HW18

108048110

2022-06-07

BACS HW - Week 18

Prerequisite

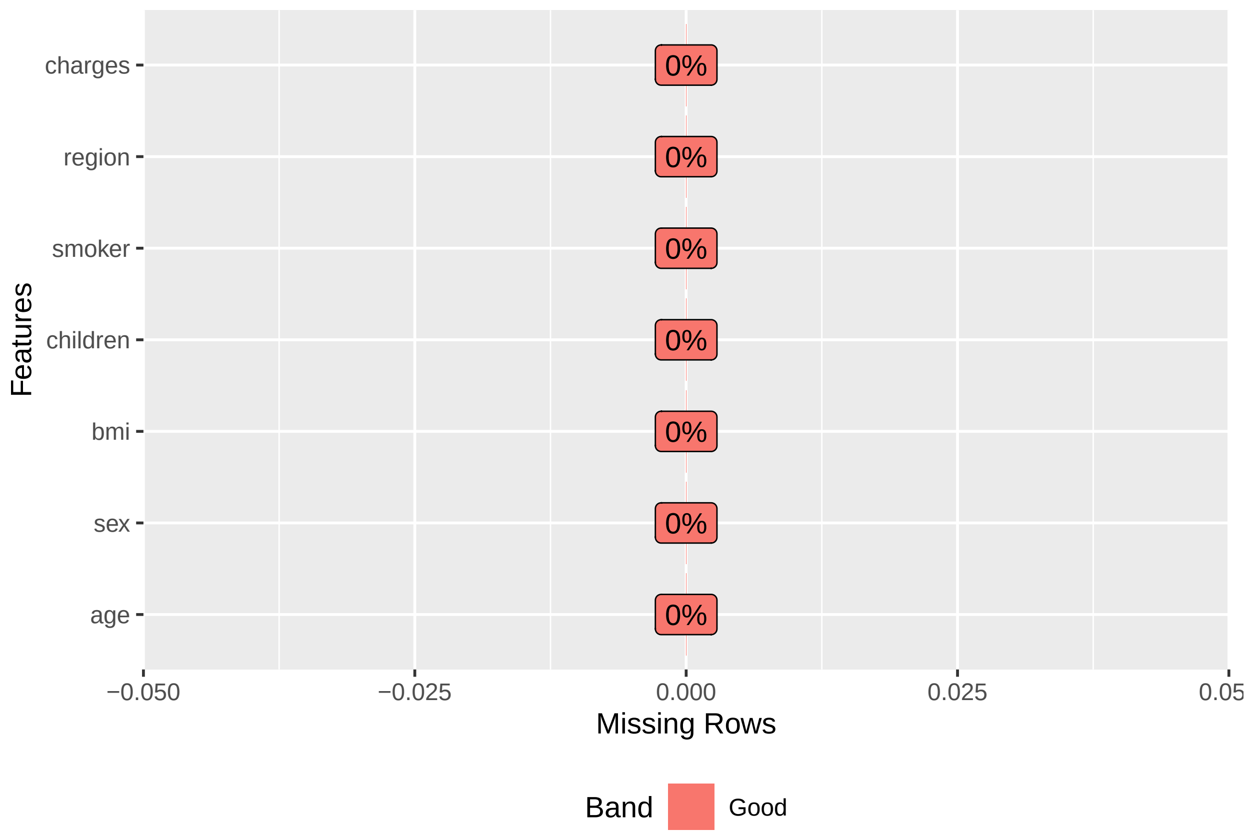
```
library(dplyr)
library(ggplot2)
library(DataExplorer)
library(rpart)
library(rpart.plot)
```

Setup

```
# loading data and remove missing values
insurance <- read.csv('data/insurance.csv')

as.matrix(lapply(insurance, \(x){sum(is.na(x))}))
```

```
##          [,1]
## age      0
## sex      0
## bmi      0
## children 0
## smoker   0
## region   0
## charges  0
```

```
plot_missing(insurance)
```




```r
# define rmse function
rmse_oos <- function(groud_truth, preds){
  sqrt(mean((groud_truth-preds)^2))
}
```

Question 1) Create Explanatory models

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

```r
# a. Ordinary Least Square regression
insurance %>% glimpse
```

