hw17

111078513

2023-06-07

```
# Download the dataset and read it into R
# Kaggle:https://www.kaggle.com/datasets/teertha/ushealthinsurancedataset
insurance <- read.csv("insurance.csv", header = T, sep = ",")
```

Question 1) Create some explanatory models to learn more about charges:

a. Create an OLS regression model and report which factors are significantly related to charges

```
insurance_full_lm <- lm(charges ~ age + sex + bmi + children +
                          smoker + region, data = insurance)
summary(insurance_full_lm)
```

```
##
## Call:
## lm(formula = charges ~ age + sex + bmi + children + smoker +
##     region, data = insurance)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11304.9  -2848.1   -982.1   1393.9  29992.8
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -11938.5      987.8 -12.086  < 2e-16 ***
## age                256.9       11.9  21.587  < 2e-16 ***
## sexmale           -131.3      332.9  -0.394 0.693348
## bmi                339.2       28.6  11.860  < 2e-16 ***
## children           475.5      137.8   3.451 0.000577 ***
## smokeryes        23848.5      413.1  57.723  < 2e-16 ***
## regionnorthwest   -353.0      476.3  -0.741 0.458769
## regionsoutheast  -1035.0      478.7  -2.162 0.030782 *
## regionsouthwest   -960.0      477.9  -2.009 0.044765 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6062 on 1329 degrees of freedom
## Multiple R-squared:  0.7509, Adjusted R-squared:  0.7494
## F-statistic: 500.8 on 8 and 1329 DF,  p-value: < 2.2e-16
```

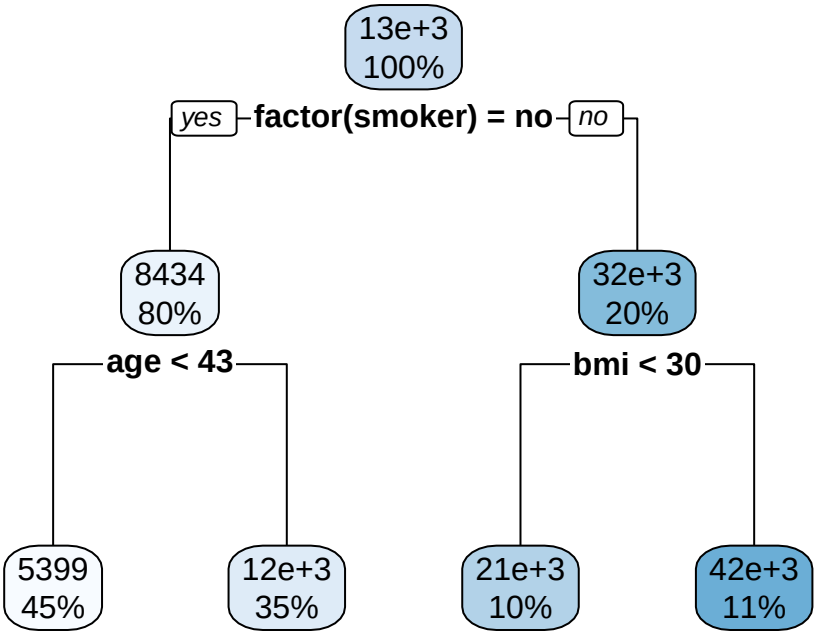
According to the summary of the OLS regression model, age, bmi, children, smoker, and region are significantly

related to charge.

b. Create a decision tree (specifically, a regression tree) with default parameters to rpart().

(i) Plot a visual representation of the tree structure

```
library(rpart)
library(rpart.plot)
insurance_full_rpart <- rpart(charges ~ age + factor(sex) + bmi + children +
                              factor(smoker) + factor(region), data = insurance)
rpart.plot(insurance_full_rpart)
```



(ii) How deep is the tree?

Based on the plot we drew above, we can notice that there are two decisions in the trees.

(iii) How many leaf groups does it suggest to bin the data into?

Based on the plot we drew above as well, it suggest to bin the data into four leaf groups.

(iv) What conditions (combination of decisions) describe each leaf group?

