HW 7

Runze Wang

You can run the following code to prepare the analysis.

```r
library(r02pro)     #INSTALL IF NECESSARY
library(tidyverse)  #INSTALL IF NECESSARY
library(MASS)
library(tree)
my_ahp <- ahp %>% dplyr::select(gar_car, liv_area, lot_area, oa_qual, sale_price) %>%
  na.omit() %>%
  mutate(type = factor(ifelse(sale_price > median(sale_price), "Expensive", "Cheap")))
tr_ind <- 1:(nrow(my_ahp)/20)
my_ahp_train <- my_ahp[tr_ind, ]
my_ahp_test <- my_ahp[-tr_ind, ]
```

0.1 Suppose we want to use tree, bagging, random forest, and boosting to predict `sale_price` and `type` using variables`gar_car`, `liv_area`, `lot_area`, and `oa_qual`. Please answer the following questions. 1. Predict `sale_price` (a continuous response) using the training data `my_ahp_train` with tree (with CV pruning), bagging, random forest, and boosting (with CV for selecting the number of trees to be used). For each method, compute the training and test MSE. (For boosting, please set `n.trees = 5000, interaction.depth = 1, cv.folds = 5`)

```r
## tree
sale.tree <- tree(sale_price ~ gar_car+liv_area+lot_area+oa_qual, data =my_ahp_train)
cv.sale <- cv.tree(sale.tree)
bestsize <- cv.sale$size[which.min(cv.sale$dev)]
sale.tree.prune <- prune.tree(sale.tree, best = bestsize)
prediction_train.tree <- predict(sale.tree.prune, newdata = my_ahp_train)
mean((my_ahp_train$sale_price - prediction_train.tree)^2)
```

```
## [1] 1016.612
```

```r
prediction_test.tree <- predict(sale.tree.prune, newdata = my_ahp_test)
mean((my_ahp_test$sale_price - prediction_test.tree)^2)
```

```
## [1] 2172.134
```

```r
## bagging
library(randomForest)
set.seed(1)
p <- ncol(my_ahp_train) - 2
bag.sale <- randomForest(sale_price ~ gar_car + liv_area + lot_area + oa_qual, data = my_ahp_train, mtr
bagging_predictions_train <- predict(bag.sale, newdata =my_ahp_train)
bagging_predictions_test <- predict(bag.sale, newdata =my_ahp_test)
```

```
mean((bagging_predictions_train - my_ahp_train$sale_price)^2)
```

```
## [1] 275.1146
```

```
mean((bagging_predictions_test - my_ahp_test$sale_price)^2)
```

```
## [1] 1389.205
```

```
  ## random forest
set.seed(1)
rf.sale <- randomForest(sale_price ~ gar_car + liv_area + lot_area + oa_qual, data = my_ahp_train,impor
rf_predictions_train <- predict(rf.sale, newdata =my_ahp_train)
rf_predictions_test <- predict(rf.sale, newdata =my_ahp_test)
mean((rf_predictions_train - my_ahp_train$sale_price)^2)
```

```
## [1] 561.7248
```

```
mean((rf_predictions_test - my_ahp_test$sale_price)^2)
```

```
## [1] 1296.647
```

```
## boosting
library(gbm)
set.seed(1)
boost.sale <- gbm(sale_price ~ gar_car + liv_area + lot_area + oa_qual,data = my_ahp_train, distributio
boosting_predictions_train <- predict(boost.sale, newdata =my_ahp_train, n.trees = 5000)
boosting_predictions_test <- predict(boost.sale, newdata =my_ahp_test, n.trees =5000)
mean((boosting_predictions_train - my_ahp_train$sale_price)^2)
```

```
## [1] 276.297
```

```
mean((boosting_predictions_test - my_ahp_test$sale_price)^2)
```

```
## [1] 2458.177
```