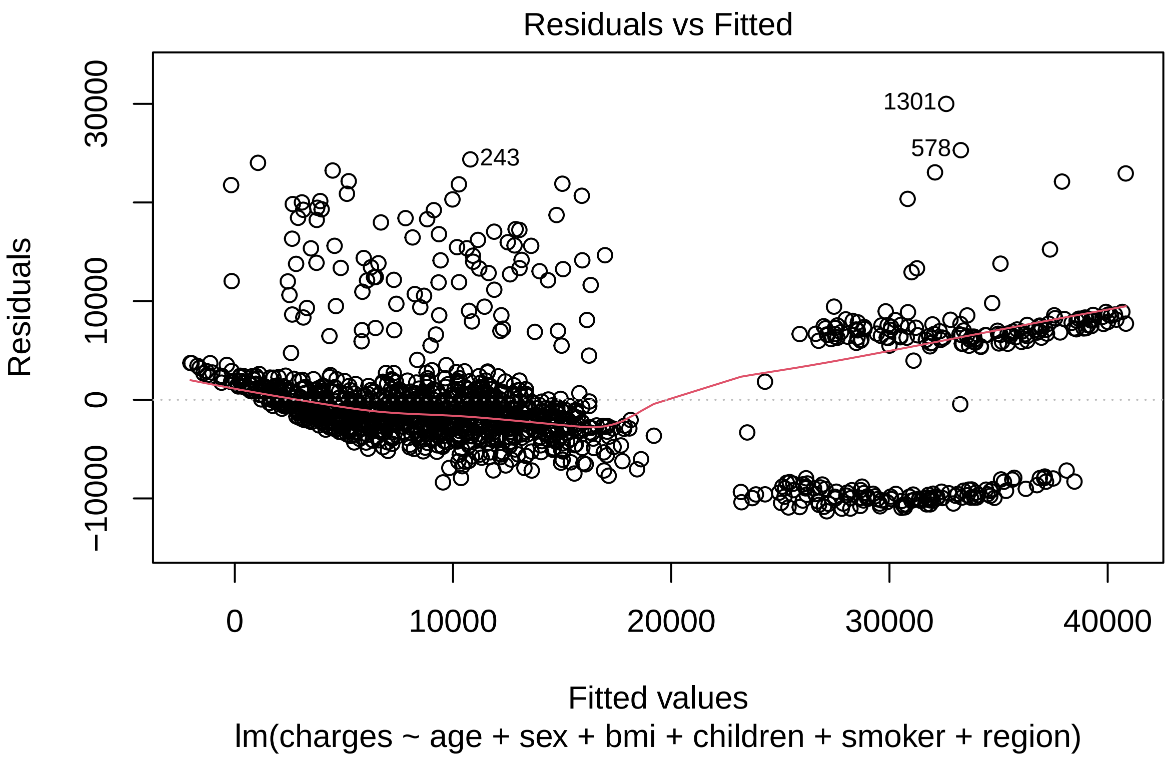
```
## Rows: 1,338
## Columns: 7
## $ age      <int> 19, 18, 28, 33, 32, 31, 46, 37, 37, 60, 25, 62, 23, 56, 27, 1~
## $ sex      <chr> "female", "male", "male", "male", "male", "female", "female",~
## $ bmi      <dbl> 27.900, 33.770, 33.000, 22.705, 28.880, 25.740, 33.440, 27.74~
## $ children <int> 0, 1, 3, 0, 0, 0, 1, 3, 2, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0~
## $ smoker   <chr> "yes", "no", "no", "no", "no", "no", "no", "no", "no", "no", ~
## $ region   <chr> "southwest", "southeast", "southeast", "northwest", "northwes~
## $ charges  <dbl> 16884.924, 1725.552, 4449.462, 21984.471, 3866.855, 3756.622,~
```

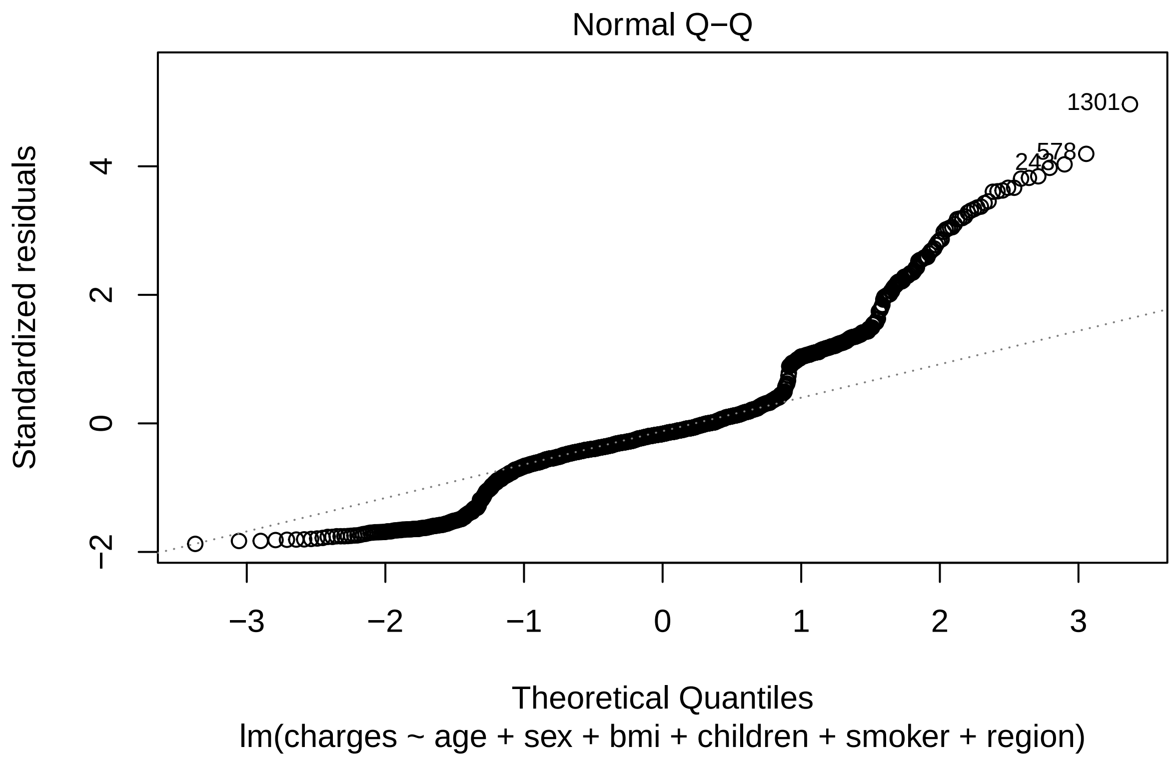
```r
insurance_lm <- lm(charges~age+
                     sex+
                     bmi+
                     children+
                     smoker+
                     region,
                   data = insurance)
summary(insurance_lm)
```

