(https://lms.monash.edu/mod/assign/view.php?id=7560449)

FIT5197 Assignment 3 Semester 2, 2020 (https://lms.monash.edu/mod/assign/view.php?id=7560449)

Authors: Dan Nguyen, Yun Zhao

Admins (Competition): Dr. Levin Kuhlmann, Yun Zhao, Anil Gurbuz

Proofreaders: Dr. Levin Kuhlmann, Yun Zhao, and other tutors

Date: Oct 2020

(https://lms.monash.edu/mod/assign/view.php?id=7560449)

Assignment Instruction (https://lms.monash.edu/mod/assign/view.php?id=7560449)

Please read through the instructions carefully, by submitting the assignment, you are considered to have read all the instructions carefully and be aware of the penalties that entail.

Part 1: Regression (50 Marks)

This part is about regression. Specifically, you will be `predicting the fuel efficiency` of a car (in kilometers per litre) based on its characteristics. This is a practical problem as Australia is one of the largest automobile markets in the world; thus, correctly predicting the fuel efficiency is necessary to control emission rates to the environment.

The dataset has many observations and predictors obtained from many retailers for car models available for sale from 2017 to 2020. The target variable is the fuel efficiency of the car measured in kilometers per litre. The higher this value, the better the fuel efficiency of the car.

PleaseProvide working/R code/justifications for each of these questions as required.

Note: If not explicitly mentioned, libraries are not allowed

In [1]:

```
# Read the data from students' side
remove(list = ls())
train <- read.csv("RegressionTrain.csv")
test <- read.csv("RegressionTest.csv")
```

In [ ]:

```
# PLEASE DO NOT ALTER THIS CODE BLOCK
# Please skip (don't run) this if you are a student
# Read in the data from marking tutors' side (ensure no cheating!)
remove(list = ls())
train <- read.csv("../data/RegressionTrain.csv")
test <- read.csv("../data/RegressionTest.csv")
label <- read.csv("../data/RegressionTestLabel.csv")
```

Question 1 (5 Marks)

Fit a **multiple linear model** to the fuel efficiency data using the `train` dataset. By checking the summary information, which predictors/variables do you think are possibly associated with fuel efficiency (use `0.05` significant level), and why? Which `three predictors/variables` appear to be the strongest predictors of fuel efficiency, and why?

Note: You don't have to worry about categorical variables here since R can deal with this automatically, focus your efforts on interpretation. Additionally, when explaining why features are strongly associated with the target, please refrain giving one or two sentences answers, these answers are not descriptive enough and will result in deduction of marks. Finally, please name the model here `lm.fit` for future marking purposes.

YOUR ANSWER HERE

In [2]:

```
# Fit a Multiple Linear regression model
# Target variable : Comb.FE
# Predictors : all the columns
lm.fit <- lm(Comb.FE ~ . , data = train)
```

In [3]:

```
# show the key statictics of the Linear model with respect to the predictors
summary(lm.fit)
```

```
Call:
lm(formula = Comb.FE ~ ., data = train)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.0256 -0.9978 -0.0644  0.7006 11.3941 

Coefficients:
                           Estimate Std. Error t value Pr(>|t|)    
(Intercept)              -1.783e+02  8.587e+01  -2.076  0.03809 *  
Model.Year                9.640e-02  4.255e-02   2.266  0.02363 *  
Eng.Displacement         -1.364e+00  1.025e-01 -13.306  < 2e-16 ***
No.Cylinders              4.644e-02  6.769e-02   0.686  0.49282    
AspirationOT             -3.452e-01  6.352e-01  -0.543  0.58693    
AspirationSC             -9.197e-01  2.282e-01  -4.031 5.85e-05 ***
AspirationTC             -1.303e+00  1.288e-01 -10.111  < 2e-16 ***
AspirationTS             -1.149e+00  4.945e-01  -2.323  0.02035 *  
No.Gears                 -1.307e-01  2.995e-02  -4.364 1.37e-05 ***
Lockup.Torque.ConverterY -8.243e-01  1.117e-01  -7.377 2.78e-13 ***
Drive.SysA               -8.339e-02  1.521e-01  -0.548  0.58356    
Drive.SysF                1.441e+00  1.711e-01   8.419  < 2e-16 ***
Drive.SysP               -2.400e-01  2.980e-01  -0.805  0.42087    
Drive.SysR                4.328e-02  1.476e-01   0.293  0.76938    
Max.Ethanol              -7.076e-03  2.967e-03  -2.385  0.01722 *  
Fuel.TypeGM               5.706e-01  4.173e-01   1.368  0.17169    
Fuel.TypeGP               4.093e-01  1.369e-01   2.990  0.00284 ** 
Fuel.TypeGPR              1.363e-01  1.401e-01   0.973  0.33096    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.598 on 1382 degrees of freedom
Multiple R-squared:  0.6628,    Adjusted R-squared:  0.6586 
F-statistic: 159.8 on 17 and 1382 DF,  p-value: < 2.2e-16
```

Which predictors/variables do you think are possibly associated with fuel efficiency (use 0.05 significant level), and why

Answer

By looking at the statistics given by `summary()` function, we can see that 6 predictors are possibly associated with Fuel efficiency. The 6 possibly associated predictors are

1. Eng.Displacement
2. AspirationSC
3. AspirationTC
4. No.Gears
5. Lockup.Torque.ConverterY
6. Drive.SysF

We can say the above 6 predictors are possibly associated to fuel efficiency because the t values are relatively far from 0, suggesting to reject the null hypothesis that `beta` is 0. And also, the t value is relatively large compared to the standard error, which indicate a relationship exists between these 6 predictors and Fuel

efficiency.

Finally looking at the last coefficeint `Pr(>|t|)` , we see for the 6 prdeictors the probability of observing any value equal or larger than t is really low. This supports our inference that the 6 predictors are related to target variable Fuel efficiency.

Which three predictors/variables appear to be the strongest predictors of fuel efficiency, and why?

Answer

The 3 predictors/variables that appear to be the strongest predictors of fuel efficiency are

1. Eng.Displacement
2. AspirationTC
3. Drive.SysF

We can say this because the value of `Pr(>|t|)` suggsets that the probablity of seeing a t value as large as this is really low. Means that we can reject the null hypothesis that `beta` is 0, thus suggesting that these 3 predictors are the most strongly related to Fuel Efficiency.

