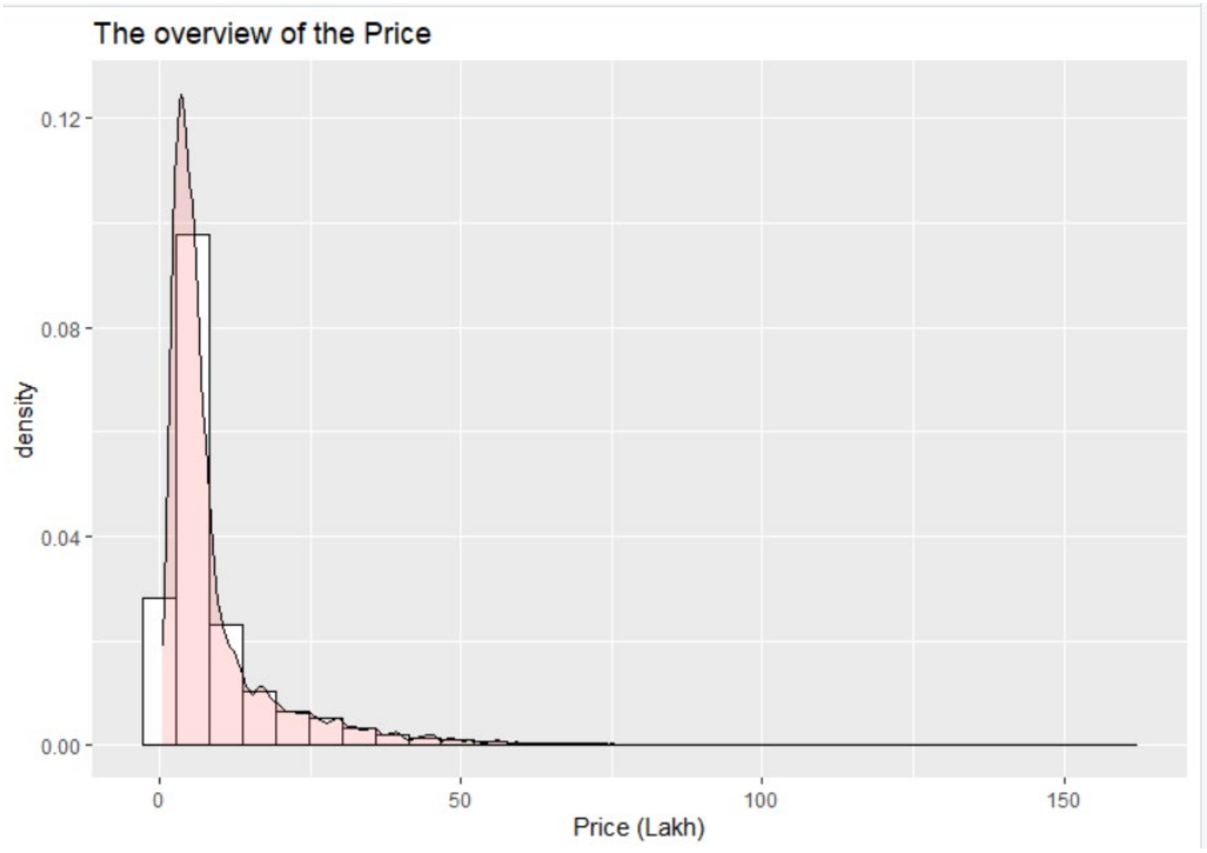
Introduction

This project is about the “Used cars prize prediction” The data set contains 6019 rows and 14 columns which including parameter, Name, Location, Year, Kilometers_Driven, Fuel_Type, Transmission, Owner_Type, Mileage, Engine_Power, Seats, New_Price and Price.

Our goal is to find the relative variable, use different model to do the price prediction, then compare different model to find out the most reasonable approach, then optimizethe model, test the accuracy and finally do the price prediction. Up till now, we have learned about three different kinds of model, KNN model, logistic regression model and random forest model. In this report, the work separate into two parts, first, do the exploration about the data set and second, use these three different method to build the model and test their accuracy.

