BACS-hw17-107070004

helper: teams

This week, we will look at a dataset of US health insurance premium charges. We will build models that can predict what someone's insurance charges might be, given several factors about them. You download the dataset, and find more information about it, at the Kaggle platform where machine learning people like to host challenges and share datasets: https://www.kaggle.com/datasets/teertha/ushealthinsurancedataset

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 <- read.csv("./insurance.csv")
insurance_lm <- lm(charges ~ age + sex + bmi + children + smoker + region, data=insurance)
summary(insurance_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
```

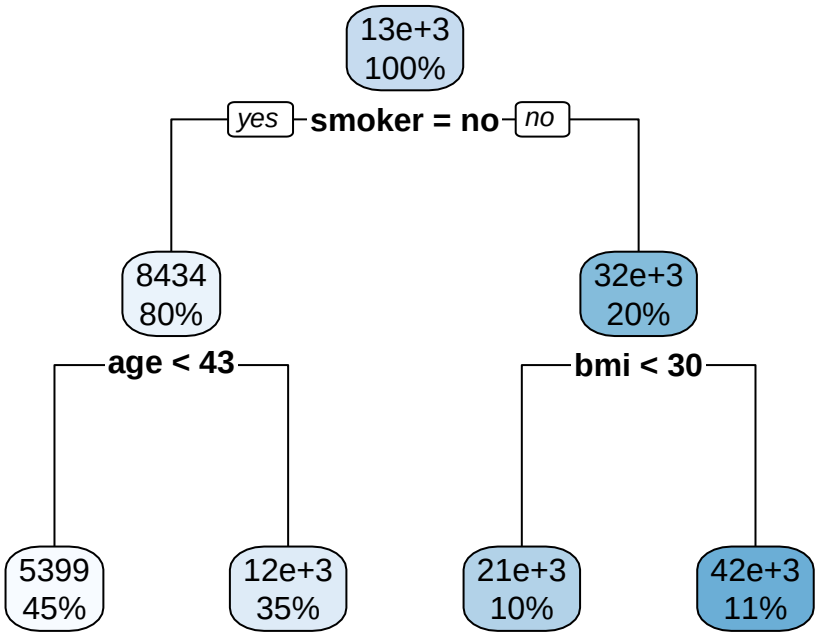
b. Create a decision (regression) tree with default parameters

(i) Plot a visual representation of the tree

```
insurance_tree <- rpart(charges ~ age + sex + bmi + children + smoker + region, data=insurance)
insurance_tree
```

```
## n= 1338
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
## 1) root 1338 196074200000 13270.420
##   2) smoker=no 1064  38188720000  8434.268
##     4) age< 42.5 596  13198540000  5398.850 *
##     5) age>=42.5 468  12505450000 12299.890 *
##   3) smoker=yes 274  36365600000 32050.230
##     6) bmi< 30.01 130   3286655000 21369.220 *
##     7) bmi>=30.01 144   4859010000 41692.810 *
```

```
rpart.plot(insurance_tree)
```



(ii) How deep is the tree (see nodes with "decisions" – ignore the leaves at the bottom)

2

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

4

(iv) What is the average charges of each leaf group?

(v) What conditions (decisions) describe each group?

table for the answers of (iv) & (v)

the condition of leaf group	average charges
smoker=yes, age<42.5=yes	5398.850
smoker=yes, age<42.5=no	12299.890
smoker=no, bmi<30.01=yes	21369.220
smoker=no, bmi<30.01=no	41692.810

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

```
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){
    shuffled_indicies <- sample(1:nrow(dataset))
    dataset <- dataset[shuffled_indicies,]
    fold_pred_errors <- sapply(1:k, function(kth) {
        fold_i_pe(kth, k, model, dataset, outcome)
      })
    pred_errors <- unlist(fold_pred_errors)
    mse <- function(errs){
      mean(errs^2)
    }
    c(is = mse(residuals(model)), oos = mse(pred_errors))
}
```

a. What is the RMSEoos for the OLS regression model?

```
sqrt(k_fold_mse(insurance_lm, insurance, "charges", 10)[2])
```

```
##      oos
## 6099.866
```

b. What is the RMSEoos for the decision tree model?

```
sqrt(k_fold_mse(insurance_tree, insurance, "charges", 10)[2])
```

```
##      oos
## 5124.771
```

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

a. Write 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 generously on Teams to get feedback, or ask others for help.

