HW17_Essemble

110078509

20220612

Setup: Download the data, load it in your script, and omit any rows with missing values (NAs)

```
df <- read.csv('insurance.csv', header = T)
insurance <- na.omit(df)
str(insurance)
```


```
df <- read.csv('insurance.csv', header = T)
insurance <- na.omit(df)
str(insurance)
```


```
## 'data.frame':    1338 obs. of  7 variables:
##  $ age     : int  19 18 28 33 32 31 46 37 37 60 ...
##  $ sex     : chr  "female" "male" "male" "male" ...
##  $ bmi     : num  27.9 33.8 33 22.7 28.9 ...
##  $ children: int  0 1 3 0 0 0 1 3 2 0 ...
##  $ smoker  : chr  "yes" "no" "no" "no" ...
##  $ region  : chr  "southwest" "southeast" "southeast" "northwest" ...
##  $ charges : num  16885 1726 4449 21984 3867 ...
```

```
k_fold_mse <- function(model, dataset, outcome, k=10) {
  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 <- \(errs) mean(errs^2)
  c(is = mse(residuals(model)), oos = mse(pred_errors))
}

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
```

(See Next Page)

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

```
ins_lm <- lm(charges ~ ., data=insurance)
summary(ins_lm)
```

```
## Call:
## lm(formula = charges ~ ., 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, childern, bmi, smokeyes, region southeast and southwest are significant. Meanwhile the rest are not which can be dropped for model improvement.

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

```
tree_model <- rpart(charges ~ ., data = insurance )

tree_model
```

```
## 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 *
```

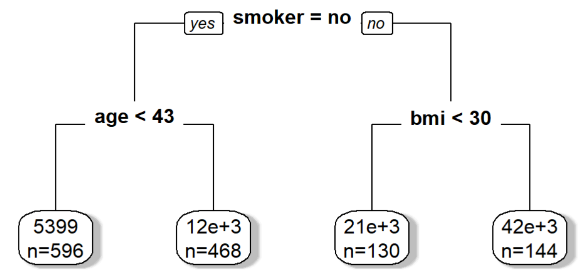
(See Next Page)

i. Plot a visual representation of the tree

Here, I only show the plot of Method 2.

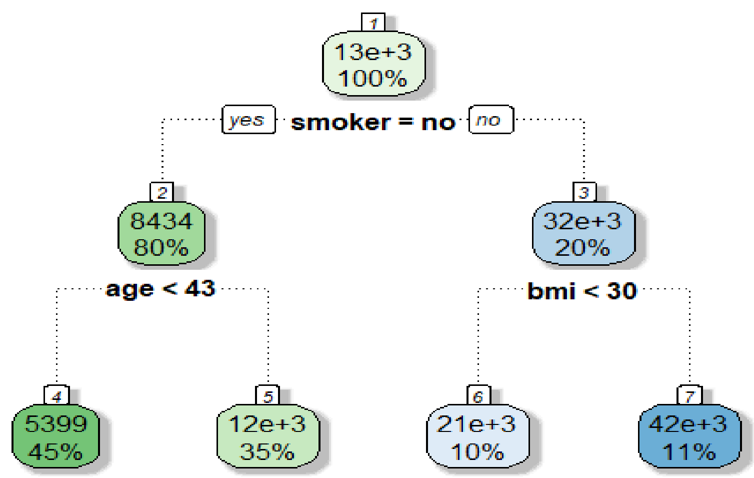
However, welcome to apply my code if you're interested.

