What Constitutes an Ensemble Method

Ensemble: a technique for combining many weak learners in an attempt to produce a strong learner

Basic Idea: run multiple models on the data and aggregate the predictive results to produce an overall prediction that is better than any of the individual models could do on their own.

Ensembles Continued

Intuition

Classifier Example with Majority Vote:

- Suppose we have 5 completely independent classifiers with an accuracy of 70% for each
  - $\binom{5}{3}(0.7)^3(0.3)^2 + \binom{5}{4}(0.7)^4(0.3)^1 + \binom{5}{5}(0.7)^5(0.3)^0$
  - yields 83.7% majority vote accuracy
- If we had 101 such classifiers
  - 99.9% majority vote accuracy

It's clear that utilizing multiple weak classifiers can yield a very strong classifier

Bootstrap Sampling

Advantages:

- Completely automatic
- Requires no theoretical calculations
- Not based on asymptotic results
- Available regardless of how complicated the estimator might be

Bootstrap Sampling

Method:

- Start with your dataset of size n
- Sample from your dataset with replacement to create 1 bootstrap sample of size n which means many of the observations will be repeated
- Repeat B times
- Each bootstrap sample can then be used as a separate dataset for estimation or model fitting

Bootstrap Aggregation (Bagging)

Use bootstrap to improve predictions by using different samples of observations and/or features to generate diverse classifiers

- Improves stability and accuracy of ML algorithms
- Also reduces variance and helps to avoid overfitting
- Although usually applied to decision tree methods, it can be used with any type of method

Bootstrap Aggregation (Bagging)

Method:

- Create bootstrap samples, fit a model from each bootstrap sample; can use a subset of possible features to build models
- Each model then gets a vote in the final results
- Final selection based on aggregate predictions
  - majority vote if classification (i.e. predicting a category)
  - average if regression (i.e. predicting a continuous value)

Random Forests vs Bagging Alone

Random forests provide an improvement over bagged trees by way of a small tweak that decorrelates the trees

- Both methods utilize bootstrap samples
- RF also randomly selects features

Intuition behind this:

- In bagged trees, a very strong predictor will show up in the top split so all trees will look quite similar and highly correlated
- In RF, the very strong predictor will be included only a portion of the time, allowing moderately strong predictors to have a chance

Random Forest Algorithm

For $b = 1, \ldots, B$:

- Draw a bootstrap sample of size N from the training data
- Grow a random-forest tree to the bootstrapped data by recursively repeating the following for each terminal node of the tree, until minimum node size is reached
  - Picking a subset of features (m) from the p features
  - Pick the best variable/split-point among the m variables
  - Split the node into two child nodes
- Output the ensemble of trees

Tuning

The following are suggested values of m:

- Classification:

$$m = \sqrt{p}$$

- Regression:

$$m = \frac{p}{3}$$

In reality we can tune this parameter

Out-of-Bag Error

The key to bagging is that models are repeatedly fit to bootstrapped subsets of observations. Therefore:

- Each bagged model makes use of about 2/3 of the observations
- Remaining 1/3 of the observations are referred to as the *out-of-bag* (OOB) observations
- We can predict the i^{th} response using the bagged model in which the observation was OOB
- This will yield approximately B/3 predictions for the i^{th} observation and aggregate the predictions as above
- We now have predictions for all observations and can then calculate the OOB Error

Out-of-Bag Error

- OOB error is a valid estimate of the test error since the observation is predicted using models that did not use that observation
- When B is sufficiently large, OOB error is virtually equivalent to leave-one-out cross-validation error
- OOB approach is convenient when bagging on large data sets for which cross-validation would be computationally intensive

Forest Error

Overall forest error rate depends on two things:

- The *correlation* between any two trees in the forest
- The *strength* of each individual tree in the forest

Reducing m reduces both the correlation and strength, while increasing it increases both.

Another trade-off, use OOB error to find the optimal value of m.

Feature Importance

Although the aggregation of bagged trees improves prediction accuracy, it is much more difficult to interpret than a single tree

- We can, however, obtain an overall summary of the importance of each predictor
  - Classification Trees: record total amount Gini index decreases due to splits over given predictor, averaged over all B trees $\longrightarrow$ Larger value indicates 'higher importance'
  - Regression Trees: record total amount RSS decreases due to splits over a given predictor, averaged over all B trees $\longrightarrow$ Larger value indicates 'higher importance'

Feature Importance - Alternative Method #1

Random permutation method ala Breiman:

- To evaluate the importance of the j^{th} variable
  - When the b^{th} tree is grown, the OOB samples are passed through the tree $\longrightarrow$ compute accuracy
  - Values of the j^{th} variable in the OOB samples are randomly permuted $\longrightarrow$ compute new (lower) accuracy
  - Average the decrease in accuracy over all the trees
- The larger the average decrease in accuracy, the more important the feature

Feature Importance - Alternative Method #2

The following is the method used in sklearn:

- The higher in the tree the feature is, the more important it is in determining the final prediction of a data point
  - The fraction of data points that reach a node is used as an estimate of that feature's importance for that particular tree
  - Average those values across all trees to get the feature's importance
  - The higher the average fraction of observations that were sent through the feature, the more important

Some Advantages of Bagging/Random Forests

- All trees are trained independently
  - possibly in parallel reducing computation time
- Can handle thousands of input variables without variable deletion
- Gives estimates of which features are important in the classification

If the number of variables is very large, forests can be run will all variables, then run again using only the most important from the first run

Bias-Variance Tradeoff

- Bias
  - the bias of the aggregated model is equivalent to the bias of a single decision tree which should have low bias
- Variance
  - By creating a forest and averaging them the variance of the final model can be greatly reduced over that of a single tree

Only limitation of the size of the forest is computing time as an infinite number of trees could be trained without ever increasing bias and with a continually decrease in the variance (although asymptotically limited)

Pruning/Overfitting

- We've shown that a single decision tree which can suffer from overfitting and needs to be pruned
- In Bagging/RF, each tree is grown to the largest extent possible (a high variance predictor)
- Pruning is not needed since high variance is not an issue as long as the number of trees (B) is large