Lecture 15: Bagging, Random Forests

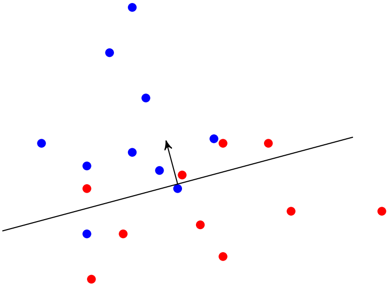
Reading: Sections 9.2, 15.2, 15.3

GU4241/GR5241 Statistical Machine Learning

Linxi Liu
April 5, 2019

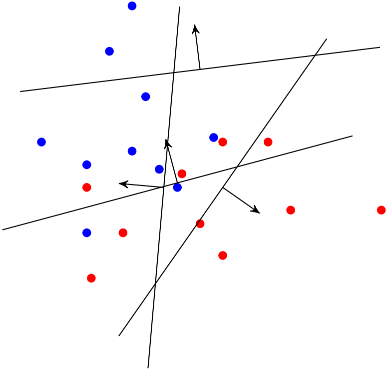
Ensembles

A *randomly* chosen hyperplane classifier has an *expected* error of 0.5 (i.e. 50%).



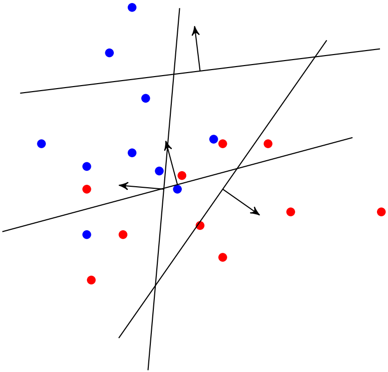
Ensembles

A *randomly* chosen hyperplane classifier has an *expected* error of 0.5 (i.e. 50%).



Ensembles

A *randomly* chosen hyperplane classifier has an *expected* error of 0.5 (i.e. 50%).



- Many random hyperplanes combined by majority vote: Still 0.5.
- A single classifier slightly better than random: $0.5 + \varepsilon$.
- What if we use m such classifiers and take a majority vote?

Voting

Decision by majority vote

- m individuals (or classifiers) take a vote. m is an odd number.
- They decide between two choices; one is correct, one is wrong.
- After everyone has voted, a decision is made by simple majority.

Note: For two-class classifiers $f_1, \ldots, f_m$ (with output ± 1):

$$\text{majority vote } = \operatorname{sgn}\Big(\sum_{j=1}^{m} f_j\Big)$$

Assumptions

Before we discuss ensembles, we try to convince ourselves that voting can be beneficial. We make some simplifying assumptions:

- Each individual makes the right choice with probability $p \in [0,1]$.
- The votes are *independent*, i.e. stochastically independent when regarded as random outcomes.

Does the Majority Make the Right Choice?

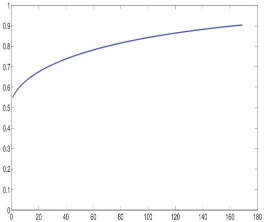
Condorcet's rule

If the individual votes are independent, the answer is

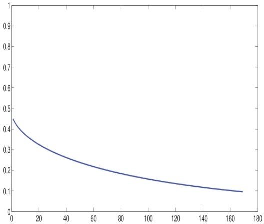
$$\Pr\{\text{ majority makes correct decision }\} = \sum_{j=\frac{m+1}{2}}^{m} \frac{m!}{j!(m-j)!} p^j (1-p)^{m-j}$$

This formula is known as **Condorcet's jury theorem**.

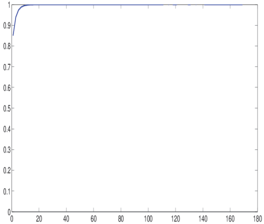
Probability as function of the number of votes



$p = 0.55$



$p = 0.45$



$p = 0.85$

Ensemble Methods

Terminology

- An **ensemble method** makes a prediction by combining the predictions of many classifiers into a single vote.
- The individual classifiers are usually required to perform only slightly better than random. For two classes, this means slightly more than 50% of the data are classified correctly. Such a classifier is called a **weak learner**.

Strategy

- We have seen above that if the weak learners are random and independent, the prediction accuracy of the majority vote will increase with the number of weak learners.
- Since the weak learners all have to be trained on the training data, producing random, independent weak learners is difficult.
- Different ensemble methods (e.g. Boosting, Bagging, etc) use different strategies to train and combine weak learners that behave relatively independently.

Methods We Will Discuss

Boosting

- After training each weak learner, data is modified using weights.
- Deterministic algorithm.

Bagging

Each weak learner is trained on a random subset of the data.

Random forests

- Bagging with decision trees as weak learners.
- Uses an additional step to remove dimensions in $\mathbb{R}^d$ that carry little information.

Bagging = Bootstrap Aggregation

- Replicate the dataset by sampling with replacement.
- We apply a learning method to each bootstrap replicate, to produce predictions $\hat{f}^{(1)}, \ldots, \hat{f}^{(B)}$.
- In Chapter 5, we were interested in the variability of these predictions:

$$\mathsf{SE}(\hat{f}(x)) \approx \mathsf{SD}(\hat{f}^{(1)}(x), \ldots, \hat{f}^{(B)}(x)).$$

- Now, we will use the average of these predictions as an estimator with reduced variance:

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

Bagging decision trees

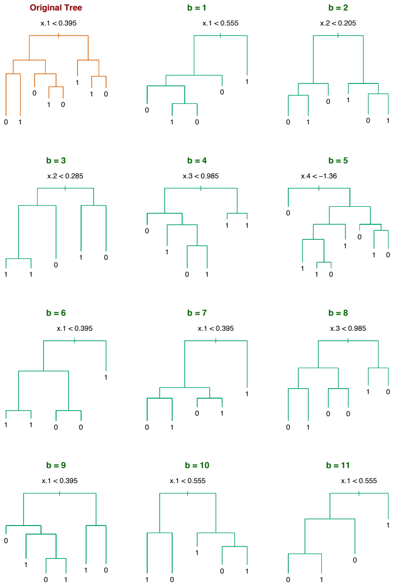
- Replicate the dataset by sampling with replacement.
- Fit a decision tree to each bootstrap replicate (growing the tree, and pruning).
- **Regression:** To make a prediction for an input point x, average the predictions of all the trees:

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

- **Classification:** To make a prediction for an input point x_0, take the majority vote from the set of predictions:

$$\hat{y}_0^{(1)}, \ldots, \hat{y}_0^{(B)}.$$

Example: Bagging decision trees

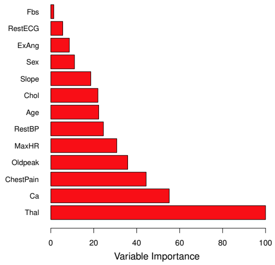


- Two classes, each with Gaussian distribution in $\mathbb{R}^5$.
- Note the variance between bootstrapped trees.

Bagging decision trees

- **Disadvantage:** Every time we fit a decision tree to a Bootstrap sample, we get a different tree T^b.

  $\rightarrow$ Loss of interpretability
- For each predictor, add up the total amount by which the RSS (or Gini index) decreases every time we use the predictor in T^b.
- Average this total over each Boostrap estimate $T^1, \ldots, T^B$.



How Often Do We See Each Sample in Bootstrap?

Sample $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, bootstrap resamples $\mathcal{B}_1, ..., \mathcal{B}_B$.

In how many sets does a given $\mathbf{x}_i$ occur?

Probability for $\mathbf{x}_i$ *not* to occur in n draws:

$$\mathsf{Pr}\{\mathbf{x}_i \notin \mathcal{B}_b\} = (1 - \frac{1}{n})^n$$

For large n:

$$\lim_{n\to\infty} \left(1 - \frac{1}{n}\right)^n = \frac{1}{e} \approx 0.3679$$

- Asymptotically, any $\mathbf{x}_i$ will appear in $\sim 63\%$ of the bootstrap resamples.
- Multiple occurrences possible.

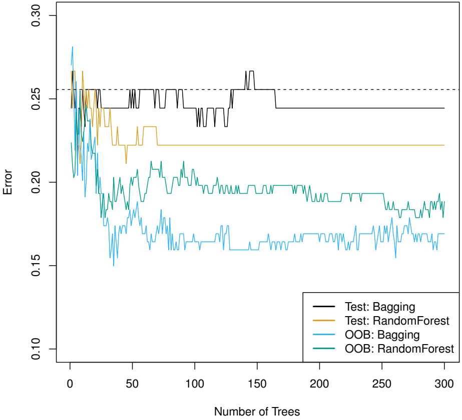
How often is $\mathbf{x}_i$ expected to occur?

The *expected* number of occurences of each $\mathbf{x}_i$ is B.

Out-of-bag (OOB) error

- To estimate the test error of a bagging estimate, we could use cross-validation.
- Each time we draw a Bootstrap sample, we only use 63% of the observations.
- **Idea:** use the rest of the observations as a test set.
- **OOB error:**
  - For each sample x_i, find the prediction $\hat{y}_i^b$ for all bootstrap samples b which do not contain x_i. There should be around $0.37B$ of them. Average these predictions to obtain $\hat{y}_i^{\text{oob}}$.
  - Compute the error $(y_i - \hat{y}_i^{\text{oob}})^2$.
  - Average the errors over all observations $i = 1, \dots, n$.
- For B large, OOB error is virtually equivalent to LOOCV.

Out-of-bag (OOB) error



The test error decreases as we increase B
(dashed line is the error for a plain decision tree).

Random Forests

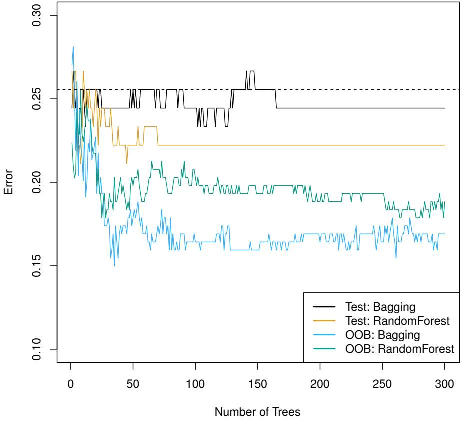
Bagging has a problem:

$\rightarrow$ The trees produced by different Bootstrap samples can be very similar.

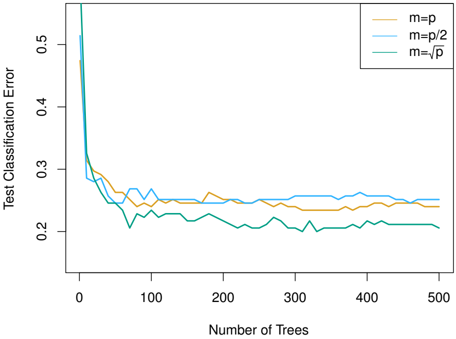
Random Forests:

- We fit a decision tree to different Bootstrap samples.
- When growing the tree, we select a random sample of $m < p$ predictors to consider in each step.
- This will lead to very different (or "uncorrelated") trees from each sample.
- Finally, average the prediction of each tree.

Random Forests vs. Bagging



Random Forests, choosing m



The optimal m is usually around $\sqrt{p}$,
but this can be used as a tuning parameter.