Question 2 (5 Marks)

Describe/discuss the effect that the year of manufacture `(Model.Year)` variable appears to have on the mean `fuel efficiency` . Additionally, describe/discuss the effect that the number of gears `(No.Gears)` variable has on the mean `fuel efficiency` of the car.

Note: This asks for your descriptions, please refrain from using one or two lines to describe/discuss the effect. Keep answers to be 4 decimal places

In [4]:

```
# Fit a Multiple Linear regression model
# Target variable : Comb.FE
# Predictor : Year
lm.fit.Year <- lm(Comb.FE ~ Model.Year , data = train)
summary(lm.fit.Year)
```

```
Call:
lm(formula = Comb.FE ~ Model.Year, data = train)

Residuals:
    Min      1Q  Median      3Q     Max 
-5.4805 -1.8537 -0.4876  1.3016 15.7799 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -9.042125 144.142701  -0.063    0.950
Model.Year    0.009656   0.071418   0.135    0.892

Residual standard error: 2.736 on 1398 degrees of freedom
Multiple R-squared:  1.308e-05, Adjusted R-squared:  -0.0007022 
F-statistic: 0.01828 on 1 and 1398 DF,  p-value: 0.8925
```

Answer

Looking at the above summary of Linear model of Comb.FE and the Model.Year we see that the t value is 0.135, which is really small. This small t value supports the null hypothesis of of `beta` being 0. Thus we can conclude that no relationship exists between `Model.Year` and `fuel efficiency` .

We now look at the probablity of getting a t value greater than 0.135.

We know that when p value < 0.05, there exists a relation between predictor and the target variable.

In this case `Pr(>|t|)` value is 0.892 > 0.05, this means that the variable `Model.Year` has no effect on `fuel effiency` .

In [5]:

```
# Fit a Multiple Linear regression model
# Target variable : Comb.FE
# Predictor : No.Gears
lm.fit.No.Gears <- lm(Comb.FE ~ No.Gears , data = train)
summary(lm.fit.No.Gears)
```

```
Call:
lm(formula = Comb.FE ~ No.Gears, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-5.6520 -1.6453 -0.2262  1.5197 15.1466

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 14.95914    0.27586   54.23   <2e-16 ***
No.Gears    -0.64691    0.03838  -16.86   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.494 on 1398 degrees of freedom
Multiple R-squared:  0.1689,    Adjusted R-squared:  0.1683
F-statistic: 284.1 on 1 and 1398 DF,  p-value: < 2.2e-16
```

Looking at the above summary of Linear model of Comb.FE and the No.Gears we see that the t-value is -16.86, which is large. This large t-value suggests to reject the null hypothesis of of `beta` being 0. Thus we can conclude that there exists a relationship between `No.Gears` and `fuel efficiency` .

We now look at the probablity of getting a t-value greater than 16.86.

We know that when p value < 0.05, there exists a relation between predictor and the target variable.

In this case `Pr(>|t|)` value is 2e-16 < 0.05, this means that the variable `Model.Year` is related to `fuel effiency` .

Question 3 (5 Marks)

Apply the stepwise selection procedure with the **BIC** penalty to prune out potentially less significant variables. Write down the final regression equation obtained after pruning, please keep the values of the parameter coefficients to 2 decimal places. Finally, also describe the pruned model.

Note: please don't change the default direction `both` in the step function, this is so that we can check your work easily. Additionally, please name this model `sw.fit`

YOUR ANSWER HERE

In [6]:

```
# perform Stepwise selection procedure with the  BIC  penalty on the Linear model
sw.fit <- step(lm.fit,k=log(nrow(train)),trace=0,direction = c("both"))

# show the statistics of the prunned linear model
summary(sw.fit)
```

```
Your code contains a unicode char which cannot be displayed in your
current locale and R will silently convert it to an escaped form when the
R kernel executes this code. This can lead to subtle errors if you use
such chars to do comparisons. For more information, please see
https://github.com/IRkernel/repr/wiki/Problems-with-unicode-on-windows (http
s://github.com/IRkernel/repr/wiki/Problems-with-unicode-on-windows)


Call:
lm(formula = Comb.FE ~ Eng.Displacement + Aspiration + No.Gears +
    Lockup.Torque.Converter + Drive.Sys + Max.Ethanol, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-4.0743 -0.9760 -0.0349  0.6566 11.3971

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              16.196874   0.282901  57.253  < 2e-16 ***
Eng.Displacement         -1.277173   0.043418 -29.416  < 2e-16 ***
AspirationOT             -0.100081   0.626276  -0.160 0.873060
AspirationSC             -0.699137   0.213768  -3.271 0.001100 **
AspirationTC             -1.144227   0.107302 -10.664  < 2e-16 ***
AspirationTS             -1.122104   0.481471  -2.331 0.019919 *
No.Gears                 -0.113537   0.029183  -3.891 0.000105 ***
Lockup.Torque.ConverterY -0.825285   0.110202  -7.489 1.23e-13 ***
Drive.SysA                0.035013   0.145617   0.240 0.810020
Drive.SysF                1.480191   0.166847   8.872  < 2e-16 ***
Drive.SysP               -0.323201   0.292617  -1.105 0.269560
Drive.SysR                0.093779   0.146329   0.641 0.521710
Max.Ethanol              -0.008344   0.002934  -2.843 0.004528 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.605 on 1387 degrees of freedom
Multiple R-squared:  0.6586,    Adjusted R-squared:  0.6557
F-statistic:   223 on 12 and 1387 DF,  p-value: < 2.2e-16
```

Answer

After applying the stepwise selection procedure with the **BIC** penalty, we can see that the below 6 predictors are important:

1. Eng.Displacement
2. AspirationSC
3. AspirationTC

4. No.Gears
5. Lockup.Torque.ConverterY
6. Drive.SysF

These 6 predictors are important as the p value is low for them, which suggest to reject the null hypothesis of `beta` being 0.

So the final regression equation after using Stepwise selection procedures using BIC penalty is as follows :

y = -(1.27)Eng.Displacement -(0.69)AspirationSC -(1.14)AspirationTC -(0.11)No.Gears -(0.82)Lockup.Torque.ConverterY + (1.48)Drive.SysF + 16.196874

Question 4 (5 Marks)

Say we are going to buy a new car and we want to improve the fuel efficiency of our new car, what does this `BIC model` suggest we should do? Provide a detailed answers of at least `150 words` .

Answer

The prunned Linear model gives us the important variables and their respective coefficients to calculate the final Fuel efficiency.

