}), \hat{f}_j(\mathbf{x})]$$

Want covariances small \rightarrow 'diverse' models

Bagging (Bootstrap Aggregating)

Given data $\mathbf{z} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$

Randomly sample with replacement



Average or Majority Vote

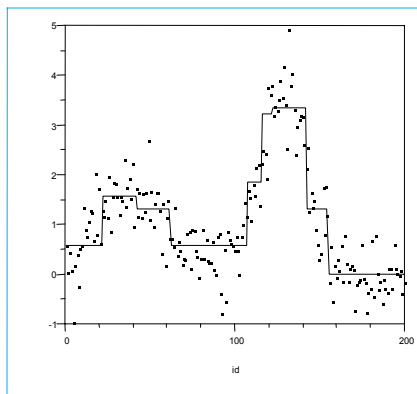
$\hat{f}(\mathbf{x})$

Bagging

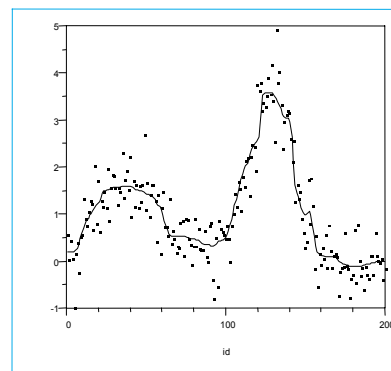
- Key concept
 - Models from an unstable method (like trees) constructed on different bootstrap samples can exhibit significant differences.
 - These differences, along with the individual ‘strength’ of the ensemble members are what reduces the variance in the predictions.
 - Therefore, bagging is not particularly useful for stable methods like Linear Discriminant Analysis

Simple Bagging Example

One CART Tree



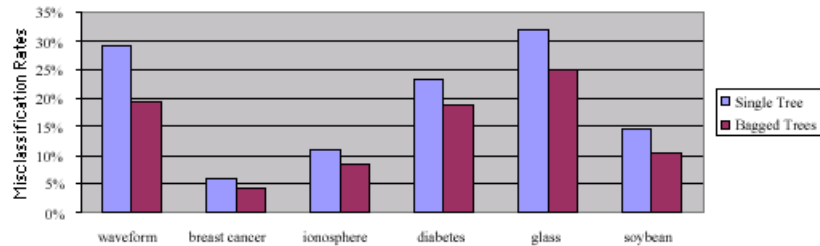
Two Hundred Bagged CART Trees



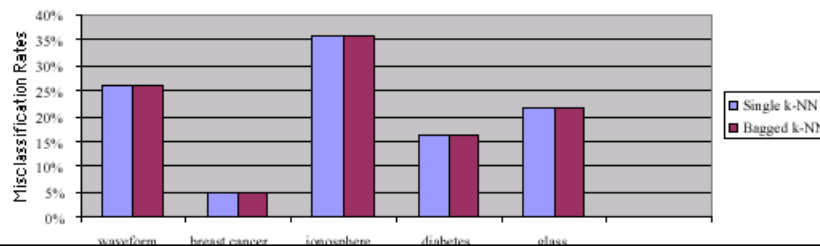
"Bagging smooths the decision boundaries"

Early Results from Breiman (1996)

Single and Bagged Decision Trees (50 Bootstrap Replicates)
Test Set Average Misclassification Rates over 100 Runs



Single and Bagged k-NN (100 Bootstrap Replicates)
Test Set Average Misclassification Rates over 100 Runs