```
set.seed(27935752)
train_indices <- sample(1:nrow(insurance), size = 0.8*nrow(insurance))
train_set <- insurance[train_indices,]
test_set <- insurance[-train_indices,]

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

```
bagged_learn <- function(model, dataset, b=100){
    lapply(1:b,\(i) {
    boot_index <- sample(nrow(dataset), replace = TRUE)
    boot_dataset <- dataset[boot_index,]
      # Get a bootstrapped (resampled w/ replacement) dataset
      update(model, data = boot_dataset)
      # Return a retrained(updated) model
     })
}
bagged_predict <- function(bagged_learning, new_data) {
  b <- length(bagged_learning)
  predictions <- lapply(1:b,\(i) {
    pred <- predict(bagged_learning[[i]], new_data)
  })
    # get b predictions of new_data
    as.data.frame(predictions) |> apply(1, mean)
    # apply a mean over the columns of predictions
}
```

b. What is the RMSEoos for the bagged OLS regression?

```
bagged_list <- bagged_learn(insurance_lm, train_set)
bagged_predict_list <- bagged_predict(bagged_list, test_set)
mse_oos(test_set$charges, unlist(bagged_predict_list))
```

```
## [1] 6022.205
```

c. What is the RMSEoos for the bagged decision tree?

```
bagged_list <- bagged_learn(insurance_tree, train_set)
bagged_predict_list <- bagged_predict(bagged_list, test_set)
mse_oos(test_set$charges, unlist(bagged_predict_list))
```

```
## [1] 4907.184
```

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

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.

```
boosted_learn <- function(model, dataset, n=100, rate=0.1){
  predictors <- dataset[,1:6] # get data frame of only predictor variables
  res <-dataset[,7] # get vector of actuals to start

  models <-list()
  # Initialize residuals and models

  for(i in 1:n) {
    this_model<-update(model, data =cbind(charges=res, predictors))
    res <- res - (rate)*predict(this_model)
    # update residuals with learning rate
    models[[i]] <-this_model
    # Store model
  }
  list(models=models, rate=rate)
}
boosted_predict <- function(boosted_learning, new_data) {
  boosted_models <- boosted_learning$models
  rate <- boosted_learning$rate
  n <- length(boosted_learning$models)
  predictions <- lapply(1:n, \(i){
    rate*predict(boosted_models[[i]], new_data)
  })
  # get predictions of new_data from each model
  pred_frame <- as.data.frame(predictions) |> unname()
```

```
  apply(pred_frame, 1, sum)
}
```

b. What is the RMSEoos for the boosted OLS regression?

```
boosted_list <- boosted_learn(insurance_lm, train_set)
boosted_predict_list <- boosted_predict(boosted_list, test_set)
mse_oos(test_set$charges, unlist(boosted_predict_list))
```

```
## [1] 6020.292
```

c. What is the RMSEoos for the boosted decision tree?

```
boosted_list <- boosted_learn(insurance_tree, train_set)
boosted_predict_list <- boosted_predict(boosted_list, test_set)
mse_oos(test_set$charges, unlist(boosted_predict_list))
```

```
## [1] 4494.743
```

Question 4) 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. Report the RMSEoos at each step.

a. Repeat the bagging of the decision tree, using a base tree of maximum depth 1, 2, . . . n while the RMSEoos keeps dropping; stop when the RMSEoos has started increasing again.

```
bagged_lm_algo <- function(tree){
  bagged_list <- bagged_learn(tree, train_set)
  bagged_predict_list <- bagged_predict(bagged_list, test_set)
  mse_oos(test_set$charges, unlist(bagged_predict_list))
}

num <- 1
pre_oos <- 100000000
oos <- 100000000
while (pre_oos >= oos) {
  pre_oos <<- oos
  old_tree_stump <- rpart(charges ~ age + sex + bmi + children + smoker + region, data=insurance, cp=0,
  oos <<- bagged_lm_algo(old_tree_stump)
  cat(num, ":", oos, "\n")
  num <<- num + 1
}
```

```
## 1 : 7697.662
## 2 : 5071.054
## 3 : 4485.238
## 4 : 4449.888
## 5 : 4433.783
## 6 : 4456.964
```

b. Repeat the boosting of the decision tree, using a base tree of maximum depth 1, 2, . . . n while the RMSEoos keeps dropping; stop when the RMSEoos has started increasing again.

```
boosted_lm_algo <- function(tree){
  boosted_list <- boosted_learn(tree, train_set)
  boosted_predict_list <- boosted_predict(boosted_list, test_set)
  mse_oos(test_set$charges, unlist(boosted_predict_list))
}

num <- 1
pre_oos <- 100000000
oos <- 100000000
while (pre_oos >= oos) {
  pre_oos <<- oos
  old_tree_stump <- rpart(charges ~ age + sex + bmi + children + smoker + region, data=insurance, cp=0,
  oos <<- boosted_lm_algo(old_tree_stump)
  cat(num, ":", oos, "\n")
  num <<- num + 1
}
```

```
## 1 : 6027.826
## 2 : 4438.873
## 3 : 4429.897
## 4 : 4513.51
```