The linear equation has the form y = mx + c, where y: Fuel efficiency, m: coefficient, x: predictor and c: intercept

Final equation after using Stepwise selection procedures using BIC penalty:

y = -(1.27)Eng.Displacement -(0.69)AspirationSC -(1.14)AspirationTC -(0.11)No.Gears -(0.82)Lockup.Torque.ConverterY + (1.48)Drive.SysF + 16.196874

Evaluating the Categorical variables: 1. Aspiration : As aspiration is a categorical variables, we evaluate each value namely N, TC, SC, OT and TS independently. Looking at the summary of the final linear model, the coefficient of Aspiration OT is the largest of the 5 values of Aspiration. This value of Aspirtaion OT will help maxizmize the fuel efficinecy.

2. Lockup.Torque.Converter: The summary of linear model suggests to have a car with Lockup Torque Converter as this will help maximize the Fuel efficiency.

3. Drive.Sys : The model suggests to have a car with Front wheel drive to have maximum fuel efficiency. We can conclude this looking at the coeffiecient of the Drive.Sys.F. It is the highest amongst the other values, thus will contribute more to the final equation.

4. Fuel.Type: Looking at the final quation of the prunned model, we can see that the Fuel.Type variable is not pressent. This means the fuel type of the car is not important and does not contribute the fuel efficiency.

Evaluating the Numerical variables:

1. Model.Year: As the final linear equation after prunning does not have Model.Year in it, we can say that Year is not important for Fuel efficiency.

2. Eng.Displacement : The coefficient of Eng.Displacement is -1.277173, which means higher the value of Eng.Displacement, lower will be the Fuel Efficiency. Hence the model suggests to have lower Eng.Displacement.

3. No.Cylinders : As the final linear equation after prunning does not have No.Cylinders in it, we can say that No.Cylinders is not important for Fuel efficiency.

4. No.Gears : The coefficient of No.Gears is -0.113537, which means higher the No.Gears, lower will be the Fuel Efficiency. Hence the model suggests to have lower No.Gears.

5. Max.Ethanol : The coefficient of Max.Ethanol is -0.008344, which means higher the percentage of Max.Ethanol, lower will be the Fuel Efficiency. Hence the model suggests to have lower % of Ethanol.

Question 5 (5 Marks)

Imagine that you are looking for a new car to buy to replace your existing car. Use the **test** dataset to inspect the first car fuel efficiency and see whether it is a good fit for you or not.

```
(a) Use your BIC model to predict the mean fuel efficiency for this new car. Provi
de a 95% confidence interval for this prediction. [2 mark]
(b) Following the previous estimation, given that the current car that you own has
a mean fuel efficiency of 9.5 km/l (measured over the life time of your ownershi
p), does your model (BIC) suggest that the new car will have better fuel efficienc
y than your current car? Why? [3 marks]
```

YOUR ANSWER HERE

Answer 5.a

Expected Mean Fule efficiency of the new car is 9.287257 km/l.

The 95% confidence interval for this prediction is [9.052956, 9.521557] km/l

In [7]:

```
predict(sw.fit,test[1,],interval = "confidence", prediction.interval = 95)
```

fit	lwr	upr
9.287257	9.052956	9.521557

Answer 5.b

Fuel efficiency of current car : 9.5 km/l

Estimated fuel efficiency of new car : 9.287257 km/l

So, the new car might not have better Fuel efficiency compared to the current car according to our Linear model. We can be 95% sure about this prediction value. Also, the upper limit for this prediction is 9.521557 km/l, so there is less probobablity of the new car being better than the old car.

Analysis of why the new Car is not better :

1. Eng.Displacement : The new car has Eng. Displacement of 3.9. From our linear model we can see that higher this value, lower is the Fuel efficiency.

2. Aspiration : The new car has aspiration TC. The coeffiecien to of AspirationTC is -1.144227, which suggests that this will decrease the Fuel efficiency of the new car. AspirationTC is the variable that contributes most to the decrease in the Fuel efficiency of the new car.

3. No.Gears : The new car has 6 gears. According to our model higher no. of gears will decrease the Fuel efficiency.

4. Lockup.Torque.Converter : The new car does not have Lockup Torque converter, so according to our model, the new car has less Fuel efficiency as it lacks this feature.

5. Drive.Sys : For good fuel efficiency our model suggests to have Front drive. As the new car has Rear wheels drive, the fuel efficiency is compariteveley less than the one with Front wheel drive.

6. Max.Ethanol : The new car has Ethanol = 10%, according to our model, less the value better is the Fuel efficiency.

Question 6 (Libraries are allowed) (25 Marks)

As a Data Scientist, one of the key tasks is to build models **most appropriate/closest** to the truth; thus, modelling will not be limitted to these steps in the assignment. To simulate for a realistic modelling process, this question will be in the form of a competition among students to find out who has the best model.

Thus, You will be graded by the performance of your model compared to your classmates', the better your model, the higher your score. Additionally, you need to write a short paragraph describing/documenting your thought process in this model building process `(300 words)` . Note that this is to explain to us why you build your current model so that we can verify that you understand the model you build and not just copy from other people.

Note Please make sure that we can install the libraries that you use in this part, the code structure can be:

```
install.packages("some package", repos='http://cran.us.r-project.org')
```

```
library("some package")
```

Remember that if we cannot run your code, we will have to give you 0 marks, our suggestion is for you to use the standard `R version 3.6.1`

You also need to name your final model `fin.mod` so we can run a check to find out your performance. A good test for your understanding would be to set the previous **BIC model** to be the final model to check if your code works Appropriately.

20 Marks for the model performance in the competition

5 Marks for logically writing down the thought process in building the final model

This is the link (https://www.kaggle.com/t/0a3c0fc91b074816a6315bb4e9b42602) to the competition

ANSWER

The main objective of this task is to build a model that most appropriately resembles the true value of Fuel efficiency. This involves predicting the numerical value of Fuel efficiency given the different features of the car. Therefore, it qualifies to be a Regression problem.

To start with the Model building process, we first use the Multiple Linear regression model which we prunned using BIC penalty. Uisng this model as the starting point, we run the predictions. This gives a Root mean Square (RMSE) value of around 1.69275.

To improve our predictions of Fuel Efficiency, we try out different Regression Algorithms to have a better model that most appropriately predicts the Fuel efficiency.