Boosting

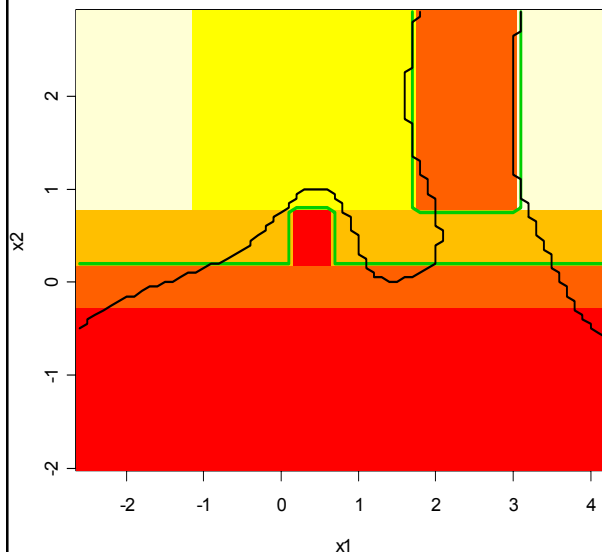
- Boosting is an ensemble method that generally outperforms bagging
- Assumes the modeling method can incorporate weights on the observations
 - e.g. Trees: bootstrap sample with probabilities = weights
- Typically used with ‘weak’ classifiers
 - e.g. only slightly better than random guessing
- Sequentially apply weak classifier to modified versions of the data
 - Produces a sequence of weak classifiers $G_m(\mathbf{x})$, $m=1,2,\dots,M$
- AdaBoost (Adaptive Boosting) most prominent algorithm

AdaBoost Algorithm

1. Initialize the observation weights $\omega_i = \frac{1}{N}, i = 1, 2, \dots, N$
2. For $m=1$ to M {
 - (a) Fit a classifier $G_m(\mathbf{x})$ to the training data using weights
 - (b) Compute $err_m = \frac{\sum_{i=1}^N \omega_i I(y_i \neq G_m(\mathbf{x}_i))}{\sum_{i=1}^N \omega_i}$
 - (c) Compute $\alpha_m = \log((1 - err_m) / err_m)$
 - (d) Adjust weights $\omega_i = \omega_i e^{\alpha_m I(y_i \neq G_m(\mathbf{x}_i))}, i = 1, 2, \dots, N$}
3. Output $G(\mathbf{x}) = \text{sign}[\sum_{m=1}^M \alpha_m G_m(\mathbf{x})]$

Back to Our Example...

Single rpart tree constructed with default options
Training set size increased from previous examples (n=2000 now)

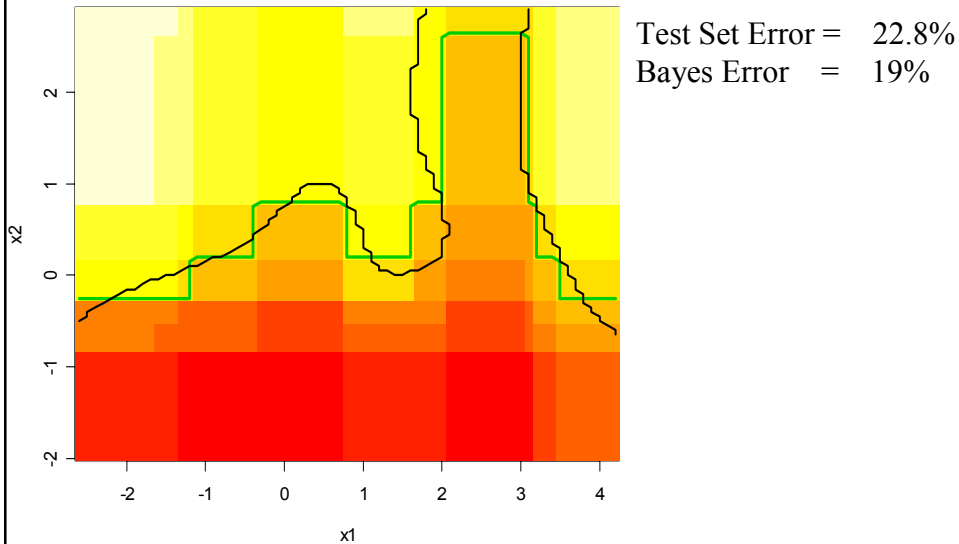


Test Set Error = 24.5%
Bayes Error = 19%

Back to Our Example

200 AdaBoost Trees (rpart using maxdepth=2)

Training set size increased from previous examples (n=2000 now)



