Bagging and Boosting

Nate Wells

Math 243: Stat Learning

November 6th, 2020

Outline

In today's class, we will. . .

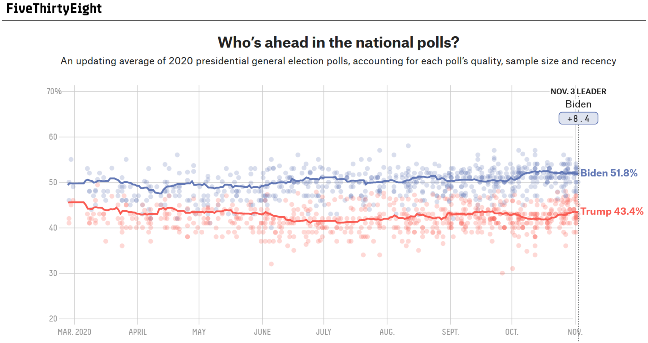
- Discuss bagging and random forests as methods for reducing variance in decision trees
- Investigate boosting as an **learning* method for improving decision trees

Section 1

Bagging and Random Forests

Election Prediction

The 538 blog tracked presidential polls over the course of 2020. How did they come up with a final prediction that Biden would win the popular vote 51.8% to 43.4%?



FiveThirtyEight

Who's ahead in the national polls?

An updating average of 2020 presidential general election polls, accounting for each poll's quality, sample size and recency

Ensemble Methods

Suppose we have m different models to predict Y based on $X_1, \ldots, X_n$. Suppose $\hat{Y}_i$ is the prediction made by the ith model.

Ensemble Methods

Suppose we have m different models to predict Y based on $X_1, \dots, X_n$. Suppose $\hat{Y}_i$ is the prediction made by the ith model.

A simple ensemble model makes a prediction $\hat{Y}$ as the weighted average of the predictions from each model:

$$\hat{Y} = w_1 \hat{Y}_1 + \cdots + w_m \hat{Y}_m \qquad \text{where } w_1 + \dots w_m = 1$$

Ensemble Methods

Suppose we have m different models to predict Y based on $X_1, \ldots, X_n$. Suppose $\hat{Y}_i$ is the prediction made by the ith model.

A simple ensemble model makes a prediction $\hat{Y}$ as the weighted average of the predictions from each model:

$$\hat{Y} = w_1 \hat{Y}_1 + \cdots + w_m \hat{Y}_m \qquad \text{where } w_1 + \ldots w_m = 1$$

Advantages of ensemble models?

Ensemble Methods

Suppose we have m different models to predict Y based on $X_1, \ldots, X_n$. Suppose $\hat{Y}_i$ is the prediction made by the ith model.

A simple ensemble model makes a prediction $\hat{Y}$ as the weighted average of the predictions from each model:

$$\hat{Y} = w_1 \hat{Y}_1 + \cdots + w_m \hat{Y}_m \qquad \text{where } w_1 + \ldots w_m = 1$$

Advantages of ensemble models?

- Significantly more flexible than a single model
- More efficient than single model
- More resilient against model-building bias

Ensemble Methods

Suppose we have m different models to predict Y based on $X_1, \ldots, X_n$. Suppose $\hat{Y}_i$ is the prediction made by the ith model.

A simple ensemble model makes a prediction $\hat{Y}$ as the weighted average of the predictions from each model:

$$\hat{Y} = w_1 \hat{Y}_1 + \cdots + w_m \hat{Y}_m \qquad \text{where } w_1 + \ldots w_m = 1$$

Advantages of ensemble models?

- Significantly more flexible than a single model
- More efficient than single model
- More resilient against model-building bias

Disadvantages?

Ensemble Methods

Suppose we have m different models to predict Y based on $X_1, \ldots, X_n$. Suppose $\hat{Y}_i$ is the prediction made by the ith model.

A simple ensemble model makes a prediction $\hat{Y}$ as the weighted average of the predictions from each model:

$$\hat{Y} = w_1 \hat{Y}_1 + \cdots + w_m \hat{Y}_m \qquad \text{where } w_1 + \ldots w_m = 1$$

Advantages of ensemble models?