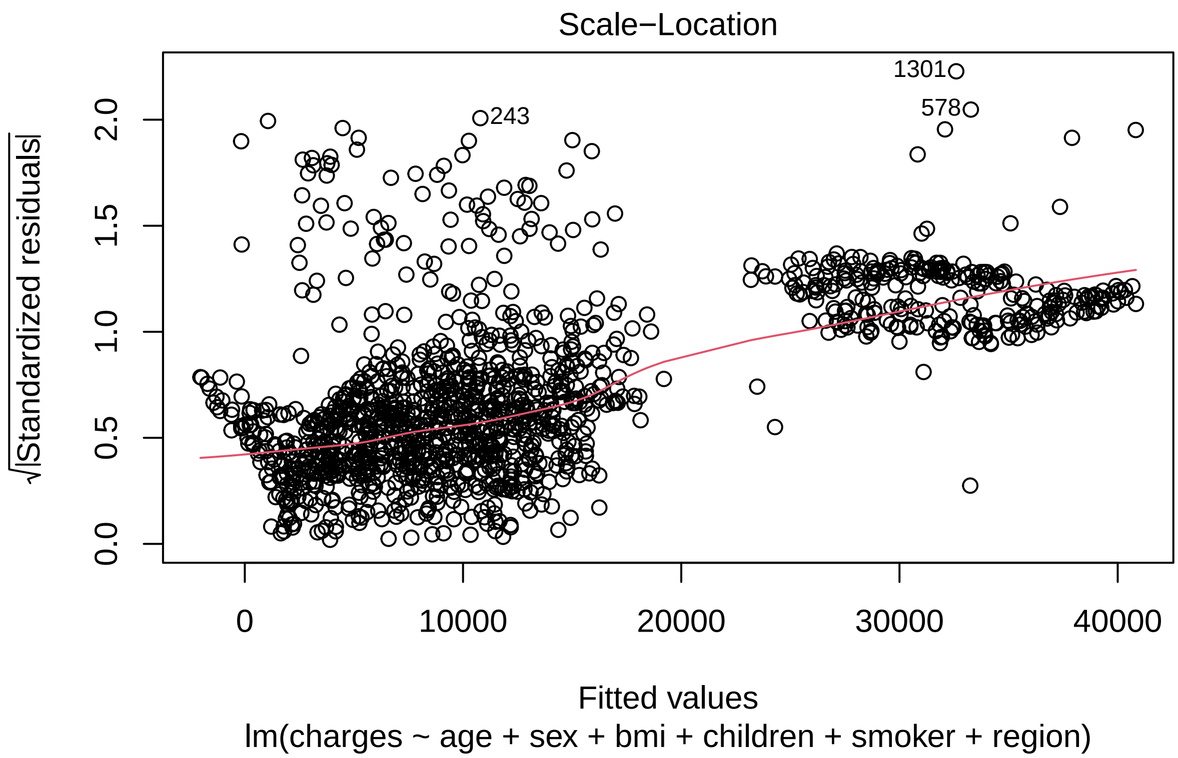
```
##
## Call:
## lm(formula = charges ~ age + sex + bmi + children + smoker +
##     region, data = insurance)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11304.9  -2848.1   -982.1   1393.9  29992.8
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (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
```