Random Forests

- “A better bagging”
- Decrease the correlation between ensemble members by forcing it in the tree construction

$$Var(\mathbf{x}) \approx \frac{\sigma_f^2}{k} + \frac{2}{k} \sum_{i < j} Cov[\hat{f}_i(\mathbf{x}), \hat{f}_j(\mathbf{x})]$$

- Prediction as good (sometimes better) than AdaBoost
- Relatively robust to outliers and noise
- Faster than bagging or boosting

Random Forests Algorithm

- Each tree grown on a bootstrap sample of data
 - Like bagging
- m ($m \ll M$, where M =number of features) is specified
 - Default usually $m = \sqrt{M}$
- Tree growing:
 - At each node randomly select m features
 - Split based on the best of the m randomly chosen features

Random Forests

- Unlike boosting, RF converges
 - Recently, stopping rules have been developed for boosting
- Like bagging, RF is tolerant to label noise
 - Boosting is not, although variants emerging that are
- The key to accuracy is low bias and low correlation
 - Grow big trees \rightarrow low bias
 - Get low correlation via randomization
- Only one adjustable parameter (m)
 - Appears to be relatively insensitive to m

Random Forests & the “Occam Dilemma”

- RF makes A+ predictors
- For interpretability they appear to get a grade of F
 - On the surface that is
- “Using complex predictors may be unpleasant, but the soundest path is to go for predictive accuracy first, then try to understand why.”
 - Leo Breiman

Random Forests – Additional Benefits

- Variable importance measures
- Effects of variables on predictions
- Proximities between observations
 - “relative” to features important for classification
 - Clustering
 - Outlier detection

Random Forests – Test Set Error Estimation

- In a bootstrap sample about 1/3 of the observations will not be used for training
 - Denote those observations not used for training as OOB (out of bag) observations
- For each tree, record the OOB predictions
- Average OOB predictions for each observation across all trees
- Built in estimate of test set error

Random Forests – Variable Importance

- Randomly permute feature m
- Re-estimate the test set error by running the OOB samples with the permuted feature back down all trees
- The amount that this new test set error exceeds the original test set error is defined as the importance of the m^{th} feature.
- Can be extended to 2 features at a time for interactions

Random Forests – Proximity Measure

- Initialize a similarity (or proximity) matrix
 - $S = s_{ij} = 0, i=1, \dots, n, j=1, \dots, n$
- Run all training cases down each tree
- If case i and case j both land in same leaf node,
 - $s_{ij} = s_{ij} + 1$
- Scale by the number of trees $s_{ij} = s_{ij} / k$
- Define self-similarity
 - $s_{ij} = 1$ if $i=j$
- Cluster, multidimensional scaling, using S

Random Forests – Outlier Detection

- Outlier determination for large data sets problematic
 - Which features should one use for defining a distance?
 - How should those features be weighted if they were given?
- Can be a critical component to real-world problems
 - Should I even make a prediction on this molecule
 - Or, should I refuse to predict because I'm well outside of what the model was trained to do
 - Frequently our observational data sets violate sampling assumptions that all of the methods we discuss assume

Random Forests – Outlier Detection

- For a newly observed case (\mathbf{x})
 - Run the case down each tree in the forest
 - Compute a similarity between \mathbf{x} and all training set samples identical to that described previously