```
# Method 1
prp(tree_model, # Model
    # No abbreviation
     faclen=0,
     # vertical branches
    fallen.leaves=TRUE,
    # Leaf's shadow
     shadow.col="gray",
# number of correct classifications / number of observations in that node
     extra = 1)
```

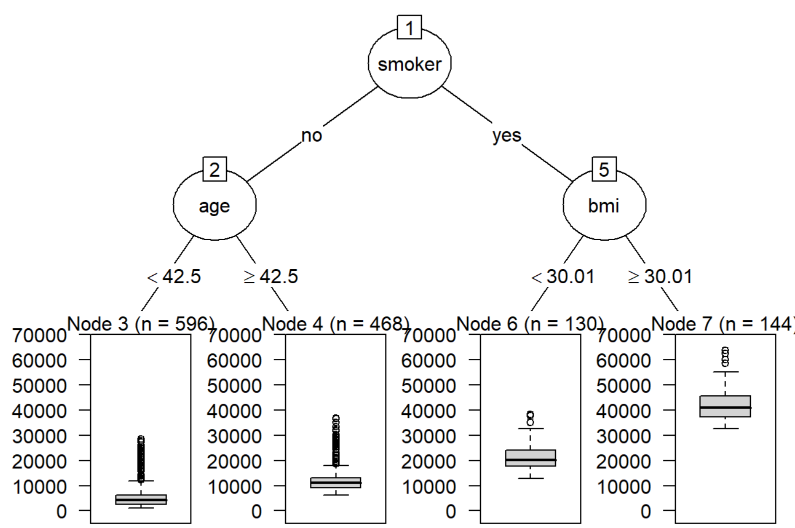



```
# Method 2
rpart.plot(tree_model,
          # color scheme
           box.palette = "GnBu",

          # dotted branch lines
           branch.lty = 3,
          # node boxes shadows
           shadow.col = "gray",
          #  show the node numbers
          nn = TRUE)
```




```
# Method 3
# plot(as.party(tree_model))
```



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

- Ans: 2 level after ingore the leafs

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

- Ans: 4 groups of leaf

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

Plz refer to the plot of ,ethod 2, and scroll down til the part of Node number 4-7

```
summary(tree_model)
## …
## 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
```

v. What conditions (decisions) describe each group?

Ans:

1. Whether is a smoker or not.
2. BMI < 30 or not
3. Age < 43 or not

Question 2) Let’s use LOOCV to se how how our models perform predictively

- Split-sample Testing

```
set.seed(22)
train.index <- sample(x=1:nrow(insurance), size=ceiling(0.8*nrow(insurance) ))
train <- insurance[train.index, ]
test <- insurance[-train.index, ]
charges_actual <- test$charges
```

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

```
# Train
OLS_model <- lm(charges ~., data = insurance )
lm_trained <- update(OLS_model, data=train)
# Predict
preds <- predict(lm_trained, test)
```

```r
# Test
RMSEoos <- function(actuals, preds) {
  (mean( (actuals - preds)^2 ))^0.5}
```

```r
# one sample RMSE
RMSE_OLS <- RMSEoos(charges_actual, preds)
cat('one sample RMSE:',RMSE_OLS, '\n\n')
## one sample RMSE: 6222.852
```

```r
# K-fold (LOOCV) RMSE
MSE <- k_fold_mse(lm_trained, insurance, "charges", k = 1000)
RMSE <- MSE^0.5

cat('K-fold (LOOCV) RMSE:\n is        oos\n',RMSE, '\n')
## K-fold (LOOCV) RMSE:
##  is        oos
##  6002.415 6086.806
```

Question 3) How 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.

The function I designed can calculate the bagging model of the Random Forest, regression (OLS), and Decision Tree. (See Next Page)

```r
bagged_learn <- function(model, dataset, model_type, b=100 , drop_out = 0.4) {
    lapply(1:b, \(i) {
        Index <- sample(x=1:nrow(dataset), size=ceiling(0.8*nrow(dataset)), replace
= T)
        Train=dataset[Index,]

        if (model_type == "OLS" ){
          Model <- update(model, data= Train)
        }

        else if (model_type == "RF" ){
          # Need rpart tree based model as input
          # it only works as dependent variable is continuous variable
          Index_row=sample(nrow(Train),round(nrow(Train)* (1-drop_out)))
          Train<-Train[Index_row,]
          Model <- update(model, data= Train)
        }
        else if (model_type == "DT" ){
          #Model<-rpart(model,data=Train ,method='anova')
          Model <- update(model, data= Train)
        }
        else{
          print("Warming: model type only allow OLS(Linear LM), RF (RandomForest), D
T (DecisionTree)")
          break
        }})}
bagged_predict <- function(bagged_models, new_data) {
    predictions <- lapply(bagged_models, \(x) predict(x, new_data))
    as.data.frame(predictions) |> apply(1, mean)}
```

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

- Bagged_learn for regression model