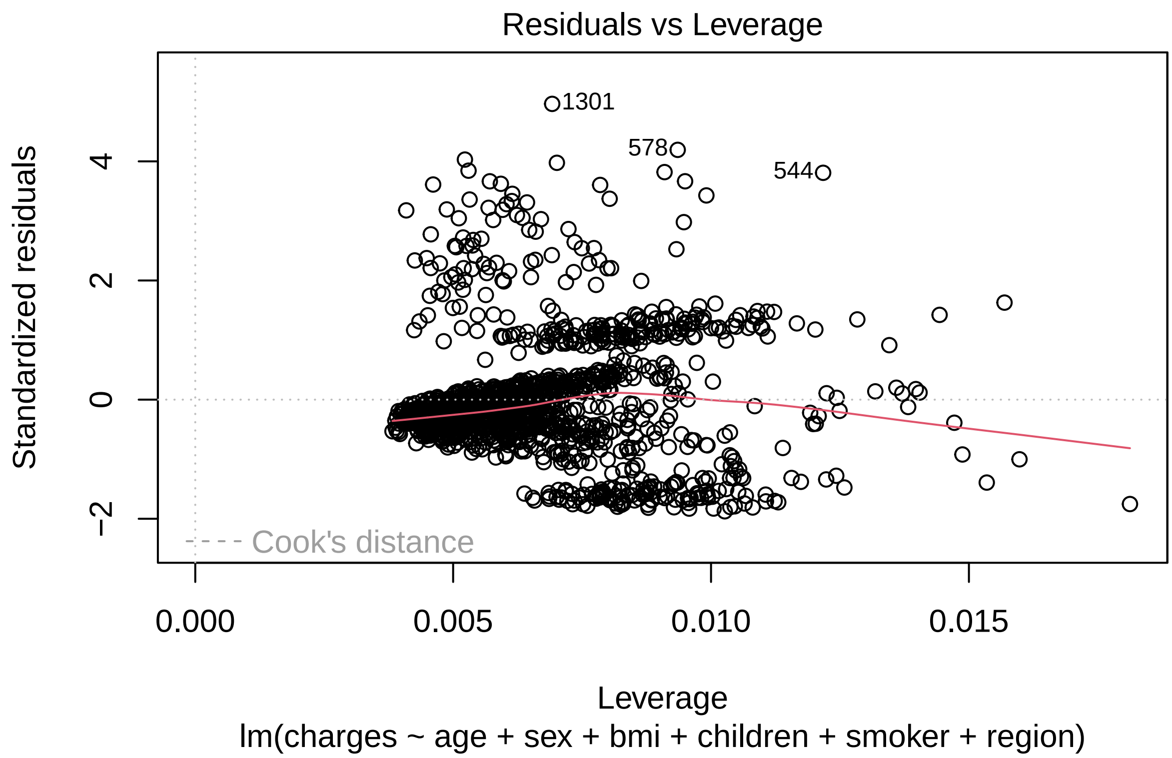
Ans. `Age`, `BMI`, `children`, and `smoker` appear to be highly correlated to charging.

```r
plot(insurance_lm)
```


Residuals vs Fitted
Residuals
Fitted values
lm(charges ~ age + sex + bmi + children + smoker + region)
30000
10000
0
−10000
0
10000
20000
30000
40000
1301
578
243


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(charges ~ age + sex + bmi + children + smoker + region)
1301
578
243
−2
0
2
4
−3
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
0
10000
20000
30000
40000
1301
578
243
Fitted values
lm(charges ~ age + sex + bmi + children + smoker + region)

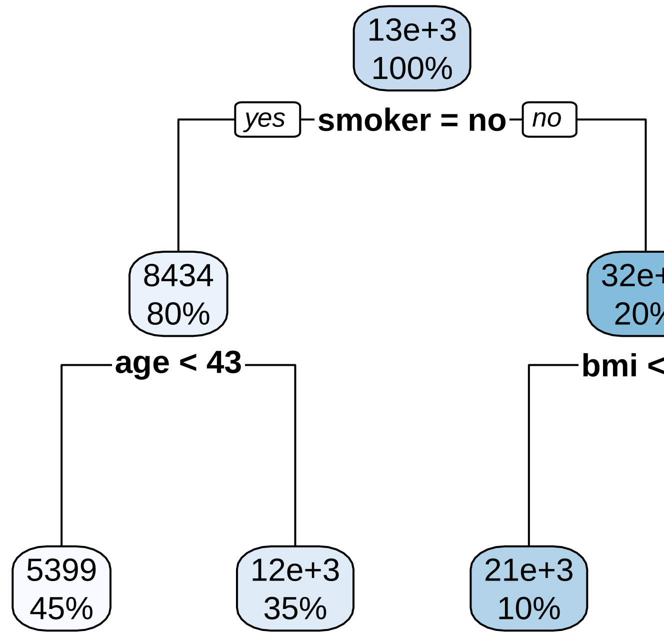


---

b. Create a decision tree.

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

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



i. **Plot a visual representation of the tree.**

ii. **How deep is the tree.** **Ans.** Depth=3.

```
insurance_tree$frame$var
```

iii. **How many leaf groups does it suggest to bin the data into?**

```
## [1] "smoker" "age"    "<leaf>" "<leaf>" "bmi"    "<leaf>" "<leaf>"
```