Following are the different algorithms I have tried for this Regression task with the respective approximate RMSE values obatined:

1. Recursive Partitioning And Regression Trees - 1.46625
2. Linear Regression model (Prunned) - 1.69275
3. kNN - 1.44106
4. SVM - 1.38612
5. Boosted regression trees - 1.55203
6. Random Forest - RMSE - 1.25855

Looking at the above RMSE values, its best to use Random Forest for this regression task. As Random forest randomly selects the predictors and performs the split, it uses the information of error to have a better guess of the next predictors. This helps improve the performance compared to using a Decision tree algorithm/

To have a robust model, I perform k-fold validation. In this the training data set is divided into k subset (k=10). Before we divide the training dataset into k subsets, I randomly shuffle the training dataset to remove any ordering of the data.

For each subset the Regression model is trained using the subset of training data and the model is updated in each iteration. The final model is used for making the prediction of the test data.

The parameters tuned for the Random Forest are `ntree` and `mtry` .

1. `ntree` : Number of tree to grow in the random forest.
2. `mtry` : Number of predictors randomly sampled as potential candidates for each split.

`mtry` = 6, because the prunned Linear model suggested we have 6 significant predictors. Also, having large `mtry` leads to overfitting.

`ntree` = 100, as the algorithm stabalises and shows no major improvement.

In [8]:

```
# Use this function to check the performance of your model
rmse <- function(pred.label, truth.label){
    # Lower is better
    return(sqrt(mean((pred.label - truth.label)^2)))
}
```

In [9]:

```
# packages to install
install.packages("caret")
install.packages("rpart")
install.packages("randomForest")
```

```
Installing package into 'C:/Users/prash/OneDrive/Documents/R/win-library/3.
6'
(as 'lib' is unspecified)

package 'caret' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\prash\AppData\Local\Temp\RtmpOoIq10\downloaded_packages

Installing package into 'C:/Users/prash/OneDrive/Documents/R/win-library/3.
6'
(as 'lib' is unspecified)

package 'rpart' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\prash\AppData\Local\Temp\RtmpOoIq10\downloaded_packages

Installing package into 'C:/Users/prash/OneDrive/Documents/R/win-library/3.
6'
(as 'lib' is unspecified)

package 'randomForest' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\prash\AppData\Local\Temp\RtmpOoIq10\downloaded_packages
```

In [9]:

```
# packages required
library(caret)
library(rpart)
library(randomForest)
```

```
Warning message:
"package 'caret' was built under R version 3.6.3"Loading required package: l
attice
Warning message:
"package 'lattice' was built under R version 3.6.3"Loading required package:
ggplot2
Warning message:
"package 'ggplot2' was built under R version 3.6.3"Warning message:
"package 'rpart' was built under R version 3.6.3"Warning message:
"package 'randomForest' was built under R version 3.6.3"randomForest 4.6-14
Type rfNews() to see new features/changes/bug fixes.

Attaching package: 'randomForest'

The following object is masked from 'package:ggplot2':

    margin
```

In [10]:

```
## Training the Regression model

#Randomly shuffle the data
train<-train[sample(nrow(train)),]

#Create 10 equally size folds
folds <- cut(seq(1,nrow(train)),breaks=10,labels=FALSE)

#Perform k fold ( k = 10) cross validation
for(i in 1:10){

    #Segement training data
    testIndexes <- which(folds==i,arr.ind=TRUE)

    # train and test data
    testData <- train[testIndexes, ]
    trainData <- train[-testIndexes, ]

    # build a model, if doesnt exist
    if(i == 1){
        fin.mod  <- randomForest(Comb.FE ~ ., data = trainData, ntree=100,mtry = 6)
    }
    # update the existing model
    else{
        update(fin.mod , data = trainData)
    }
}
```

In [11]:

```
# get the common columns between training and test data
common <- intersect(names(train), names(test))

for (p in common)
{
    # make the levels same for factors of the test and training data
    if (class(train[[p]]) == "factor")
    {
        levels(test[[p]]) <- levels(train[[p]])
    }
}
```

In [12]:

```
# make predictions using the final model
pred.label <- predict(fin.mod, test)
```

In [13]:

```
# PLEASE DO NOT ALTER THIS CODE BLOCK
# put this label in a csv file to commit to the Leaderboard
write.csv(data.frame("RowIndex" = seq(1, length(pred.label)), "Prediction" = pred.label),
          "RegressionPredictLabel.csv", row.names = F)
```

In [ ]:

```
## PLEASE DO NOT ALTER THIS CODE BLOCK
## Please skip (don't run) this if you are a student
## For teaching team use only
RMSE.fin <- rmse(pred.label, label$Label)
cat(paste("RMSE is", RMSE.fin))
```

Part 2: Classification (50 Marks)

In this part, you are going to work with "Census Income Dataset" which was originally donated by Ronny Kohavi and Barry Becker to UCI (University of California, Irvine) in 1996. This is a trimmed dataset used for machine learning students to study classification.

This dataset has collected over 40,000 records (we excluded some data in our version) regarding personal yearly income with 12 attributes (predictors). The attributes comprise many aspects of a person that may contribute to the yearly income. You can use summary() function to obtain the attributes information. Your prediction task is to determine whether a person makes over 50K a year.

We have splitted the dataset into a trainning and a testing set. There are 27245 records in the training set while 13631 records in the testing set. Besides the 12 predictors, there is one more column named Salary indicating whether a person's yearly income is over 50K. The label information is a seperated file for the testing set and will be used by us to asess your performance later. Note the label TRUE means an individual's yearly salary exceeds 50K while FALSE means an individual's yearly salary is under 50K.

Note: If not explicitly mentioned, libraries are not allowed

In [14]:

```
# Read the data from students' side
remove(list = ls())
train <- read.csv("ClassTrain.csv")
test  <- read.csv("ClassTest.csv")
```

In [ ]:

```
## PLEASE DO NOT ALTER THIS CODE BLOCK
# Please skip (don't run) this if you are a student
# Read in the data from marking tutors' side (ensure no cheating!)
remove(list = ls())
train <- read.csv("../data/ClassTrain.csv")
test  <- read.csv("../data/ClassTest.csv")
label <- read.csv("../data/ClassTestLabel.csv")
```

Question 1 (10 Marks)

Fit a **Generalized Linear Model (Logistic Regression)** to predict level of income (salary) (≥ 50 K, or < 50 K) using the `train` dataset. Using the results of fitting this model, which predictors do you think are possibly associated with the level of Salary (use `0.05` significant level), and why? Which `three variables` appear to be the strongest predictors of salary, and why?