There are four conditions based on the information of the tree:

1. The guys who are smokers and under age 43.
2. The guys who are smokers but over age 43.
3. The guys who are not smokers and their bmi are under 30.
4. The guys who are not smokers and their bmi are over 30.

Question 2) Let's use LOOCV to see how how our models perform predictively overall

```
# Some functions we'll need.
mse <- function(errs) mean(errs^2)
mse_in <- function(model) mse(residuals(model))
mse_out <- function(actual, predicted) mse(actual - predicted)

fold_i_pe <- function(i, k, model, dataset, outcome){
  folds <- cut(1:nrow(dataset), breaks=k, labels=FALSE)
  test_indices <- which(folds==i)
  test_set <- dataset[test_indices, ]
  train_set <- dataset[-test_indices, ]
  trained_model <- update(model, data = train_set)
  predictions <- predict(trained_model, test_set)
  dataset[test_indices, outcome] - predictions
}

k_fold_mse <- function(model, dataset, outcome, k=nrow(dataset)){
  shuffled_indicies <- sample(1:nrow(dataset))
  dataset <- dataset[shuffled_indicies,]
  fold_pred_errors <- sapply(1:k, \(kth){
    fold_i_pe(kth, k, model, dataset, outcome)})
  pred_errors <- unlist(fold_pred_errors)
  mse(pred_errors)
}
```

a. What is the $RMSE_{out}$ for the OLS regression model

```
lm_RMSE_out <- sqrt(k_fold_mse(insurance_full_lm, insurance, "charges"))
print(paste("The RMSE_out for the OLS regression model is ",lm_RMSE_out, sep = ""))
```

```
## [1] "The RMSE_out for the OLS regression model is 6087.38800655031"
```

b. What is the $RMSE_{out}$ for the decision tree model?

```
rpart_RMSE_out <- sqrt(k_fold_mse(insurance_full_rpart, insurance, "charges"))
print(paste("The RMSE_out for the decision tree model is ",rpart_RMSE_out, sep = ""))
```

```
## [1] "The RMSE_out for the decision tree model is 5135.1747343426"
```

Question 3) Let's see if bagging helps our models

For bagging and boosting, we will only use split-sample testing to save time: partition the data to create training and test sets using an 80:20 split. Use the regression model and decision tree you created in Question 1 for bagging and boosting.

```
shuffled_indicies <- sample(1:nrow(insurance))[1:(nrow(insurance)*0.8)]
train_set <- insurance[shuffled_indicies, ]
test_set <- insurance[-shuffled_indicies, ]
```

a. Implement the bagged_learn(...) and bagged_predict(...) functions using the hints in the class notes and help from your classmates on Teams. Feel free to share your code on Teams to get feedback, or ask others for help.

```
bagged_learn <- function(model, dataset, b=100) {
  lapply(1:b, \(i) {
    n = nrow(dataset)
    train_set <- dataset[sample(1:n, n, replace = TRUE), ]
    update(model, data = train_set)
  })
}

bagged_predict <- function(bagged_models, new_data) {
  predictions <- lapply(bagged_models, \(model){
    predict(model, new_data)
  })
  as.data.frame(predictions) |> apply(X = _, 1, mean)
}

rmse_oos<-function(actuals,preds){
  sqrt(mean( (actuals - preds)^2 ))
}
```

b. What is the $RMSE_{out}$ for the bagged OLS regression?

```
bagged_OLS_RMSE_out <- bagged_learn(insurance_full_lm, train_set, b=100) |>
  bagged_predict(test_set) |>
  rmse_oos(test_set$charges, preds = _)
print(paste("The RMSE_out for the bagged OLS regression is ",bagged_OLS_RMSE_out, sep = ""))
```

```
## [1] "The RMSE_out for the bagged OLS regression is 6281.79206159755"
```

c. What is the $RMSE_{out}$ for the bagged decision tree?

```
bagged_rpart_RMSE_out <- bagged_learn(insurance_full_rpart, train_set, b=100) |>
  bagged_predict(test_set) |>
  rmse_oos(test_set$charges, preds = _)
print(paste("The RMSE_out for the bagged decision tree is ",bagged_rpart_RMSE_out, sep = ""))
```

```
## [1] "The RMSE_out for the bagged decision tree is 5046.88905622828"
```