Ans. 4 leaf groups.

```
insurance_tree %>% summary
```

iv. **What is the average charges of each leaf group?**

```
## Call:
## rpart(formula = charges ~ age + sex + bmi + children + smoker +
##     region, data = insurance)
##   n= 1338
```

```
##
##          CP nsplit rel error    xerror       xstd
## 1 0.6197648      0 1.0000000 1.0024155 0.05197376
## 2 0.1439247      1 0.3802352 0.3824550 0.01903235
## 3 0.0636735      2 0.2363104 0.2398627 0.01450672
## 4 0.0100000      3 0.1726369 0.1810231 0.01349367
##
## Variable importance
## smoker    bmi    age region    sex
##     71     17      8      3      1
##
## Node number 1: 1338 observations,    complexity param=0.6197648
##   mean=13270.42, MSE=1.465428e+08
##   left son=2 (1064 obs) right son=3 (274 obs)
##   Primary splits:
##       smoker   splits as  LR,          improve=0.61976480, (0 missing)
##       age      < 42.5    to the left,  improve=0.07793137, (0 missing)
##       bmi      < 30.17   to the left,  improve=0.04212369, (0 missing)
##       children < 1.5     to the left,  improve=0.00831500, (0 missing)
##       region   splits as  LLRL,        improve=0.00547327, (0 missing)
##
## Node number 2: 1064 observations,    complexity param=0.0636735
##   mean=8434.268, MSE=3.589166e+07
##   left son=4 (596 obs) right son=5 (468 obs)
##   Primary splits:
##       age      < 42.5    to the left,  improve=0.326922000, (0 missing)
##       children < 1.5     to the left,  improve=0.019154040, (0 missing)
##       bmi      < 20.955  to the left,  improve=0.012695030, (0 missing)
##       region   splits as  RRLL,        improve=0.004790031, (0 missing)
##       sex      splits as  RL,          improve=0.003171961, (0 missing)
##   Surrogate splits:
##       bmi < 35.6325 to the left,  agree=0.58, adj=0.045, (0 split)
##
## Node number 3: 274 observations,    complexity param=0.1439247
##   mean=32050.23, MSE=1.327212e+08
##   left son=6 (130 obs) right son=7 (144 obs)
##   Primary splits:
##       bmi      < 30.01   to the left,  improve=0.776006300, (0 missing)
##       age      < 43.5    to the left,  improve=0.126767300, (0 missing)
##       region   splits as  LLRL,        improve=0.029264480, (0 missing)
##       sex      splits as  LR,          improve=0.010246720, (0 missing)
##       children < 1.5     to the left,  improve=0.003975908, (0 missing)
##   Surrogate splits:
##       region   splits as  LLRR,        agree=0.602, adj=0.162, (0 split)
##       sex      splits as  LR,          agree=0.566, adj=0.085, (0 split)
##       age      < 21.5    to the left,  agree=0.536, adj=0.023, (0 split)
##       children < 2.5     to the right, agree=0.536, adj=0.023, (0 split)
##
## Node number 4: 596 observations
##   mean=5398.85, MSE=2.214521e+07
##
## Node number 5: 468 observations
##   mean=12299.89, MSE=2.672104e+07
##
```

```
## Node number 6: 130 observations
##   mean=21369.22, MSE=2.528196e+07
##
## Node number 7: 144 observations
##   mean=41692.81, MSE=3.374313e+07
```

Ans. 5398.85, 12299.89, 21369.22, 441692.81

v. **What conditions (decisions) describe each group?** **Ans.** `smoker`==no, `age`<43, `bmi`<30.

Question 2) Use LOOCV to see how how our models perform predictively.

```r
fold_i_pred_err <- function(i, k, dataset, model){
  # cut the data set
  folds <- cut(1:nrow(dataset), k, labels=FALSE)

  # pick data that is labeled as "i"
  test_indices <- which(folds==i)

  test_set <- dataset[test_indices, ]
  train_set <- dataset[-test_indices, ]

  # trained model
  trained_model <- update(model, data=train_set)

  predictions <- predict(trained_model, test_set)
  test_set[,length(test_set)]-predictions
}


# calculates mse_oos across all folds
k_fold_rmse <- function(model, data, k=10){

  # randomly shuffle the data
  shuffled_indices = sample(1:nrow(data))
  data = data[shuffled_indices,]

  # get prediction errors of each folds
  fold_pred_error <- sapply(1:k, \(i){
    fold_i_pred_err(i, k, data, model)
  })

  pred_error <- unlist(fold_pred_error)

  rmse <- \(errs){sqrt(mean(errs^2))}

  c(in_sample = rmse(residuals(model)), out_of_sample = rmse(pred_error))
}
```

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

```
k_fold_rmse(insurance_lm, insurance, k=nrow(insurance))
```

```
##     in_sample out_of_sample
##      6041.680      6087.388
```

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

```
k_fold_rmse(insurance_tree, insurance, k=nrow(insurance))
```

```
##     in_sample out_of_sample
##      5029.781      5135.175
```

Bagging and Boosting

- **Note.** For bagging and boosting, we will partition the data to create training and test sets using an 80:20 split-sample testing to save time.

```
train_indicies = sample(1:nrow(insurance),
                        size=0.8*nrow(insurance))
train_set = insurance[train_indicies,]
test_set = insurance[-train_indicies,]
```

Question 3)

- Bagging

a. Write `bagged` functions

to train and predict on the data.

```
bagged_train <- function(model, dataset, b=250){
  lapply(1:b, \(i){
    # get a bootstrapped data set
    train_set <- dataset[sample(1:nrow(dataset), nrow(dataset), replace=TRUE),]
    train_models = update(model, data=train_set)
  })
}
```

```
bagged_predict <- function(bagged_model, new_data){
  predictions <- lapply(1:length(bagged_model), \(i){
    predict(bagged_model[[i]], new_data)
  })
```

```r
  # take the mean of 100 predictions on rows of mpg
  as.data.frame(predictions) |> apply(1, FUN=mean)
}
```