```
model_list <- bagged_learn(OLS_model,data = insurance, b =100, model_type = "OLS")

bagged_pred <- unlist(bagged_predict(model_list, test))

preds <- predict(tree_model, test)
RMSoos_OLS <- RMSEoos(charges_actual, bagged_pred)

RMSoos_OLS ## [1] 6175.428
```

Ans: The RMSEoos for the bagged OLS regression is `6175.428`

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

- Bagged_learn for Decision Tree Model

```
model_list <- bagged_learn(tree_model,data = insurance, b =100, model_type = 'DT')
bagged_pred <- unlist(bagged_predict(model_list, test))
RMSoos_Tree <- RMSEoos(charges_actual, bagged_pred)
RMSoos_Treem ## [1] 4815.196
```

Ans: The RMSEoos for the bagged decision tree is `4815.196`

Question 3) How 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.

```
boost_learn <- function(model, dataset, outcome, n=100, rate=0.1) {
    # get data frame of only predictor variable
    predictors <- dataset[, colnames(dataset)!= outcome]
    res <- dataset[, colnames(dataset) == outcome]
    models <- list()

    for (i in 1:n) {
        this_model <- update(model, data = cbind(charges=res, predictors))
        # update residual with learning rate
        res <- res - rate*predict(this_model)
        # model storage
        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
  # get predictions of new_data from each model
  predictions <- lapply(1:length(boosted_models), \(i)
    rate*predict((boosted_models[[i]]), new_data))
  pred_frame <- as.data.frame(predictions) |> unname()
  # apply a sum over the columns of predictions, weighted by learning rate
```

(See Next Page)

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

```
set.seed(110078509)
model <- boost_learn(lm_trained,train,"charges", n=100, rate=0.1)

pred <- model|> boost_predict(test)
RMSEoos(charges_actual , pred )
```

```
## [1] 6222.843
```

The RMSEoos for the boosted OLS regression is 6222.843

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

```
boost_prediction <- boost_learn(tree_model,train,"charges",  n=100, rate=0.1) |>
boost_predict(test)

RMSEoos(charges_actual , boost_prediction )
```

```
## [1] 4677.864
```

The RMSEoos for the boosted decision tree is 4677.864

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.

```
ls <- c()
for (i in 1:5){
old_tree_stump <- rpart(charges~., train, cp = 0, control =list( maxdepth = i)
set.seed(110078509)
preds <- bagged_learn(model = old_tree_stump,dataset =  train,model_type = 'DT' , b
= 100) |> bagged_predict(test)
RMSE_this <- RMSEoos(charges_actual, preds)

ls <- append(ls,RMSE_this)
}
names <- c(multi_items(" Depth-", 1:5))
cat(names, '\n' ,ls )
```

```
##  Depth-1  Depth-2  Depth-3  Depth-4  Depth-5
##  8022.451 5080.104 4632.197 4554.376 4588.228
```

```
Ans: Should stop as the depth of the tree equal to 4
```

(See Next Page)

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.

```
ls <- c()
for (i in 1:5){
    old_tree_stump <- rpart(charges~., train, cp = 0,  control =list( maxdepth = i))
    set.seed(110078509)
    preds_i <- boost_learn(old_tree_stump, train, 'charges', n=100, rate=0.1) |>
    boost_predict(test)
    RMSE_this <- RMSEoos(charges_actual, preds_i)
    ls <- append(ls,RMSE_this)
    }
names <- c(multi_items(" Depth-", 1:5))
cat(names, '\n' ,ls )
```

```
##  Depth-1  Depth-2  Depth-3  Depth-4  Depth-5
##  6207.541 4575.485 4646.622 4750.675 4852.06
```

```
Ans: Should stop as the depth of the tree equal to 2
```