Bagging and Random Forests

Nate Wells

Math 243: Stat Learning

November 15th, 2021

Outline

In today's class, we will...

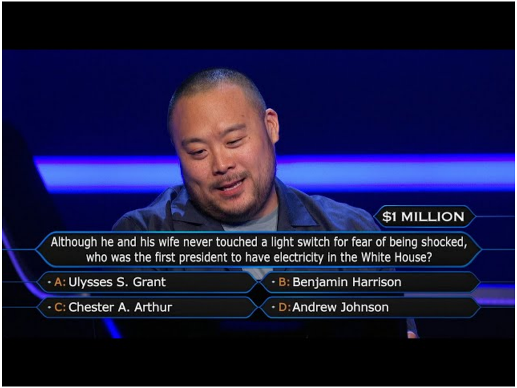
- Introduce ensemble modeling as means of improving low accuracy models
- Discuss bagging and random forests as methods for reducing variance in decision trees
- Implement random forests in R

Section 1

Ensemble Models

Who Wants to Be a Millionaire?

- *Who Wants to Be a Millionaire* is a television gameshow that debuted in the 1990s and in which contestants answer a series of increasingly difficult multiple choice questions in order to win the grand prize of $1,000,000.



Who Wants to Be a Millionaire?

- The original show included 3 “lifeline” options contestants could use to answer questions:
  - **50:50**: Two randomly selected incorrect answers are eliminated
  - **Phone a Friend**: The contestant calls a friend and is given 30 seconds to discuss
  - **Ask the Audience**: Audience members each vote on the answer they think is correct

Who Wants to Be a Millionaire?

- The original show included 3 "lifeline" options contestants could use to answer questions:
  - **50:50**: Two randomly selected incorrect answers are eliminated
  - **Phone a Friend**: The contestant calls a friend and is given 30 seconds to discuss
  - **Ask the Audience**: Audience members each vote on the answer they think is correct

- Which lifeline has the highest chance of producing the correct answer?

Who Wants to Be a Millionaire?

- The original show included 3 "lifeline" options contestants could use to answer questions:
  - **50:50**: Two randomly selected incorrect answers are eliminated
  - **Phone a Friend**: The contestant calls a friend and is given 30 seconds to discuss
  - **Ask the Audience**: Audience members each vote on the answer they think is correct

- Which lifeline has the highest chance of producing the correct answer?

Who Wants to Be a Millionaire?

- The original show included 3 “lifeline” options contestants could use to answer questions:
  - **50:50**: Two randomly selected incorrect answers are eliminated
  - **Phone a Friend**: The contestant calls a friend and is given 30 seconds to discuss
  - **Ask the Audience**: Audience members each vote on the answer they think is correct

- Which lifeline has the highest chance of producing the correct answer?

Why?

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 $X_1, \ldots, X_n$. 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:

$$\hat{Y} = w_1 \hat{Y}_1 + \cdots + w_m \hat{Y}_m \qquad \text{where } w_1 + \ldots w_m = 1, \quad w_i \geq 0$$

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:

$$\hat{Y} = w_1 \hat{Y}_1 + \cdots + w_m \hat{Y}_m \qquad \text{where } w_1 + \ldots w_m = 1, \quad w_i \geq 0$$

- 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:

$$\hat{Y} = w_1 \hat{Y}_1 + \cdots + w_m \hat{Y}_m \qquad \text{where } w_1 + \ldots w_m = 1, \quad w_i \geq 0$$

- 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 $X_1, \dots, X_n$. 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:

$$\hat{Y} = w_1 \hat{Y}_1 + \cdots + w_m \hat{Y}_m \qquad \text{where } w_1 + \dots w_m = 1, \quad w_i \geq 0$$

- 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:

$$\hat{Y} = w_1 \hat{Y}_1 + \cdots + w_m \hat{Y}_m \qquad \text{where } w_1 + \ldots w_m = 1, \quad w_i \geq 0$$