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

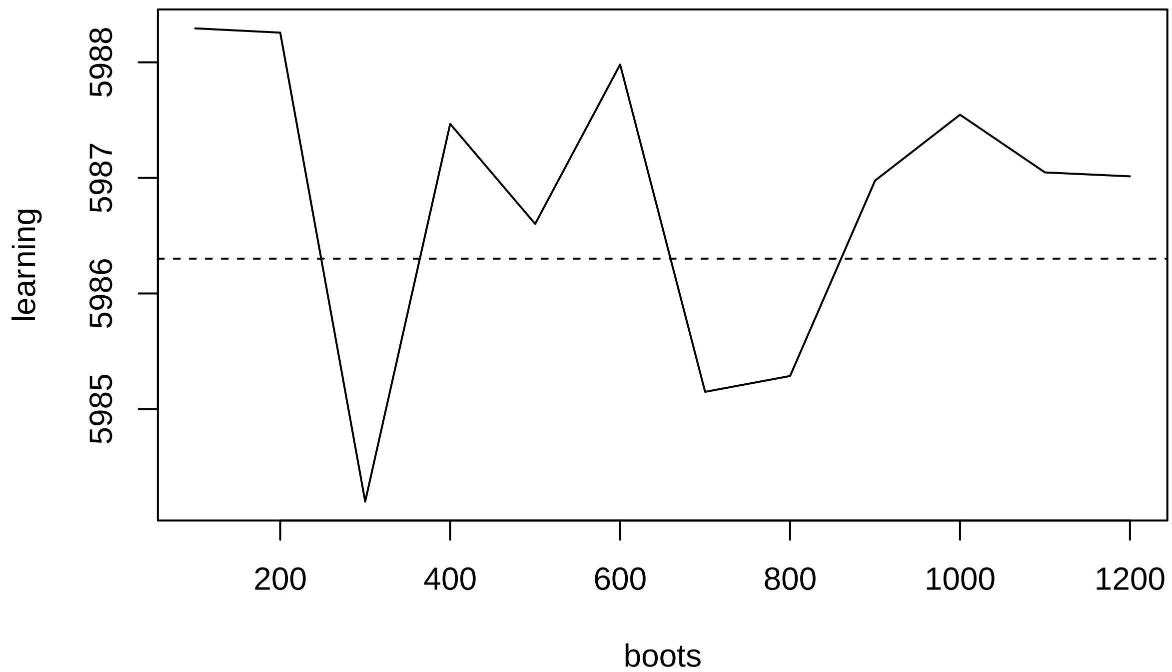
```r
old_learning <- update(insurance_lm, data=train_set) |>
  predict(object=_, test_set) |>
  rmse_oos(test_set$charges, preds=_)

print(old_learning)
```

```
## [1] 5986.301
```

```r
set.seed(1)
# bagged
bagged_train(insurance_lm, train_set, b=100) |>
  bagged_predict(test_set) |>
  rmse_oos(test_set$charges, preds=_)
```

```
## [1] 5986.005
```

```r
boots <- seq(100, 1200, by=100)
learning <- sapply(boots, \(b){
  bagged_train(insurance_lm, train_set, b=b) |>
    bagged_predict(test_set) |>
    rmse_oos(test_set$charges, preds=_)
})

plot(boots, learning, type="l")
abline(h=old_learning, lty='dashed')
```



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

```
old_learning <-
  predict(insurance_tree, test_set) |>
  rmse_oos(test_set$charges, preds=_)

print(old_learning)
```