Furthermore, you can see that you have much more predictors in this part than in the `linear model` from Part 1 $\Rightarrow$ manually checking information is counterproductive. Thus, please write a function to automate these processes **(1)** selecting important feature against 0.05 threshold and **(2)** Selecting three most important features.

Note: You don't have to worry about categorical variables here since R can deal with this automatically, focus your efforts on interpretation. Additionally, when explaining why features are strongly associated with the target, please refrain from giving one or two sentences answers, these answers are not descriptive and will result in a deduction of marks. Finally, please name the model here `glm.fit` and have the parameter in the model set to `family = binomial` .

YOUR ANSWER HERE

In [15]:

```
# Building a Generalized Linear Model for Classification
# Predictors : all columns
# Target variable : Salary
glm.fit = glm(Salary~. , data=train, family=binomial)
```

```
Warning message:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"
```

In [16]:

```
# obtain statistics of the GLM model
summary(glm.fit)
```

```
Call:
glm(formula = Salary ~ ., family = binomial, data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-5.1013  -0.5296  -0.1926   0.0276   3.4349

Coefficients:
                                  Estimate Std. Error z value Pr(>|z|)
(Intercept)                     -7.614e+00  4.525e-01 -16.826  < 2e-16 **
*
Age                              2.626e-02  1.779e-03  14.762  < 2e-16 **
*
WorkClassLocal-gov              -7.214e-01  1.168e-01  -6.179 6.46e-10 **
*
WorkClassPrivate                -4.734e-01  9.693e-02  -4.884 1.04e-06 **
*
WorkClassSelf-emp-inc           -2.974e-01  1.283e-01  -2.317 0.020506 *
WorkClassSelf-emp-not-inc       -9.994e-01  1.139e-01  -8.772  < 2e-16 **
*
WorkClassState-gov              -7.757e-01  1.294e-01  -5.996 2.03e-09 **
*
FinalWeight                      7.896e-07  1.822e-07   4.334 1.46e-05 **
*
Education11th                    6.909e-02  2.201e-01   0.314 0.753589
Education12th                    5.005e-01  2.940e-01   1.702 0.088676 .
Education7th-8th                -6.213e-01  2.592e-01  -2.397 0.016530 *
Education9th                    -2.472e-01  2.856e-01  -0.865 0.386877
EducationAssoc-acdm              1.302e+00  1.843e-01   7.066 1.60e-12 **
*
EducationAssoc-voc               1.263e+00  1.772e-01   7.127 1.02e-12 **
*
EducationBachelors               1.931e+00  1.647e-01  11.724  < 2e-16 **
*
EducationDoctorate               3.076e+00  2.380e-01  12.926  < 2e-16 **
*
EducationHS-grad                 7.790e-01  1.598e-01   4.874 1.09e-06 **
*
EducationMasters                 2.319e+00  1.767e-01  13.126  < 2e-16 **
*
EducationProf-school             2.874e+00  2.145e-01  13.396  < 2e-16 **
*
EducationSome-college            1.108e+00  1.622e-01   6.832 8.36e-12 **
*
MaritalStatusMarried-civ-spouse  2.345e+00  3.050e-01   7.687 1.51e-14 **
*
MaritalStatusMarried-spouse-absent -3.345e-02  2.697e-01  -0.124 0.901286
MaritalStatusNever-married      -4.513e-01  9.187e-02  -4.912 9.01e-07 **
*
MaritalStatusSeparated          -9.621e-02  1.733e-01  -0.555 0.578829
MaritalStatusWidowed             1.484e-01  1.656e-01   0.896 0.370163
OccupationCraft-repair           5.906e-02  8.379e-02   0.705 0.480884
OccupationExec-managerial        7.693e-01  8.089e-02   9.511  < 2e-16 **
*
OccupationFarming-fishing       -9.919e-01  1.457e-01  -6.805 1.01e-11 **
```

```
*
OccupationHandlers-cleaners        -7.641e-01  1.529e-01  -4.999 5.77e-07 **
*
OccupationMachine-op-inspct        -2.794e-01  1.073e-01  -2.605 0.009191 **
OccupationOther-service            -8.967e-01  1.300e-01  -6.900 5.22e-12 **
*
OccupationProf-specialty            4.654e-01  8.613e-02   5.403 6.54e-08 **
*
OccupationProtective-serv           6.229e-01  1.302e-01   4.784 1.72e-06 **
*
OccupationSales                     2.770e-01  8.625e-02   3.211 0.001322 **
OccupationTech-support              6.359e-01  1.159e-01   5.488 4.07e-08 **
*
OccupationTransport-moving         -1.027e-01  1.032e-01  -0.995 0.319541
RelationshipNot-in-family           6.652e-01  3.021e-01   2.202 0.027683 *
RelationshipOther-relative         -4.067e-01  2.918e-01  -1.394 0.163395
RelationshipOwn-child              -6.044e-01  2.943e-01  -2.054 0.039994 *
RelationshipUnmarried               5.707e-01  3.174e-01   1.798 0.072185 .
RelationshipWife                    1.332e+00  1.103e-01  12.071  < 2e-16 **
*
RaceAsian-Pac-Islander              9.879e-01  2.997e-01   3.296 0.000979 **
*
RaceBlack                           3.929e-01  2.427e-01   1.619 0.105400
RaceOther                           1.524e-01  4.397e-01   0.347 0.728862
RaceWhite                           5.396e-01  2.305e-01   2.340 0.019266 *
GenderMale                          8.679e-01  8.403e-02  10.328  < 2e-16 **
*
CapitalGain                         3.191e-04  1.107e-05  28.830  < 2e-16 **
*
CapitalLoss                         6.503e-04  3.999e-05  16.264  < 2e-16 **
*
HoursWork                           2.965e-02  1.774e-03  16.714  < 2e-16 **
*
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 31005  on 27244  degrees of freedom
Residual deviance: 17976  on 27196  degrees of freedom
AIC: 18074

Number of Fisher Scoring iterations: 7
```

1.Selecting important feature against 0.05 threshold

In [17]:

```
# Function to return important features against 0.05 significant level
get_important_features = function(model){

    # store pvalue and predictor as data frame
    df_imp_features = data.frame(summary(model)$coef[(summary(model)$coef[,4] < 0.05), 4])

    # rename column
    names(df_imp_features) = "p_value"

    # return pvalue and predictor as data frame
    return (df_imp_features)
}
```