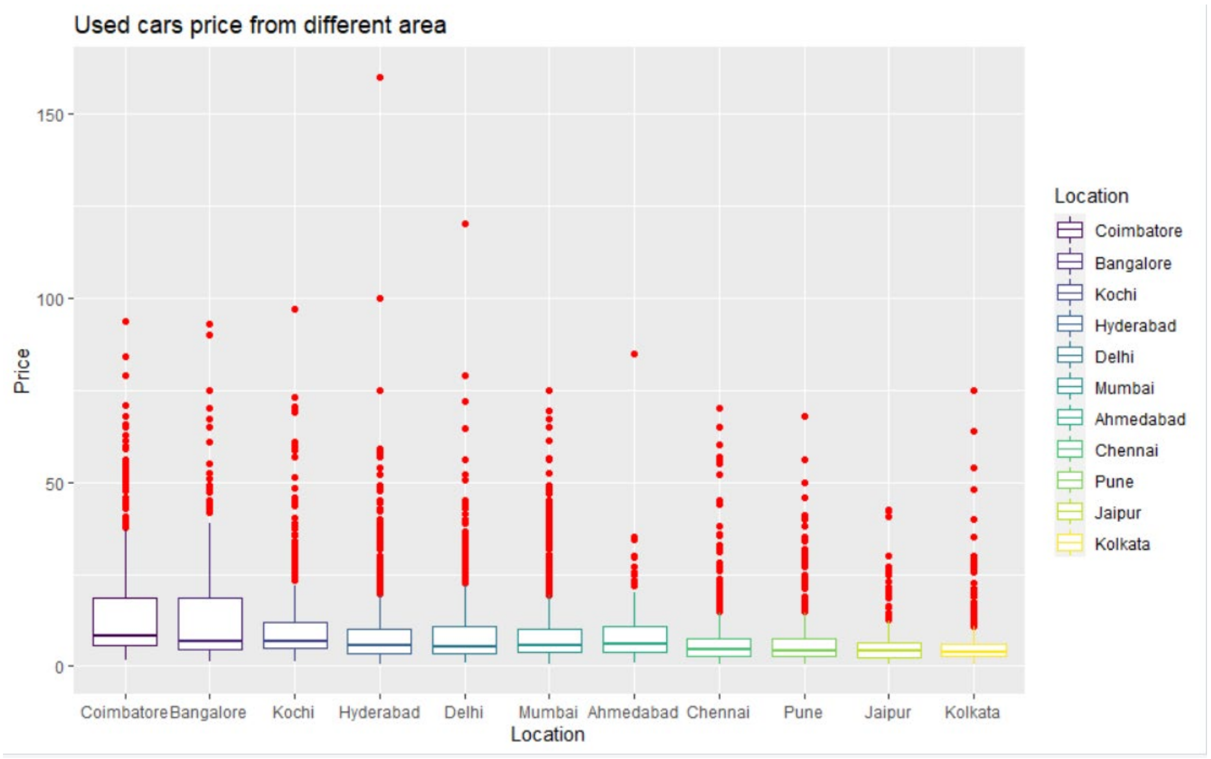
Data exploration

First of all, since our subject is about the used cars prize, we should browse the price distribution



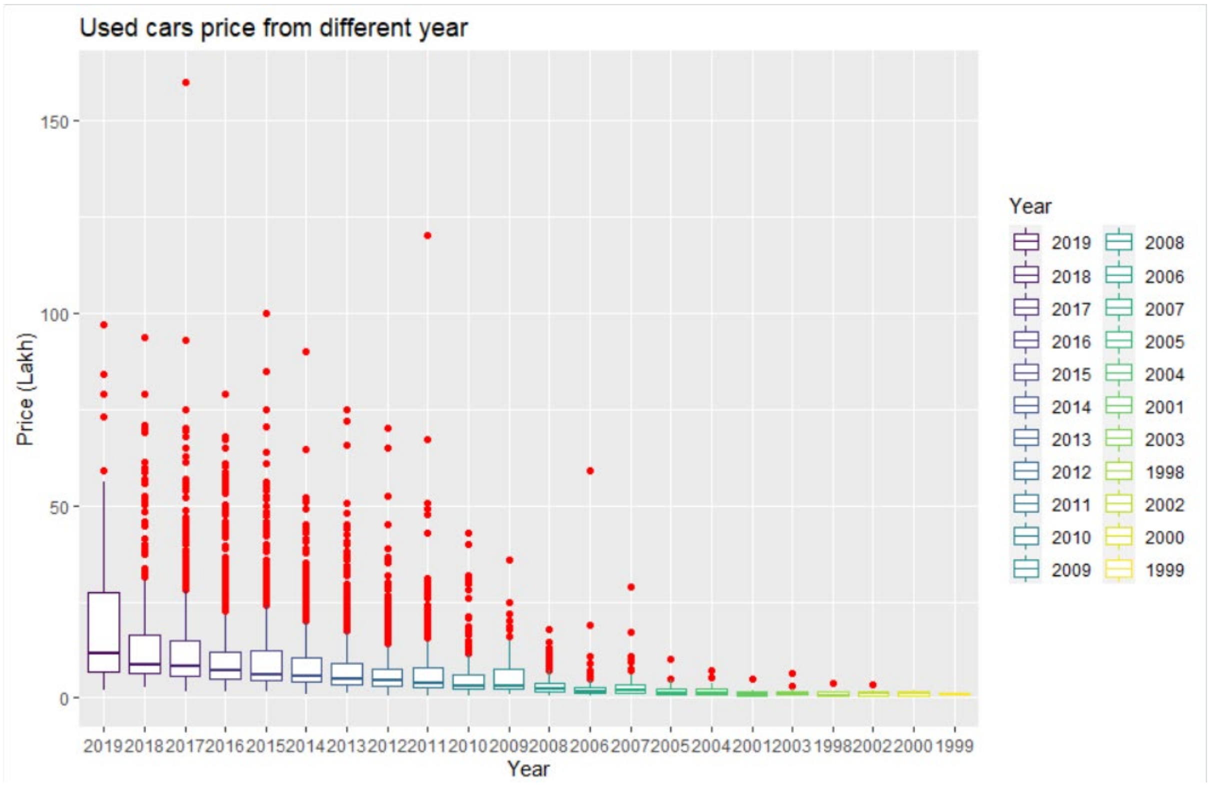
As we can see from the plot, the used car price mostly arrange from 0 to 50 (Lakh), and concentrate on the range from 5 to 15 (Lakh), New few plots we are going to show the relationship between price and each variables.