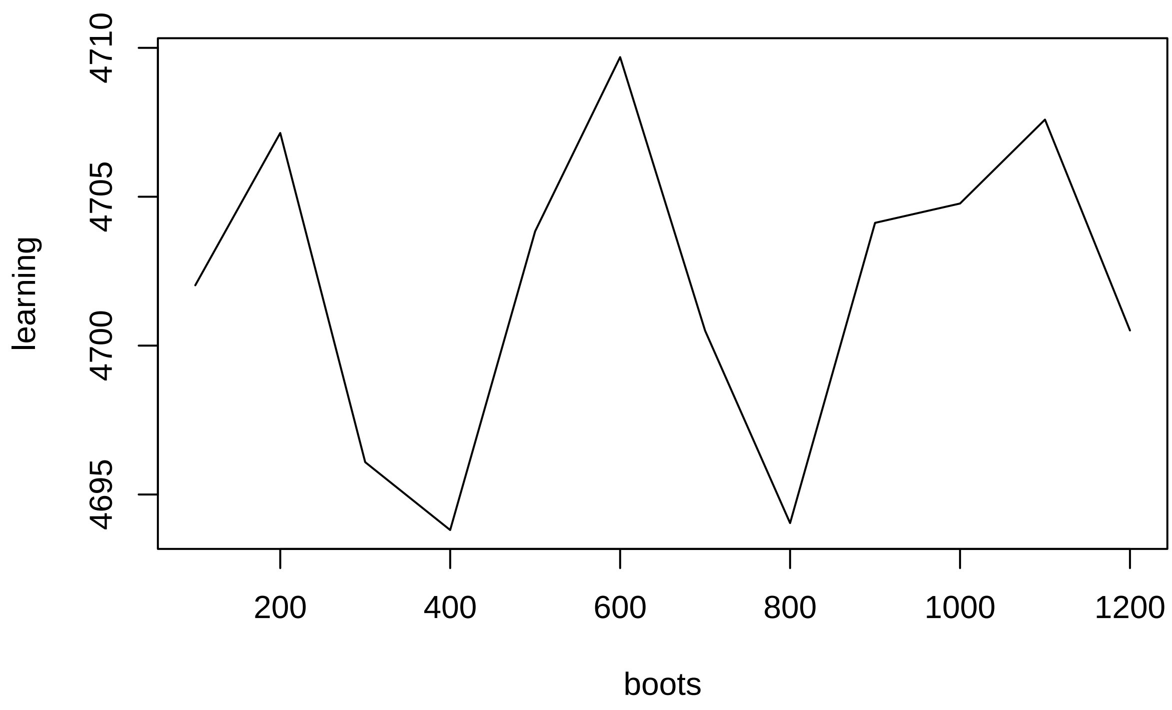
```
## [1] 4829.817
```

```
bagged_train(insurance_tree, train_set, b=100) |>
  bagged_predict(test_set) |>
  rmse_oos(test_set$charges, preds=_)
```

```
## [1] 4732.701
```

```
boots <- seq(100, 1200, by=100)
learning <- sapply(boots, \(b){
  bagged_train(insurance_tree, train_set, b=b) |>
    bagged_predict(test_set) |>
    rmse_oos(test_set$charges, preds=_)
})
```

```r
plot(boots, learning, type="l")
abline(h=old_learning, lty='dashed')
```



- Boosting

a. Write `boosted` functions to train and predict on the data.

```r
boost_train <- function(model, dataset, target, n=100, lr=0.1){

  # get target column index
  target_index = which(colnames(dataset)==target)

  # get data.frame of only predictor variable
  predictors <- dataset[, -target_index]

  # Initialize residuals and models
  res <- dataset[, target_index] # get vector of GT to start

  models <- list()

  for(i in 1:n){

    # fit predictor and residuals
```

```r
    new_model <- update(model, data = cbind(charges=res, predictors))

    # update residuals with lr: e = e - a * y_hat
    res <- res-sapply(predict(new_model, dataset), \(i){lr*i})


    models[[i]] <- new_model
  }

  list(models=models, lr=lr)
}
```

```r
boost_predict <- function(boosted_training, new_data){
  boosted_model = boosted_training$models
  boosted_lr = boosted_training$lr

  # predict target for models and store the predictions
  predictions = lapply(boosted_model, \(model){predict(model, new_data)})

  predictions_frame = as.data.frame(predictions) |> unname()

  apply(predictions_frame, 1, sum)*boosted_lr
}
```

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

```r
boost_train(insurance_lm, train_set, target="charges", n=1000) |>
  boost_predict(test_set) |>
  rmse_oos(test_set$charges, preds = _)
```

```
## [1] 5986.301
```

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

```r
boost_train(insurance_tree, train_set, target="charges", n=1000) |>
  boost_predict(test_set) |>
  rmse_oos(test_set$charges, preds = _)
```

```
## [1] 4408.916
```

Question 4) Repeat the `bagging` and `boosting` decision tree several times to see what kind of base tree helps us learn the fastest.

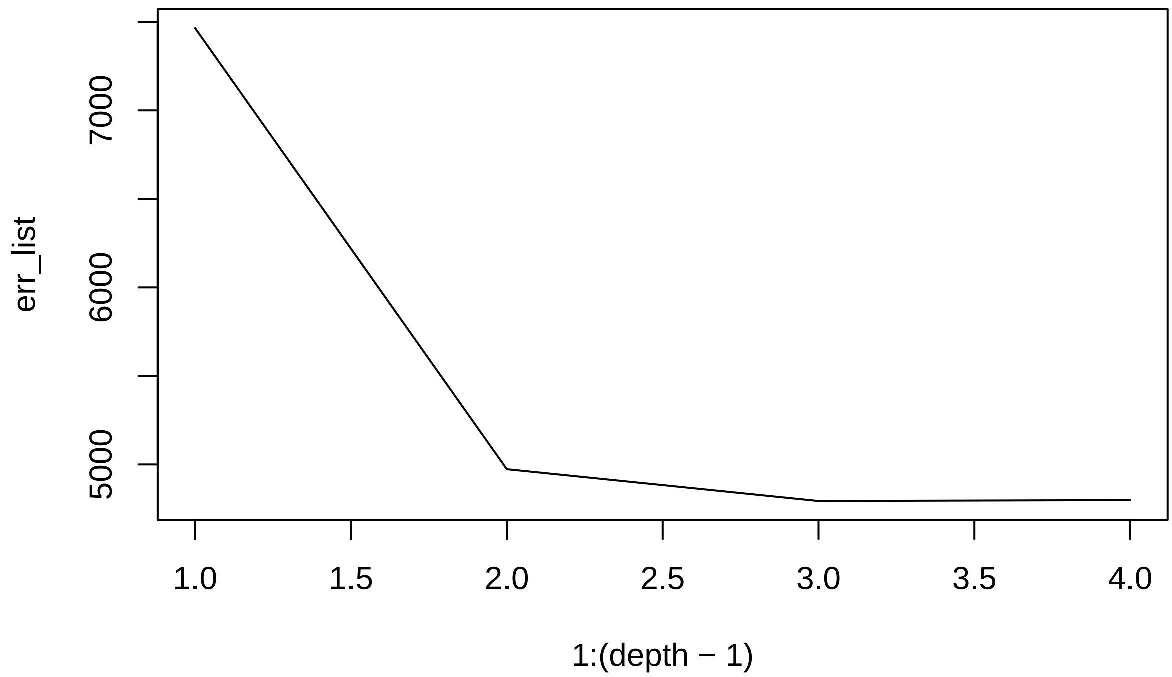
Note. Report the $RMSE_{oos}$ at each step.