- Significantly more flexible than a single model
- More efficient than single model
- More resilient against model-building bias

Disadvantages?

- Making predictions is more computationally expensive
- Favors models with low test time
- Diminishing returns on the number models that can be incorporated in ensemble

Bagging

Suppose we only have one training set, but still want to build an ensemble of regression tree models. How can we do it?

Bagging

Suppose we only have one training set, but still want to build an ensemble of regression tree models. How can we do it?

- Bagging (**B**ootstrap **agg**regation) was one of the earliest ensemble techniques

Bagging

Suppose we only have one training set, but still want to build an ensemble of regression tree models. How can we do it?

- Bagging (**B**ootstrap **agg**regation) was one of the earliest ensemble techniques

To create a bagged model, create many bootstrap samples from the original training set, and fit a decision tree to each. Average the resulting predictions.

Bagging

Suppose we only have one training set, but still want to build an ensemble of regression tree models. How can we do it?

- Bagging (**B**ootstrap **agg**regation) was one of the earliest ensemble techniques

To create a bagged model, create many bootstrap samples from the original training set, and fit a decision tree to each. Average the resulting predictions.

Why?

Bagging

Suppose we only have one training set, but still want to build an ensemble of regression tree models. How can we do it?

- Bagging (**B**ootstrap **agg**regation) was one of the earliest ensemble techniques

To create a bagged model, create many bootstrap samples from the original training set, and fit a decision tree to each. Average the resulting predictions.

Why?

- Recall that decision trees tend to have high variance. But averaging the results of independent (or weakly dependent) variables decreases variance
  - Think about the Central Limit Theorem

Bagging

Suppose we only have one training set, but still want to build an ensemble of regression tree models. How can we do it?

- Bagging (**B**ootstrap **agg**regation) was one of the earliest ensemble techniques

To create a bagged model, create many bootstrap samples from the original training set, and fit a decision tree to each. Average the resulting predictions.

Why?

- Recall that decision trees tend to have high variance. But averaging the results of independent (or weakly dependent) variables decreases variance
  - Think about the Central Limit Theorem
- Unlike a single tree model, we do not prune (we instead control variance by averaging)

Test Error for Bagged Models

Recall from a previous homework that an individual observation has probability $1 - e^{-1} \approx 0.632$ of appearing in a bootstrap sample.

Test Error for Bagged Models

Recall from a previous homework that an individual observation has probability $1 - e^{-1} \approx 0.632$ of appearing in a bootstrap sample.

- For each bootstrap, approximately 1/3 of observations are not included (called *out-of-bag* observations)

Test Error for Bagged Models

Recall from a previous homework that an individual observation has probability $1 - e^{-1} \approx 0.632$ of appearing in a bootstrap sample.

- For each bootstrap, approximately 1/3 of observations are not included (called *out-of-bag* observations)
- The out-of-bag observations can be used as a natural validation set for the bootstrap model.

Test Error for Bagged Models

Recall from a previous homework that an individual observation has probability $1 - e^{-1} \approx 0.632$ of appearing in a bootstrap sample.

- For each bootstrap, approximately $1/3$ of observations are not included (called *out-of-bag* observations)
- The out-of-bag observations can be used as a natural validation set for the bootstrap model.
- We get an overall estimate of test MSE for the bagged model by averaging the MSE of each bootstrap model on its out-of-bag observations