- 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

Section 2

Bagging

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

A Bag of Trees

We return to the pdxTrees data set, this time expanding both our data set size and number of predictors:

```
names(my_pdxTrees)
```

```
## [1] "DBH"                        "Condition"
## [3] "Tree_Height"                "Crown_Width_NS"
## [5] "Crown_Width_EW"             "Crown_Base_Height"
## [7] "Functional_Type"            "Mature_Size"
## [9] "Carbon_Sequestration_lb"
```

```
dim(my_pdxTrees)
```

```
## [1] 3015    9
```

```
set.seed(1)
library(rsample)
my_pdxTrees_split <- initial_split(my_pdxTrees )
my_pdxTrees_train <- training(my_pdxTrees_split)
my_pdxTrees_test <- testing(my_pdxTrees_split)
```

A Bag of Trees

We return to the pdxTrees data set, this time expanding both our data set size and number of predictors:

```
names(my_pdxTrees)
```

```
## [1] "DBH"                        "Condition"
## [3] "Tree_Height"                "Crown_Width_NS"
## [5] "Crown_Width_EW"             "Crown_Base_Height"
## [7] "Functional_Type"            "Mature_Size"
## [9] "Carbon_Sequestration_lb"
```

```
dim(my_pdxTrees)
```

```
## [1] 3015    9
```

```
set.seed(1)
library(rsample)
my_pdxTrees_split <- initial_split(my_pdxTrees )
my_pdxTrees_train <- training(my_pdxTrees_split)
my_pdxTrees_test <- testing(my_pdxTrees_split)
```

- Can we improve on our previous model predicting `Carbon_Sequestration_lb`, now using more data and more predictors?

Bagged pdXTrees

- Let's get a few bootstrap samples using `rsample`:

Bagged pdXTrees

- Let's get a few bootstrap samples using `rsample`:

```
library(rsample)
set.seed(1115)
pdx_bootstrap <- bootstraps(my_pdxTrees_train, times = 4)
```

Bagged pdXTrees

- Let's get a few bootstrap samples using `rsample`:

```
library(rsample)
set.seed(1115)
pdx_bootstrap <- bootstraps(my_pdxTrees_train, times = 4)
```

- And now build trees on each:

Bagged pdXTrees

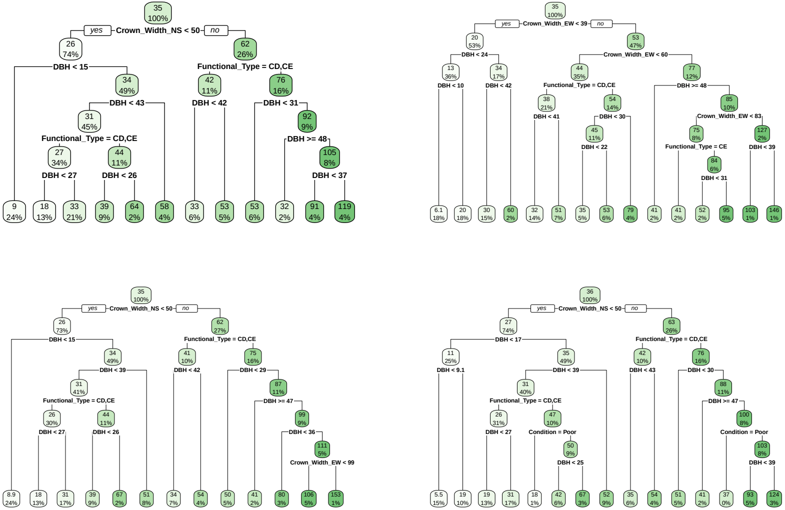
- Let's get a few bootstrap samples using rsample:

```
library(rsample)
set.seed(1115)
pdx_bootstrap <- bootstraps(my_pdxTrees_train, times = 4)
```

- And now build trees on each:

```
library(rpart)
get_tree <- function(split){
  bootstrap_sample <- analysis(split)
  model <- rpart(Carbon_Sequestration_lb ~., data = bootstrap_sample)
}
pdx_bootstrap$model <- map(pdx_bootstrap$splits, get_tree)
```

A few trees



Performance

- Let's get predictions for each bootstrap:

Performance

- Let's get predictions for each bootstrap:

```
get_predictions <- function(model){
  predictions <- predict(model, my_pdxTrees_test)
  tibble(obs = my_pdxTrees_test$Carbon_Sequestration_lb, preds = predictions)
}
pdx_bootstrap$predictions <- map(pdx_bootstrap$model, get_predictions)
```

Performance

- Let's get predictions for each bootstrap:

```
get_predictions <- function(model){
  predictions <- predict(model, my_pdxTrees_test)
  tibble(obs = my_pdxTrees_test$Carbon_Sequestration_lb, preds = predictions)
}
pdx_bootstrap$predictions <- map(pdx_bootstrap$model, get_predictions)
```

- And calculate `rmse` on each using `yardstick`

Performance

- Let's get predictions for each bootstrap:

```
get_predictions <- function(model){
  predictions <- predict(model, my_pdxTrees_test)
  tibble(obs = my_pdxTrees_test$Carbon_Sequestration_lb, preds = predictions)
}
pdx_bootstrap$predictions <- map(pdx_bootstrap$model, get_predictions)
```

- And calculate rmse on each using yardstick

```
library(yardstick)
results <- map_dfr(pdx_bootstrap$predictions, rmse, obs, preds)
results
```

```
## # A tibble: 4 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        14.0
## 2 rmse    standard        13.2
## 3 rmse    standard        15.1
## 4 rmse    standard        12.8
```

```
mean(results$.estimate)
```

```
## [1] 13.75185
```

Variation in Model Predictions

- How do individual tree predictions compare?

Variation in Model Predictions

- How do individual tree predictions compare?

```
## # A tibble: 6 x 5
## # Rowwise:
##   tree1 tree2 tree3 tree4 bagged
##   <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1  91.3  79.9  95.5  93.4   90.0
## 2  39.5  39.2  19.6  42.5   35.2
## 3  91.3  79.9  95.5  93.4   90.0
## 4 119.  106.   78.8 124.   107.
## 5  53.1  50.2  52.3  50.9   51.6
## 6  64.2  67.3  95.5  67.1   73.5
```

Variation in Model Predictions

- How do individual tree predictions compare?

```
## # A tibble: 6 x 5
## # Rowwise:
##   tree1 tree2 tree3 tree4 bagged
##   <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1  91.3  79.9  95.5  93.4   90.0
## 2  39.5  39.2  19.6  42.5   35.2
## 3  91.3  79.9  95.5  93.4   90.0
## 4 119.  106.   78.8 124.   107.
## 5  53.1  50.2  52.3  50.9   51.6
## 6  64.2  67.3  95.5  67.1   73.5
```

- How does the bagged model RMSE compare to each individual tree's RMSE?

Variation in Model Predictions

- How do individual tree predictions compare?

```
## # A tibble: 6 x 5
## # Rowwise:
##   tree1 tree2 tree3 tree4 bagged
##   <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1  91.3  79.9  95.5  93.4   90.0
## 2  39.5  39.2  19.6  42.5   35.2
## 3  91.3  79.9  95.5  93.4   90.0
## 4 119.  106.   78.8 124.   107.
## 5  53.1  50.2  52.3  50.9   51.6
## 6  64.2  67.3  95.5  67.1   73.5
```

- How does the bagged model RMSE compare to each individual tree's RMSE?

```
## # A tibble: 5 x 4
##   model  .metric .estimator .estimate
##   <chr>  <chr>   <chr>          <dbl>
## 1 tree 1 rmse    standard        14.0
## 2 tree 2 rmse    standard        13.2
## 3 tree 3 rmse    standard        15.1
## 4 tree 4 rmse    standard        12.8
## 5 bagged rmse    standard        12.4
```

Variation in Model Predictions

- How do individual tree predictions compare?

```
## # A tibble: 6 x 5
## # Rowwise:
##   tree1 tree2 tree3 tree4 bagged
##   <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1  91.3  79.9  95.5  93.4   90.0
## 2  39.5  39.2  19.6  42.5   35.2
## 3  91.3  79.9  95.5  93.4   90.0
## 4 119.  106.   78.8 124.   107.
## 5  53.1  50.2  52.3  50.9   51.6
## 6  64.2  67.3  95.5  67.1   73.5
```