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

```
prev_err=100000000
pred_err=10000000
err_list <- c()
depth=1
while(pred_err<prev_err){

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

  if(pred_err < prev_err){prev_err = pred_err}

  pred_err = bagged_train(insurance_tree, insurance) |>
    bagged_predict(insurance) |>
    rmse_oos(insurance$charges, preds = _)

  err_list <- c(err_list, pred_err)
  depth = depth+1
}

length(err_list)
```

```
## [1] 4
```

```
plot(1:(depth-1), err_list, type="l")
```




```
paste0("Max depth = ", length(err_list)-1)
```

```
## [1] "Max depth = 3"
```

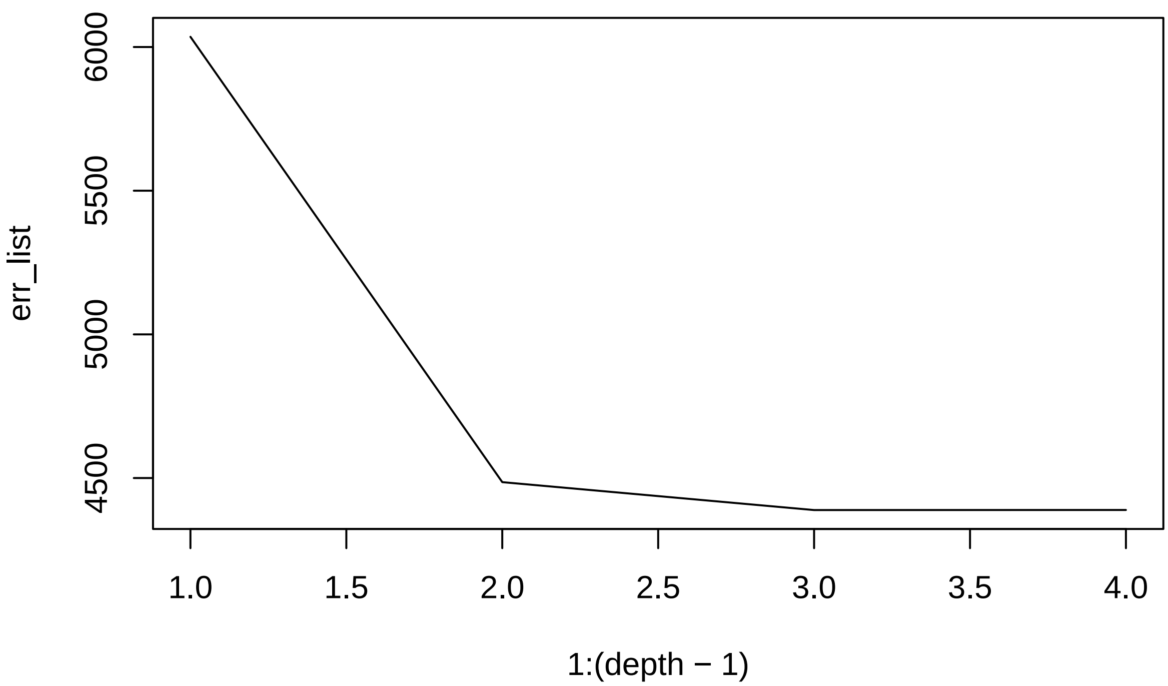
b. Repeat the `boosting` of the decision tree, using a base tree of maximum depth 1, 2, … n while the $RMSE_{oos}$ keeps dropping; stop when the $RMSE_{oos}$ has started increasing again.

```
prev_err=100000000
pred_err=10000000
err_list <- c()
depth=1
while(pred_err<prev_err){

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

  if(pred_err < prev_err){prev_err = pred_err}
```

```r
  pred_err = boost_train(insurance_tree, insurance, target="charges") |>
    boost_predict(insurance) |>
    rmse_oos(insurance$charges, preds = _)

  err_list <- c(err_list, pred_err)
  depth = depth+1
}

length(err_list)
```

```
## [1] 4
```

```r
plot(1:(depth-1), err_list, type="l")
```



Ans. Max depth: 3