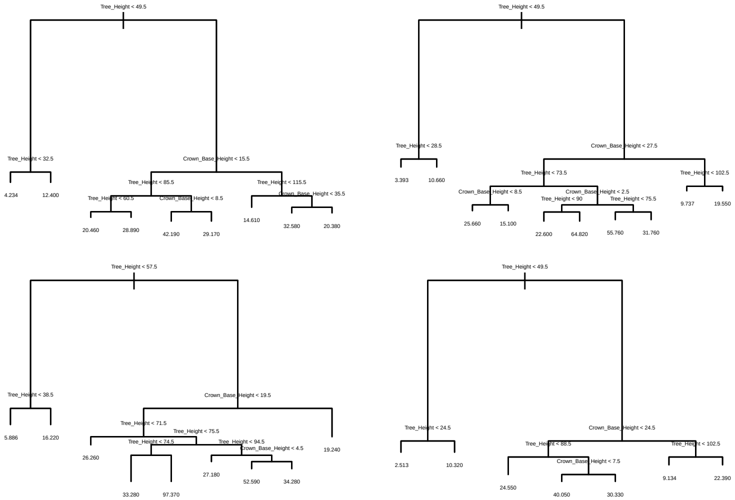
Bagged pdXTrees

```r
set.seed(1)
library(pdxTrees)
all_trees <- get_pdxTrees_parks()
my_trees <- all_trees %>%  select(Pollution_Removal_oz,
                                  Tree_Height,
                                  Crown_Base_Height,
                                  Condition) %>%
  sample_n(1000)
```

```r
set.seed(1)
library(tree)
my_models<-list()


for (i in 1:4){
  bootstrap<-sample_n(my_trees, size = nrow(my_trees), replace = T)
  my_models[[i]]<-tree(Pollution_Removal_oz ~., data = bootstrap)
}
```

A few trees


Tree_Height < 49.5
Tree_Height < 32.5
4.234
12.400
Crown_Base_Height < 15.5
Tree_Height < 85.5
Tree_Height < 60.5
20.460
28.890
Crown_Base_Height < 8.5
42.190
29.170
Tree_Height < 115.5
14.610
Crown_Base_Height < 35.5
32.580
20.380
Tree_Height < 49.5
Tree_Height < 28.5
3.393
10.660
Crown_Base_Height < 27.5
Tree_Height < 73.5
Crown_Base_Height < 8.5
25.660
15.100
Crown_Base_Height < 2.5
Tree_Height < 90
22.600
64.820
Tree_Height < 75.5
55.760
31.760
Tree_Height < 102.5
9.737
19.550
Tree_Height < 57.5
Tree_Height < 38.5
5.886
16.220
Crown_Base_Height < 19.5
Tree_Height < 71.5
26.260
Tree_Height < 75.5
Tree_Height < 74.5
33.280
97.370
Tree_Height < 94.5
27.180
Crown_Base_Height < 4.5
52.590
34.280
19.240
Tree_Height < 49.5
Tree_Height < 24.5
2.513
10.320
Crown_Base_Height < 24.5
Tree_Height < 88.5
24.550
Crown_Base_Height < 7.5
40.050
30.330
Tree_Height < 102.5
9.134
22.390

Performance

```
my_predictions<-list()
for (i in 1:4){
  my_predictions[[i]]<-  predict(my_models[[i]], test_trees )
}

MSE<-c()
for (i in 1:4){
  MSE[i]<-mean((my_predictions[[i]] - test_trees$Pollution_Removal_oz)^2, na.rm = T)
}

data.frame(model = 1:4, MSE)
```

```
##   model      MSE
## 1     1 106.8568
## 2     2 112.4455
## 3     3 126.5883
## 4     4 121.8193
```

Bagged Performance

```
bagged_prediction<-data.frame(model1 = my_predictions[[1]],
                              model2 = my_predictions[[2]],
                              model3 = my_predictions[[3]],
                              model4 = my_predictions[[4]]) %>%
  mutate(bagged = (model1 + model2 + model3 + model4)/4)

head(bagged_prediction)
```