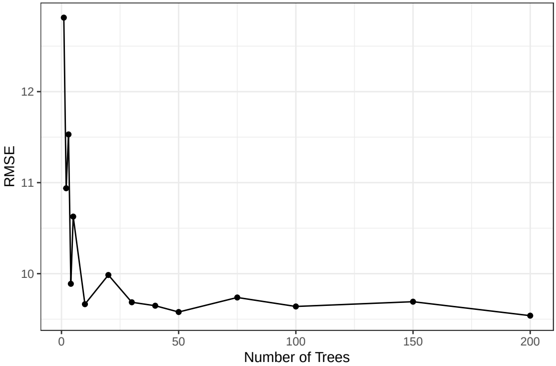
- How does the bagged model RMSE compare to each individual tree's RMSE?

```
## # A tibble: 5 x 4
##   model  .metric .estimator .estimate
##   <chr>  <chr>   <chr>          <dbl>
## 1 tree 1 rmse    standard        14.0
## 2 tree 2 rmse    standard        13.2
## 3 tree 3 rmse    standard        15.1
## 4 tree 4 rmse    standard        12.8
## 5 bagged rmse    standard        12.4
```

- Note that the RMSE for the bagged tree is **NOT** simply the average RMSE. It is significantly *lower*!

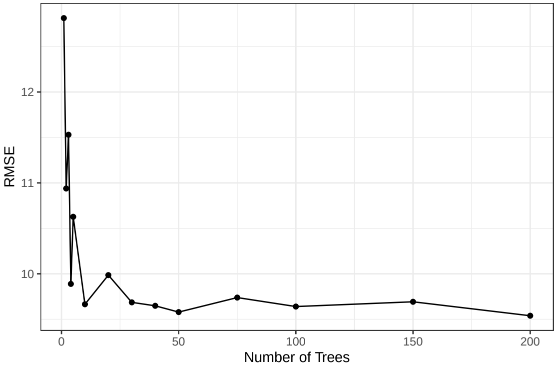
The more trees the merrier?

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



The more trees the merrier?

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



- Greatest gains by adding a small number of additional trees
- Moderately small gains thereafter

Section 3

Random Forests

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 (possible Senior Thesis project!)

Hand-drawn Example

Section 4

Bagging and Random Forests in R

Random Forest in R

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

```
library(randomForest)
rfmodel <- randomForest(Carbon_Sequestration_lb ~ ., data = my_pdxTrees_train)
rfmodel
```

```
##
## Call:
##  randomForest(formula = Carbon_Sequestration_lb ~ ., data = my_pdxTrees_train)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##           Mean of squared residuals: 123.172
##                     % Var explained: 84.48
```

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

```
set.seed(1)
rfmodel2 <- randomForest(Carbon_Sequestration_lb ~ ., data = my_pdxTrees_train,
                         ntree = 10, mtry = 5)
rfmodel2
```

```
##
## Call:
##  randomForest(formula = Carbon_Sequestration_lb ~ ., data = my_pdxTrees_train,
##                Type of random forest: regression
##                      Number of trees: 10
## No. of variables tried at each split: 5
##
##           Mean of squared residuals: 142.6305
##                     % Var explained: 82.02
```

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(Carbon_Sequestration_lb ~ ., data = my_pdxTrees_train,
                         ntree = 10, mtry = 5)
rfmodel2
```

```
##
## Call:
##  randomForest(formula = Carbon_Sequestration_lb ~ ., data = my_pdxTrees_train,
##                Type of random forest: regression
##                      Number of trees: 10
## No. of variables tried at each split: 5
##
##           Mean of squared residuals: 142.6305
##                     % Var explained: 82.02
```

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(Carbon_Sequestration_lb ~ ., data = my_pdxTrees_train,
                         ntree = 10, mtry = 5)
rfmodel2
```

```
##
## Call:
##  randomForest(formula = Carbon_Sequestration_lb ~ ., data = my_pdxTrees_train,
##                Type of random forest: regression
##                      Number of trees: 10
## No. of variables tried at each split: 5
##
##           Mean of squared residuals: 142.6305
##                     % Var explained: 82.02
```

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

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