In [18]:

```
# get the important features of the glm.fit model
imp_features = get_important_features(glm.fit)
imp_features
```

	p_value
(Intercept)	1.561804e-63
Age	2.591338e-49
WorkClassLocal-gov	6.457094e-10
WorkClassPrivate	1.039016e-06
WorkClassSelf-emp-inc	2.050583e-02
WorkClassSelf-emp-not-inc	1.747638e-18
WorkClassState-gov	2.026573e-09
FinalWeight	1.464846e-05
Education7th-8th	1.653000e-02
EducationAssoc-acdm	1.599694e-12
EducationAssoc-voc	1.023955e-12
EducationBachelors	9.636007e-32
EducationDoctorate	3.224490e-38
EducationHS-grad	1.094343e-06
EducationMasters	2.349945e-39
EducationProf-school	6.362770e-41
EducationSome-college	8.363157e-12
MaritalStatusMarried-civ-spouse	1.510914e-14
MaritalStatusNever-married	9.009754e-07
OccupationExec-managerial	1.887085e-21
OccupationFarming-fishing	1.007389e-11
OccupationHandlers-cleaners	5.765385e-07
OccupationMachine-op-inspct	9.191178e-03
OccupationOther-service	5.217529e-12
OccupationProf-specialty	6.543650e-08
OccupationProtective-serv	1.718776e-06
OccupationSales	1.321770e-03
OccupationTech-support	4.065421e-08
RelationshipNot-in-family	2.768284e-02
RelationshipOwn-child	3.999385e-02
RelationshipWife	1.499247e-33
RaceAsian-Pac-Islander	9.792498e-04
RaceWhite	1.926637e-02
GenderMale	5.272260e-25
CapitalGain	9.086632e-183

	p_value
CapitalLoss	1.791718e-59
HoursWork	1.033315e-62

After selecting the important features against 0.05 significance, we can say that the below 36 predictors are possibly associated with Salary.

1. 'Age'
2. 'WorkClassLocal-gov'
3. 'WorkClassPrivate'
4. 'WorkClassSelf-emp-inc'
5. 'WorkClassSelf-emp-not-inc'
6. 'WorkClassState-gov'
7. 'FinalWeight'
8. 'Education7th-8th'
9. 'EducationAssoc-acdm'
10. 'EducationAssoc-voc'
11. 'EducationBachelors'
12. 'EducationDoctorate'
13. 'EducationHS-grad'
14. 'EducationMasters'
15. 'EducationProf-school'
16. 'EducationSome-college'
17. 'MaritalStatusMarried-civ-spouse'
18. 'MaritalStatusNever-married'
19. 'OccupationExec-managerial'
20. 'OccupationFarming-fishing'
21. 'OccupationHandlers-cleaners'
22. 'OccupationMachine-op-inspct'
23. 'OccupationOther-service'
24. 'OccupationProf-specialty'
25. 'OccupationProtective-serv'
26. 'OccupationSales'
27. 'OccupationTech-support'
28. 'RelationshipNot-in-family'
29. 'RelationshipOwn-child'
30. 'RelationshipWife'
31. 'RaceAsian-Pac-Islander'
32. 'RaceWhite'
33. 'GenderMale'
34. 'CapitalGain'
35. 'CapitalLoss'
36. 'HoursWork'

This is because the p value for these predictors is very small which suggests a relation between the predictors and the target variable Salary.

2.Selecting three most important features.

In [19]:

```
# Function to get the top 3 predictors
get_top_3_predictors = function(model_summary){

    # remove 'Intercept' as it is not a predictor of the model
    df_pvalue = subset(model_summary, rownames(model_summary) != '(Intercept)')

    # sort the df according to p_value
    lst = sort(df_pvalue[,1], index.return=TRUE, decreasing=FALSE)

    # get the top 3 predictors after sort
    imp_3_features = lapply(lst, `[`,lst$x %in% head(unique(lst$x),3))

    # return the name of the top 3 predictors
    return (rownames(df_pvalue)[imp_3_features$ix])
}
```

In [20]:

```
# get the top 3 predictors of our 'glm' model
get_top_3_predictors(imp_features)
```

'CapitalGain' 'HoursWork' 'CapitalLoss'

To get the top 3 most important predictors we see the smallest p-value. A small p-value suggests a relation between the predictors and the target variable.

'CapitalGain', 'HoursWork' and 'CapitalLoss' have the smallest p-values. Therefore, these are the top 3 most signinficant predictors.

Question 2 (10 Marks)

Firstly, please use the model created in the previous question to predict for the labels of the **train** data. Consequently, our objective is to compare this `predict.label` with the `truth.label` from the **test** data. However, as we don't know the **test** label, we have to estimate model performance using **train** data at this moment.

Secondly, since our objective is to estimate the performance of this model in making correct predictions; thus, this question also asks you to explore different performance metrics (https://en.wikipedia.org/wiki/Precision_and_recall) for classification models. The metrics we will use are **Accuracy, Precision, Recall, and F1 Score**, please create a function to calculate these value and print them out properly using the given structure.

Additionally, please also discuss the results of these values in the context of your model.

Note: This asks for your descriptions, please refrain from using one or two lines to describe/discuss the effect. Keep answers to be 4 decimal places

YOUR ANSWER HERE

In [21]:

```
# Apply your previous model to perform prediction, keep type = "response"
# Don't worry if you receive some warnings, they are benign
predict.label <- predict(glm.fit, train, type="response")
# Truth label from train data
truth.label <- train$Salary
```

In [22]:

```
# Model statistics function
mod.stat <- function(predict.label, truth.label){
    # instantiate the variables
    accuracy <- NULL
    precision <- NULL
    recall <- NULL
    F1 <- NULL

    # convert to TRUE and FALSE
    predict.label = ifelse(predict.label >= 0.5, TRUE, FALSE)

    # accuracy
    accuracy = mean(predict.label == truth.label)

    # True positives
    true_positives  = sum( (predict.label == TRUE ) & (truth.label == TRUE ) )
    # False positives
    false_positives = sum( (predict.label == TRUE) & (truth.label == FALSE) )

    # Precision
    precision = true_positives / (true_positives + false_positives)

    # False negatives
    false_negatives = sum( (predict.label == FALSE) & (truth.label == TRUE) )

    # Recall
    recall = true_positives / (true_positives + false_negatives)

    # f score
    F1 = 2*precision*recall / (precision+recall)

    # Return a list of value
    return(list("accuracy" = round(accuracy,4), "precision" = round(precision,4), "recall"
}
```

In [23]:

```
# Run the function to get statistics, provide description/discussion after this
mod.stat(predict.label, truth.label)
```

$accuracy
0.8452
$precision
0.7395
$recall
0.6107
$fscore
0.669

Accuracy :

The Generalised Linear Regression model classifies the test data corretly 84.52% of the times. This means it predicts TRUE whens the gorund truth is TRUE, and predicts FALSE when ground truth is FALSE for 84.52% of the times of the test dataset.