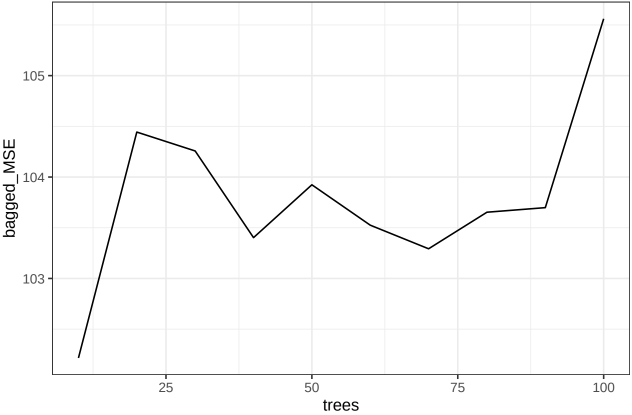
```
##      model1   model2    model3    model4    bagged
## 1 12.403597 16.22254 10.661326 10.321645 12.402276
## 2 42.193827 52.59375 31.763025 40.052174 41.650694
## 3  4.233571  5.88597  3.392694  2.513298  4.006383
## 4 14.612389 19.23693  9.736957  9.134286 13.180141
## 5 28.886577 27.18400 31.763025 24.553114 28.096679
## 6 20.375000 19.23693 19.546667 22.385714 20.386078
```

```
bagged_MSE<-mean((bagged_prediction$bagged - test_trees$Pollution_Removal_oz)^2, na.
data.frame(bagged_MSE)
```

```
##   bagged_MSE
## 1   104.0186
```

The more trees the merrier?

If 4 trees improved performance over 1, what if we bagged 10 trees? 100?



Further Performance Improvements

Suppose we have m ensemble models built from the same data set and that it turns out that all m models are very similar.

Further Performance Improvements

Suppose we have m ensemble models built from the same data set and that it turns out that all m models are very similar.

- Do we expect the ensemble model to have high or low variance?

Further Performance Improvements

Suppose we have m ensemble models built from the same data set and that it turns out that all m models are very similar.

- Do we expect the ensemble model to have high or low variance?
    - High variance (since the models are very correlated)

Further Performance Improvements

Suppose we have m ensemble models built from the same data set and that it turns out that all m models are very similar.

- Do we expect the ensemble model to have high or low variance?
  - High variance (since the models are very correlated)
- When bagging trees, if one predictor accounts for large amount of deviation in the response, it will usually be selected as the first split (regardless of the bootstrap sample used)

Further Performance Improvements

Suppose we have m ensemble models built from the same data set and that it turns out that all m models are very similar.

- Do we expect the ensemble model to have high or low variance?
  - High variance (since the models are very correlated)
- When bagging trees, if one predictor accounts for large amount of deviation in the response, it will usually be selected as the first split (regardless of the bootstrap sample used)
- To artificially increase the variety among trees, we randomly restrict which predictors can be used at each split point.

Further Performance Improvements

Suppose we have m ensemble models built from the same data set and that it turns out that all m models are very similar.

- Do we expect the ensemble model to have high or low variance?
  - High variance (since the models are very correlated)
- When bagging trees, if one predictor accounts for large amount of deviation in the response, it will usually be selected as the first split (regardless of the bootstrap sample used)
- To artificially increase the variety among trees, we randomly restrict which predictors can be used at each split point.
- Although counterintuitive, this restriction tends to increase accuracy of the ensemble by breaking correlations among the participant trees

Random Forests

To create a random forest:

1. Select the number of models m to build and a number of predictors k to use at each step t
2. Generate a bootstrap sample for each model
3. Build a tree on the bootstrap sample where at each step, a random selection of k of the p predictors can be used (independent of prior predictors selected)
4. Aggregate the models to create an ensemble model.

Random Forests

To create a random forest:

1. Select the number of models m to build and a number of predictors k to use at each step t
2. Generate a bootstrap sample for each model
3. Build a tree on the bootstrap sample where at each step, a random selection of k of the p predictors can be used (independent of prior predictors selected)
4. Aggregate the models to create an ensemble model.

Advantages of the random forest?

Random Forests

To create a random forest:

1. Select the number of models m to build and a number of predictors k to use at each step t
2. Generate a bootstrap sample for each model
3. Build a tree on the bootstrap sample where at each step, a random selection of k of the p predictors can be used (independent of prior predictors selected)
4. Aggregate the models to create an ensemble model.

Advantages of the random forest?

- Individual models are less correlated, so ensemble has lower variance
- Each tree is quicker to build (why?)