Question 4) Let's see if boosting helps our models. You can use a learning rate of 0.1 and adjust it if you find a better rate.

a. Write boosted_learn(...) and boosted_predict(...) functions using the hints in the class notes and help from your classmates on Teams. Feel free to share your code generously on Teams to get feedback, or ask others for help.

```
boost_learn <- function(model, dataset, outcome, n=100, rate=0.1){
  predictors <- dataset[, !(colnames(dataset) == outcome)]
  res <- dataset[, c(outcome)]
  models <- list()
```

```
  for (i in 1:n) {
    this_model <- update(model, data = cbind(predictors, charges=res))
    res <-  res - rate*predict(this_model, predictors)
    models[[i]] <- this_model
  }
  list(models=models, rate=rate)
}

boost_predict <- function(boosted_learning, new_data) {
  boosted_models <- boosted_learning$models
  rate <- boosted_learning$rate
  n <- nrow(new_data)
  predictions <- lapply(boosted_models, \(model){predict(model, new_data)})
  pred_frame <- as.data.frame(predictions) |> unname()
  apply(pred_frame, 1, \(predict)(0.1*sum(predict)))
}
```

b. What is the $RMSE_{out}$ for the boosted OLS regression?

```
boost_OLS_RMSE_out <-
  boost_learn(insurance_full_lm, train_set, outcome="charges", n=1000) |>
  boost_predict(test_set) |>
  rmse_oos(test_set$charges, preds = _)
print(paste("The RMSE_out for the Boosted OLS regression is ",boost_OLS_RMSE_out, sep = ""))
```

```
## [1] "The RMSE_out for the Boosted OLS regression is 6283.02768752982"
```

c. What is the $RMSE_{out}$ for the boosted decision tree?

```
boost_rpart_RMSE_out <-
  boost_learn(insurance_full_rpart, train_set, outcome="charges", n=1000) |>
  boost_predict(test_set) |>
  rmse_oos(test_set$charges, preds = _)
print(paste("The RMSE_out for the Boosted decision tree is ",boost_rpart_RMSE_out, sep = ""))
```

```
## [1] "The RMSE_out for the Boosted decision tree is 4690.16919581886"
```

Question 5) Let's engineer the best predictive decision trees. Let's repeat the bagging and boosting decision tree several times to see what kind of base tree helps us learn the fastest. But this time, split the data 70:20:10 — use 70% for training, 20% for fine-tuning, and use the last 10% to report the final $RMSE_{out}$.

```
shuffled_indicies <- sample(1:nrow(insurance))
n = nrow(insurance)
train_set <- insurance[shuffled_indicies[1:(n*0.7)], ]
tuning_set <- insurance[shuffled_indicies[(n*0.7):(n*0.9)], ]
test_set <- insurance[shuffled_indicies[(n*0.9):n], ]
```