Splice Site Intron/Exon Example

Goal is to predict intron/exon interfaces in nucleotide sequences



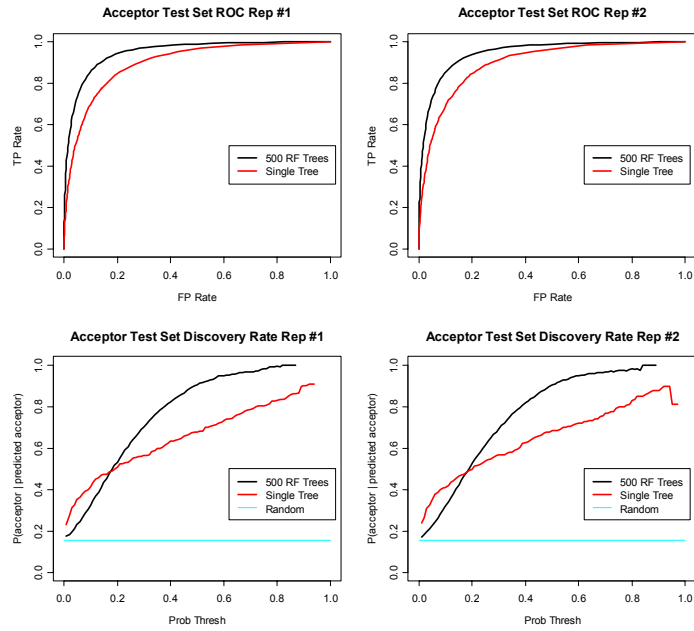
+/- 70 nt window around GT



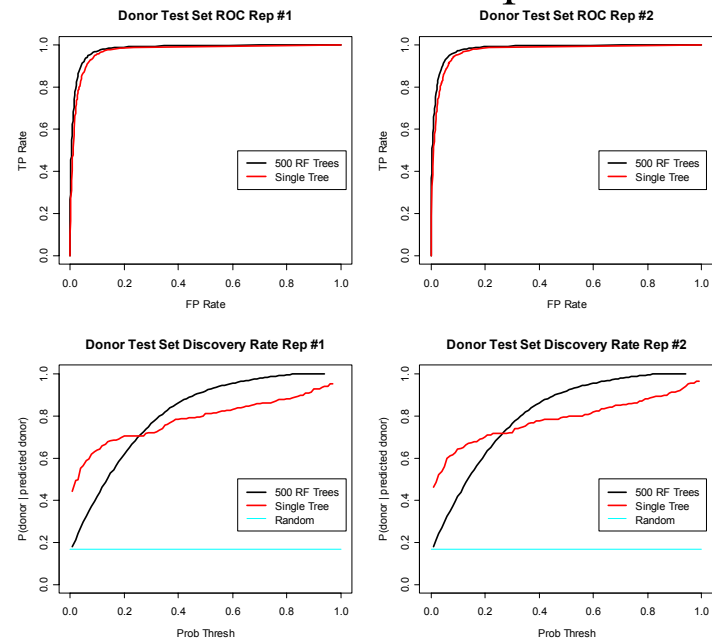
140 categorical features (ATCG) and binary response
(donor site/ not donor site)

(Acceptor site constructed in same manner around AG)

Intron/Exon Example



Intron/Exon Example



Selected References

1. Breiman, L. (1996). Bagging predictors, *Machine Learning*, **26**: 123-140. Breiman, L. (1996). Bias, variance, and arcing classifiers, UC Berkeley Statistics
2. Breiman, L. (2000). Some infinity theory for predictor ensembles, UC Berkeley Statistics Department Technical Report 577.
3. Breiman, L. (2001). Statistical modeling: the two cultures, *Statistical Science*, **16**(3):199-231.
4. Breiman, L. (2001). Random forests, *Machine Learning*, **45**:5-32.
5. Department Technical Report 460.
6. Efron, ., Tibshirani, R.J. (1993). *An Introduction to the Bootstrap*, Chapman and Hall, London.
7. Schapire, R. (1990). The strength of weak learnability, *Machine Learning*, **5**(2):197-227.

<http://stat-www.berkeley.edu/users/breiman/>

<http://www-stat.stanford.edu/~jhf/>