Random Forests

To create a random forest:

1. Select the number of models *m* to build and a number of predictors *k* to use at each step *t*
2. Generate a bootstrap sample for each model
3. Build a tree on the bootstrap sample where at each step, a random selection of *k* of the *p* predictors can be used (independent of prior predictors selected)
4. Aggregate the models to create an ensemble model.

Advantages of the random forest?

- Individual models are less correlated, so ensemble has lower variance
- Each tree is quicker to build (why?)

Disadvantages?

Random Forests

To create a random forest:

1. Select the number of models m to build and a number of predictors k to use at each step t
2. Generate a bootstrap sample for each model
3. Build a tree on the bootstrap sample where at each step, a random selection of k of the p predictors can be used (independent of prior predictors selected)
4. Aggregate the models to create an ensemble model.

Advantages of the random forest?

- Individual models are less correlated, so ensemble has lower variance
- Each tree is quicker to build (why?)

Disadvantages?

- Difficult to interpret
- Theoretically properties less well-studied

Hand-drawn Example

Random Forests in R

To create both bagged trees and random forests, we use the `randomForest` function in the `randomForest` package in R:

```
set.seed(1)
library(randomForest)
rfmodel <- randomForest(Pollution_Removal_oz ~ ., data = my_trees_na)
rfmodel
```

```
##
## Call:
##  randomForest(formula = Pollution_Removal_oz ~ ., data = my_trees_na)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 153.6827
##                     % Var explained: 44.36
```

Modifications

We can control how many trees are generated with `ntrees =` and the number of predictors at each split with `mtry=`

Modifications

We can control how many trees are generated with `ntrees =` and the number of predictors at each split with `mtry=`

- By default, `randomForest` uses $p/3$ predictors for regression and $\sqrt{p}$ predictors for classification

Modifications

We can control how many trees are generated with `ntrees =` and the number of predictors at each split with `mtry=`

- By default, `randomForest` uses $p/3$ predictors for regression and $\sqrt{p}$ predictors for classification

```r
set.seed(1)
rfmodel2 <- randomForest(Pollution_Removal_oz ~ ., data = my_trees_na,
                         ntrees = 10, mtry = 3)
rfmodel2
```

```
##
## Call:
##  randomForest(formula = Pollution_Removal_oz ~ ., data = my_trees_na,      ntrees
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 170.2656
##                     % Var explained: 38.36
```

Modifications

We can control how many trees are generated with `ntrees =` and the number of predictors at each split with `mtry=`

- By default, `randomForest` uses $p/3$ predictors for regression and $\sqrt{p}$ predictors for classification

```r
set.seed(1)
rfmodel2 <- randomForest(Pollution_Removal_oz ~ ., data = my_trees_na,
                         ntrees = 10, mtry = 3)
rfmodel2
```

```
##
## Call:
##  randomForest(formula = Pollution_Removal_oz ~ ., data = my_trees_na,      ntrees
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 170.2656
##                     % Var explained: 38.36
```

How can we create a bagged model using the `randomForest` function?

Modifications

We can control how many trees are generated with `ntrees =` and the number of predictors at each split with `mtry=`

- By default, `randomForest` uses $p/3$ predictors for regression and $\sqrt{p}$ predictors for classification

```
set.seed(1)
rfmodel2 <- randomForest(Pollution_Removal_oz ~ ., data = my_trees_na,
                         ntrees = 10, mtry = 3)
rfmodel2
```

```
##
## Call:
##  randomForest(formula = Pollution_Removal_oz ~ ., data = my_trees_na,      ntrees
##                 Type of random forest: regression
##                       Number of trees: 500
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 170.2656
##                     % Var explained: 38.36
```

How can we create a bagged model using the `randomForest` function?

- Set `mtry= p`, where `p` is the total number predictors available

Making predictions

So you have your randomForest model. How do you make predictions?

```
my_preds<- predict(rfmodel, test_trees)

data.frame(my_preds,actual = test_trees$Pollution_Removal_oz) %>% head()
```

```
##    my_preds actual
## 1 14.089043   16.6
## 2 31.478264   14.7
## 3  6.004437    0.2
## 4 19.351968   15.0
## 5 28.102784   41.4
## 6 20.041636   10.5
```