a. Repeat the bagging of the decision tree, using a base tree of maximum depth 1, 2, . . . n, keep training on the 70% training set while the $RMSE_{out}$ of your 20% set keeps dropping; stop when the $RMSE_{out}$ has started increasing again (show prediction error at each depth). Report the final $RMSE_{out}$ using the final 10% of the data as your test set.

```
# Prepared the function we'll need.
repeat_bagging <- function(model, train_set, tuning_set){
  this_model <- update(model, control = rpart.control(maxdepth = 1))
  rmse_out <- bagged_learn(this_model, train_set, b=100) |>
    bagged_predict(tuning_set) |>
    rmse_oos(tuning_set$charges, preds = _)
  k <- 1
  print(paste("--- k =", k, "---"))
  print(rmse_out)
  while(TRUE){
    k <- k + 1
    print(paste("--- k =", k, "---"))
    control_text <- paste("update(this_model, control = rpart.control(maxdepth =",
                          k, "))")
    temp_model <- eval(parse(text = control_text))
    rmse_out_temp <- bagged_learn(temp_model, train_set, b=100) |>
      bagged_predict(tuning_set) |>
      rmse_oos(tuning_set$charges, preds = _)
    if (rmse_out_temp > rmse_out){
      print(paste(rmse_out_temp, "(Stop)"))
      result = list(model = this_model, k = k-1)
      return(result)
    }
    rmse_out <- rmse_out_temp
    this_model <- temp_model
    print(rmse_out)
  }
}
```

```
best_tree_bagging <- repeat_bagging(insurance_full_rpart, train_set, tuning_set)
```

```
## [1] "--- k = 1 ---"
## [1] 7792.749
## [1] "--- k = 2 ---"
## [1] 4886.91
## [1] "--- k = 3 ---"
## [1] 4679.723
## [1] "--- k = 4 ---"
## [1] 4672.831
## [1] "--- k = 5 ---"
## [1] "4687.4655868804 (Stop)"
```

```
repeat_bagged_rmse_out <- bagged_learn(best_tree_bagging$model, train_set, b=100) |>
  bagged_predict(test_set) |>
  rmse_oos(test_set$charges, preds = _)
print(paste("The RMSE_out for the Repeat Bagged decision tree is ",
            round(repeat_bagged_rmse_out, 2),
            ", where max depth = ", best_tree_bagging$k,
```

```
            sep = ""))
```

```
## [1] "The RMSE_out for the Repeat Bagged decision tree is 4748.13, where max depth = 4"
```

b. Repeat the boosting of the decision tree, using a base tree of maximum depth 1, 2, . . . n, keep training on the 70% training set while the $RMSE_{out}$ of your 20% set keeps dropping; stop when the $RMSE_{out}$ has started increasing again (show prediction error at each depth). Report the final $RMSE_{out}$ using the final 10% of the data as your test set.

```
# Prepared the function we'll need.
repeat_boosting <- function(model, train_set, tuning_set, maxdepth = 1){
  this_model <- update(model, control = rpart.control(maxdepth = 1))
  rmse_out <- boost_learn(this_model, train_set, outcome="charges", n=1000) |>
    boost_predict(tuning_set) |>
    rmse_oos(tuning_set$charges, preds = _)
  k <- 1
  print(paste("--- k =", k, "---"))
  print(rmse_out)
  while(TRUE){
    k <- k + 1
    print(paste("--- k =", k, "---"))
    control_text <- paste("update(this_model, control = rpart.control(maxdepth =",
                          k, "))")
    temp_model <- eval(parse(text = control_text))
    rmse_out_temp <- boost_learn(temp_model, train_set, outcome="charges", n=1000) |>
      boost_predict(tuning_set) |>
      rmse_oos(tuning_set$charges, preds = _)
    if (rmse_out_temp >= rmse_out){
      print(paste(rmse_out_temp, "(Stop)"))
      result = list(model = this_model, k = k-1)
      return(result)
    }
    rmse_out <- rmse_out_temp
    this_model <- temp_model
    print(rmse_out)
  }
}
```

```
best_tree_boosting <- repeat_boosting(insurance_full_rpart, train_set, tuning_set)
```

```
## [1] "--- k = 1 ---"
## [1] 6192.397
## [1] "--- k = 2 ---"
## [1] 4389.575
## [1] "--- k = 3 ---"
## [1] 4323.735
## [1] "--- k = 4 ---"
## [1] 4303.152
## [1] "--- k = 5 ---"
## [1] "4319.00604913415 (Stop)"
```

```
repeat_boost_rmse_out <-
  boost_learn(best_tree_boosting$model, train_set, outcome="charges", n=1000) |>
```

```
  boost_predict(test_set) |>
  rmse_oos(test_set$charges, preds = _)
print(paste("The RMSE_out for the Repeat Boosted decision tree is ",
            round(repeat_boost_rmse_out, 2),
            ", where max depth = ", best_tree_boosting$k, sep = ""))
```

```
## [1] "The RMSE_out for the Repeat Boosted decision tree is 4676.84, where max depth = 4"
```