Bagging, Random Forests, and Boosting

Prof. Sharon McNicholas

STATS 780/CSE 780

Introduction

- In the last class, we saw classification trees.
- We observed that they had some advantages, e.g., they are nice to visualize and easy to understand/explain.
- But we also saw that they are not as good as some other classifiers.
- On a related note: for a given data set, we saw that as we change the training set and refit the tree, there is quite a bit of variation.
- Today we will look at ways to improve classification performance, i.e., to reduce variation.

Ensemble Methods

- Ensemble methods are very powerful.
- The idea is that a combination of different learning approaches can work better than any of the constituent approaches.
- As with other topics we have seen, there are entire books based on ensemble methods, e.g., Zhou (2012)[a].

[a] Zhou, Z.-H. (2012), *Ensemble Methods: Foundations and Algorithms*, Boca Raton: Chapman & Hall/CRC Press.

Bagging

- In bagging (bootstrap aggregating), we:
  - use bootstrapping to resample the training set,
  - fit a learning method to each bootstrap sample (ensemble), and
  - then average (or otherwise combine) the resulting predictions.
- Let's consider the specific example of a regression tree.

Bagging for a Regression Tree

- Generate M bootstrap ensembles from the training set.
- Use $\hat{f}^m(\mathbf{x})$ to denote the prediction (regression tree) trained on the mth bootstrap ensemble.
- Averaging these predictors gives

$$\hat{f}_{\text{bag}}(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} \hat{f}^m(\mathbf{x}).$$

- This is **bagging**.

Bagging Comments

- Seems very simple. . . but it works.
- To see why, consider the distribution of the mean.
- Note that the M trees are not pruned.
- Bagging naturally gives a very nice method for estimating the (test) error. . . .

Out-of-Bag Error Estimation

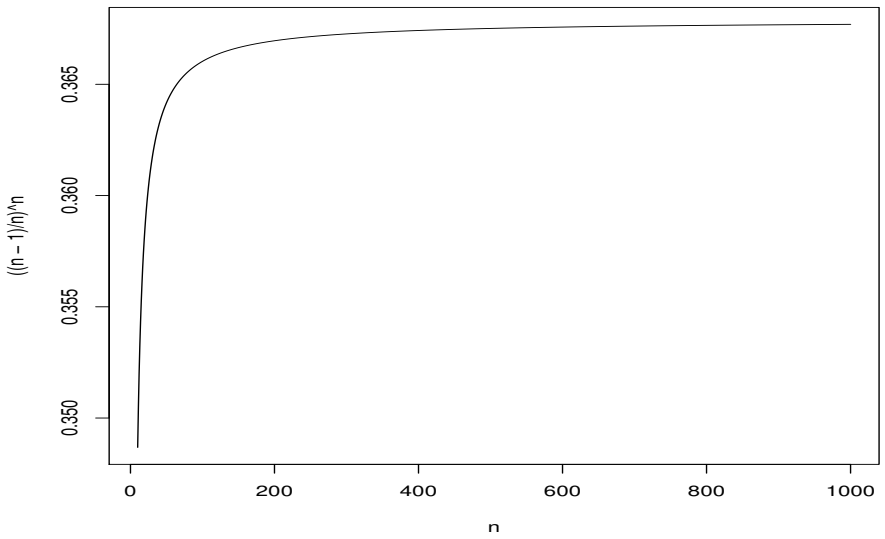
- When we create a bootstrap ensemble, some observations will be included more than once and some will be left out altogether — the left out observations are said to be **out-of-bag**.

- The response for the ith observation can be predicted using the trees for which it was out-of-bag.

- Note that there will be, on average, around $M/3$ predictions for each observation.

- These can then be averaged to give a prediction for the ith observation.

Why Approximately $M/3$?



Out-of-Bag Error Estimation contd.

- Then, the out-of-bag MSE can be computed.
- That is,

$$\text{MSE}_{\text{oob}} = \frac{1}{M}\sum_{m=1}^{M}\left(\hat{f}^m(\mathbf{x}) - \hat{f}_{\text{bag}}(\mathbf{x})\right)^2.$$

- This can be used as an estimate of the test error and, as such, is an alternative to the (cross) validation approach.
- And we are getting it "for free".

Measuring Predictor Importance

- Bagging can (perhaps greatly) improve regression tree performance.
- However, this comes at the expense of easy visualization and easy interpretability — i.e., we no longer enjoy these advantages of a regression tree.
- We can evaluate the importance of each predictor by considering the extent to which it decreases the residual sum of squares, averaged over all trees.
- Let's look at some examples in R.

Bagging for Classification Trees

- Proceeds in a very similar pattern to bagging for regression trees.
- We need a different way to combine the predictions from the M bootstrap samples; this can be done, e.g., via majority vote.
- Rather than looking at MSE, we look at misclassification rate (or ARI).
- People also consider the residual mean deviance:

$$-\frac{2}{n-|T_0|}\sum_{t=1}^{|T_0|}\sum_{g=1}^{G} n_{tg}\log\hat{p}_{tg}.$$

- Let's look at some examples in R.

Random Forests

- An extension of bagging that decorrelates the M trees.
- For each tree, rather than all predictors being available for each split, a random sample of $\mathcal{M}$ is taken at each split.
- A (different) random sample of $\mathcal{M}$ predictors is considered at each split, for each tree.
- Often, $\mathcal{M} = \sqrt{p}$ is used, where p is the number of predictors.

Random Forests contd.

- At first, this might seem counter intuitive — why limit the number of predictors at each split?
- But it prevents one (or a small number) of predictors from dominating the trees.
- In other words, it avoids a situation where the M trees look very similar.
- Let's look at some examples in R.

Comments

- For random forests and bagging, choosing M too large is not a concern.
- One way to choose M is to increase it until the error rate levels off.
- Note that bagging can be used in situations beyond CART.
- Next, we will look at boosting.

Boosting

- Like bagging, boosting can be applied beyond classification trees.
- Unlike bagging, where the M trees are grown independently; in boosting, trees are grown sequentially.
- For each tree, the response is the (current) residuals.
- Each tree can be quite small.

Pseudocode

```
Input: training data X, response Y, M, d, lambda
Set f(x)=0, r_i=y_i
for m in 1:M
     Fit tree f_m(x) to (X, r) using d splits
     f(x) += lambda f_m(x)
     r_i(x) -= lambda f_m(x)
end for
return f(x)
```

Note: this pseudocode is (loosely) based on Algorithm 8.2 in James et al. (2013).

Comments

- Like with bagging and random forests, M is just the number of trees.
- λ is a shrinkage parameter; a small number, e.g., $\lambda = 0.01$.
- d is the number of splits for each tree; usually small, e.g., $d = 1$.
- There is a tradeoff between how small λ is and how large M needs to be.
- Boosting will become more clear after some examples in R.