Precision :

Precision of our model is 73.95%. This means of all the data classified as TRUE by the model, the prediction was correct for 73.95% of the times. In our case, we correctly identify salary being > 50K out of all the person actually having this to be true.

Recall :

Recall of the model our model is 61.07%. This means, that the prediction was TRUE for 61.07% of the times with the ground truth. In our case, the model was able to classsiy the salary > 50K correcly for 61.07% of the times.

fscore :

Both precision and recall independently are not good to test our model. A model can have a good precision and bad recall, and vice versa. This issue can be solved using the fscore, which is the Harmonic means of the recall and precision. It provides a way to describe both the precision and recall in one metric.

Our model has a fscore of 0.669 which is not bad. A good fscore is 1 and a bad fcore is 0.

Question 3 (5 Marks)

Use the stepwise selection procedure with the **BIC** penalty to prune out potentially unimportant variables. Checking the performance of your model using the created `mod.stat()` function, please give your discussion as how this model is compared with the `glm.fit` (you can run the `mod.stat()` function for this as well if you want to).

Note: please don't change the default direction `both` in the step function, this is so that we can check your work easily. Additionally, please name this model `sw.fit` . Don't worry about the warnings, they are benign

In [24]:

```
# Setting to suppress warnings
options(warn=-1)

# Fit a stepwise model
sw.fit <- step(glm.fit,k=log(nrow(train)),trace=0,direction = c("both"))

# Setting to suppress warnings
options(warn=0)

# Getting the summary to understand the result
summary(sw.fit)
```

```
Call:
glm(formula = Salary ~ Age + WorkClass + FinalWeight + Education +
    MaritalStatus + Occupation + Relationship + Gender + CapitalGain +
    CapitalLoss + HoursWork, family = binomial, data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-5.0961  -0.5291  -0.1936   0.0279   3.4393

Coefficients:
                                Estimate Std. Error z value Pr(>|z|)
(Intercept)                   -7.130e+00  3.929e-01 -18.146  < 2e-16
***
Age                            2.644e-02  1.778e-03  14.869  < 2e-16
***
WorkClassLocal-gov            -7.159e-01  1.163e-01  -6.154 7.54e-10
***
WorkClassPrivate              -4.588e-01  9.626e-02  -4.766 1.88e-06
***
WorkClassSelf-emp-inc         -2.776e-01  1.278e-01  -2.173  0.02977
*
WorkClassSelf-emp-not-inc     -9.857e-01  1.133e-01  -8.703  < 2e-16
***
WorkClassState-gov            -7.653e-01  1.290e-01  -5.930 3.02e-09
***
FinalWeight                    7.496e-07  1.800e-07   4.164 3.13e-05
***
Education11th                  7.190e-02  2.201e-01   0.327  0.74395
Education12th                  5.065e-01  2.939e-01   1.724  0.08479
.
Education7th-8th              -6.220e-01  2.593e-01  -2.399  0.01643
*
Education9th                  -2.417e-01  2.851e-01  -0.848  0.39663
EducationAssoc-acdm            1.323e+00  1.841e-01   7.187 6.60e-13
***
EducationAssoc-voc             1.277e+00  1.770e-01   7.215 5.38e-13
***
EducationBachelors             1.947e+00  1.645e-01  11.838  < 2e-16
***
EducationDoctorate             3.089e+00  2.377e-01  12.998  < 2e-16
***
EducationHS-grad               7.881e-01  1.596e-01   4.937 7.95e-07
***
EducationMasters               2.337e+00  1.765e-01  13.239  < 2e-16
***
```

```
EducationProf-school                2.894e+00  2.145e-01  13.495  < 2e-16
***
EducationSome-college               1.117e+00  1.621e-01   6.893 5.47e-12
***
MaritalStatusMarried-civ-spouse     2.357e+00  3.045e-01   7.743 9.75e-15
***
MaritalStatusMarried-spouse-absent -3.049e-02  2.690e-01  -0.113  0.90977
MaritalStatusNever-married         -4.482e-01  9.172e-02  -4.886 1.03e-06
***
MaritalStatusSeparated             -1.101e-01  1.728e-01  -0.637  0.52405
MaritalStatusWidowed                1.497e-01  1.655e-01   0.904  0.36583
OccupationCraft-repair              6.350e-02  8.372e-02   0.759  0.44812
OccupationExec-managerial           7.726e-01  8.083e-02   9.558  < 2e-16
***
OccupationFarming-fishing          -9.869e-01  1.456e-01  -6.779 1.21e-11
***
OccupationHandlers-cleaners        -7.678e-01  1.528e-01  -5.026 5.02e-07
***
OccupationMachine-op-inspct        -2.857e-01  1.072e-01  -2.665  0.00770
**
OccupationOther-service            -9.059e-01  1.298e-01  -6.981 2.92e-12
***
OccupationProf-specialty            4.656e-01  8.599e-02   5.414 6.15e-08
***
OccupationProtective-serv           6.192e-01  1.301e-01   4.760 1.94e-06
***
OccupationSales                     2.797e-01  8.618e-02   3.246  0.00117
**
OccupationTech-support              6.444e-01  1.157e-01   5.568 2.57e-08
***
OccupationTransport-moving         -1.082e-01  1.031e-01  -1.050  0.29391
RelationshipNot-in-family           6.761e-01  3.016e-01   2.242  0.02499
*
RelationshipOther-relative         -4.031e-01  2.920e-01  -1.381  0.16742
RelationshipOwn-child              -5.877e-01  2.935e-01  -2.002  0.04525
*
RelationshipUnmarried               5.744e-01  3.168e-01   1.813  0.06984
.
RelationshipWife                    1.332e+00  1.103e-01  12.076  < 2e-16
***
GenderMale                          8.747e-01  8.401e-02  10.412  < 2e-16
***
CapitalGain                         3.184e-04  1.106e-05  28.784  < 2e-16
***
CapitalLoss                         6.509e-04  3.998e-05  16.281  < 2e-16
***
HoursWork                           2.968e-02  1.774e-03  16.735  < 2e-16
***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 31005  on 27244  degrees of freedom
Residual deviance: 17992  on 27200  degrees of freedom
AIC: 18082

Number of Fisher Scoring iterations: 7
```