0.2 2.Predict type (a binary response) using the training data my_ahp_train with tree (with CV pruning), bagging, random forest, and boosting (with CV for selecting the number of trees to be used). For each method, compute the training and test classification error. (For boosting, please set n.trees = 5000, interaction.depth = 1, cv.folds = 5)

```
## tree
tree_fit2 <- tree(type ~ gar_car+liv_area+lot_area+oa_qual, data = my_ahp_train)
cv_tree_fit2 <- cv.tree(tree_fit2)
best_size2 <- cv_tree_fit2$size[which.min(cv_tree_fit2$dev)]
tree_prune2 <- prune.tree(tree_fit2, best = best_size2)
tree_predictions_train <- predict(tree_prune2, my_ahp_train,type = "class")
tree_predictions_test <- predict(tree_prune2, my_ahp_test, type = "class")
mean(tree_predictions_train != my_ahp_train$type)
```

```
## [1] 0.1764706
```

```
mean(tree_predictions_test != my_ahp_test$type)
```

```
## [1] 0.2185567
```

```
## bagging
set.seed(1)
p <- ncol(my_ahp_train) - 2
bag.type <- randomForest(type ~ gar_car+liv_area+lot_area+oa_qual, data = my_ahp_train, mtry = p, impor
bag.type_predictions_train <- predict(bag.type,  newdata = my_ahp_train)
bag.type_predictions_test <- predict(bag.type,  newdata = my_ahp_test)
mean(bag.type_predictions_train != my_ahp_train$type)
```

```
## [1] 0
```

```
mean(bag.type_predictions_test != my_ahp_test$type)
```

```
## [1] 0.1479381
```

```
## random forest
set.seed(1)
rf.type <- randomForest(type ~ gar_car+liv_area+lot_area+oa_qual, data = my_ahp_train,importance = TRUE
rf_predictions_train2 <- predict(rf.type,  newdata = my_ahp_train)
rf_predictions_test2 <- predict(rf.type,  newdata = my_ahp_test)
mean((rf_predictions_train2 != my_ahp_train$type)^2)
```

```
## [1] 0
```

```
mean((rf_predictions_test2 != my_ahp_test$type)^2)
```

```
## [1] 0.1304124
```

```
## boosting

boost.type <- gbm(type ~gar_car + liv_area + lot_area + oa_qual,data = my_ahp_train, distribution = "mul
```

```
## Warning: Setting `distribution = "multinomial"` is ill-advised as it is
## currently broken. It exists only for backwards compatibility. Use at your own
## risk.
```

```
boosting_predictions_train2 <- predict(boost.type, newdata = my_ahp_train, n.trees = 5000, type = "resp
boosting_predictions_test2 <- predict(boost.type, newdata = my_ahp_test, n.trees = 5000, type = "respons
```

```r
train_pred_boost <- levels(my_ahp_train$type)[apply(boosting_predictions_train2, 1, which.max)]
test_pred_boost <- levels(my_ahp_test$type)[apply(boosting_predictions_test2, 1, which.max)]
mean((train_pred_boost != my_ahp_train$type)^2)
```

```
## [1] 0
```

```r
mean((test_pred_boost != my_ahp_test$type)^2)
```

```
## [1] 0.1819588
```

0.3 Question 8.4.2 on Page 362 in ISLRv2.

When $d = 1$ in

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)$$

,in which case each tree is a stump, consisting of a single split. All these terms are summed up making the model additive.

0.4 4. Question 8.4.5 on Page 362 in ISLRv2. Suppose we produce ten bootstrapped samples from a data set containing red and green classes. We then apply a classification tree to each bootstrapped sample and, for a specific value of X, produce 10 estimates of P(Class is Red|X): 0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, and 0.75. There are two common ways to combine these results together into a single class prediction. One is the majority vote approach discussed in this chapter. The second approach is to classify based on the average probability. In this example, what is the final classification under each of these two approaches?

For majoritry vote, 6 out of 10, the $P(ClassisRed|X)$ is geater than $P(ClassisGreen|X)$. The final classification is red.

For average probability, the avarage probability of $P(ClassisRed|X)$ is 0.45 and the avarage probability of $P(ClassisGreen|X)$ is 0.55. The final classification is green.