In [25]:

```
# Making prediction using train data and view the statistics
predict.label.sw <- predict(sw.fit, train, type="response")

# Only run the below if you have labels, in your submission, this must be UNCOMMENTED
# Truth label from train data
truth.label <- train$Salary

# view statistics
mod.stat(predict.label.sw, truth.label)
```

$accuracy
0.8451
$precision
0.7396
$recall
0.6104
$fscore
0.6688

Answer

The prunned GLM model has almost similar values of Accuracy, Precision, Recall and Fscore. This means the prunned model is as good as the original GLM model. One other thing we can infer from this is that all the predictors of the originl model are important as we can see them in the prunned model as well.

The old model had a accuracy of 84.52% while the prunned model has an accuracy of 84.51% which is slight a decrease.

The old model had a precision of 73.95% while the prunned model has an precision of 73.96% which is slight a increase. This means the number of True positives increased or the False positives decreased.

The old model had a recall of 61.07% while the prunned model has a recall of 61.04% which is slight a decrease. This means the number of True positives increased or the False negatives decreased.

The old model had a fscore of 66.9% while the prunned model has a fscore of 66.88% which is slight a decrease.

Question 4 (Libraries are allowed) (25 Marks)

Similar to the first part, to simulate for a realistic modelling process, this question will be in the form of a competition among students to find out who has the best model.

Thus, You will be graded by the performance of your model compared to your classmates', the better your model, the higher your score. Additionally, you need to write a short paragraph describing/documenting your thought process in this model building process `(300 words)` . Note that this is to explain to us why you build your current model so that we can verify that you understand the model you build and not just copy from other people.

Note Please make sure that we can install the libraries that you use in this part, the code structure can be:

```
install.packages("some package", repos='http://cran.us.r-project.org')
```

```
library("some package")
```

Remember that if we cannot run your code, we will have to give you a deduction, our suggestion is for you to use the standard `R version 3.6.1`

You also need to name your final model `fin.mod` so we can run a check to find out your performance. A good test for your understanding would be to set the previous **BIC model** to be the final model to check if your code works perfectly.

20 Marks for the model performance in the competition

5 Marks for logically writing down the thought process in building the final model

This is the link (https://www.kaggle.com/t/1bdebc96607742dbaf47ab36cd3ae421) to the competition

YOUR ANSWER HERE

ANSWER

The main objective of this task is to build a model that most appropriately classifies the Salary of a person to be greater than 50K. The outcome of our model is Binary, which mean its a Classification problem.

To start with the Model building process, we first use the Generalized Linear Model which we prunned using BIC penalty. Using this model as the starting point, we run the predictions. This gives a mean accuracy value of around 0.8498.

To improve our classification model, we try out different classification Algorithms to have a better model that most appropriately classifies the person salary.

Following are the different algorithms I have tried for this Classification task with the respective approximate mean F Score values obatined:

1. Generalised linear model (Prunned) - 0.84984
2. Linear Discriminant Analysis (LDA) - 0.84397
3. Classification and Regression Trees - 0.81193
4. kNN - 0.78429
5. Boosted regression trees - 0.88334
6. Random Forest - 0.8362

Looking at the above mean Accuracy values, its best to use Boosted regression trees for this Classification task. Boosting is a iteractive learning way to improve the performance of Supervised learning algorithms in which the weight of the observation is based on the last classification. In this technique, the weight of incorreclty data is increased. The main goal is to learn from the error of the last tree and increase the accuracy. This technique is good as it removes the efforts of cleaning the dataset and learns the underlying non linear relationships between predictors.

To have a robust classification model, we carry out the sampling process 10 times as specified in the `trainControl()` .

In [36]:

```
## install required packages
install.packages("xgboost")
install.packages("caret")
```

```
Installing package into 'C:/Users/prash/OneDrive/Documents/R/win-library/3.
6'
(as 'lib' is unspecified)
Warning message:
"package 'xgboost' is in use and will not be installed"Installing package in
to 'C:/Users/prash/OneDrive/Documents/R/win-library/3.6'
(as 'lib' is unspecified)
Warning message:
"package 'caret' is in use and will not be installed"
```

In [26]:

```
# load the libraries
library(caret)
library(xgboost)
```

```
Warning message:
"package 'xgboost' was built under R version 3.6.3"
```

In [*]:

```
# convert the target variable as Factor
train$Salary = as.factor(train$Salary)

# Fit the model on the training set
set.seed(123)
fin.mod <- train(Salary ~., data = train, method = "xgbTree",trControl = trainControl("cv",
```

In [53]:

```
pred.label <- predict(fin.mod, test)
```

In [55]:

```
# PLEASE DO NOT ALTER THIS CODE BLOCK
# Use this csv file to commit to the leaderboard
write.csv(data.frame("RowIndex" = seq(1, length(pred.label)), "Prediction" = pred.label),
          "ClassPredictLabel.csv", row.names = F)
```

In [ ]:

```
## PLEASE DO NOT ALTER THIS CODE BLOCK
## Please skip (don't run) this if you are a student
## For teaching team use only
source("../data/modassess.r")
model.perf <- mod.stat.test(pred.label,label$Label)
print(model.perf)
```

References

Brownlee, J. (2020). How to Calculate Precision, Recall, and F-Measure for Imbalanced Classification. Retrieved 9 November 2020, from https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/ (https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/)

A complete guide to the random forest algorithm. (2020). Retrieved 9 November 2020, from https://builtin.com/data-science/random-forest-algorithm (https://builtin.com/data-science/random-forest-algorithm)

Classification Algorithms | Types of Classification Algorithms | Edureka. (2020). Retrieved 9 November 2020, from https://www.edureka.co/blog/classification-algorithms/ (https://www.edureka.co/blog/classification-algorithms/)

Garg, R. (2020). A Primer to Ensemble Learning – Bagging and Boosting. Retrieved 9 November 2020, from https://analyticsindiamag.com/primer-ensemble-learning-bagging-boosting/ (https://analyticsindiamag.com/primer-ensemble-learning-bagging-boosting/)