Price vs. Location



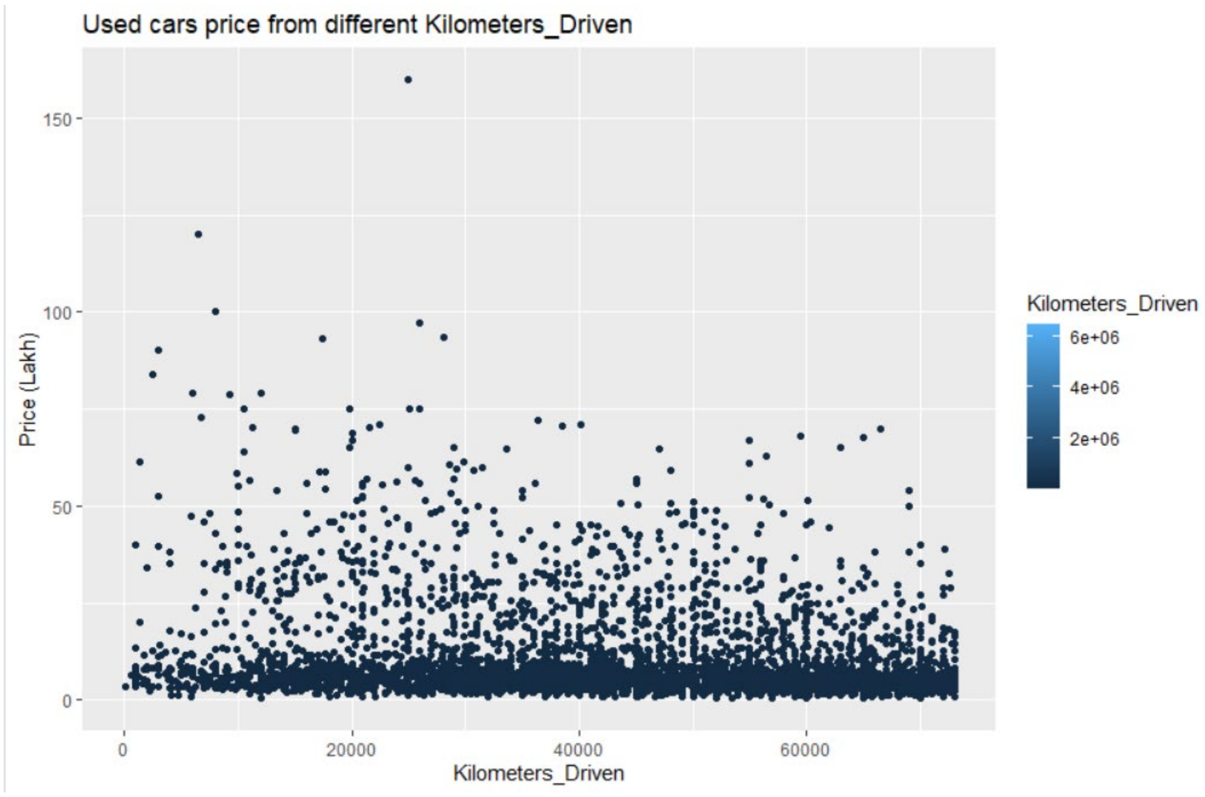
This box-plot is sort by the mean price of different location, as we can see, used cars from the Coimbatore has the highest price and from the Kolkata has the lowest price. Which is out of our expectation, since the Kolkata is the capital of the Indian state of West Bengal, we thought its used cars price might be higher than Coimbatore which is famous of its car industries.

Price vs. Year



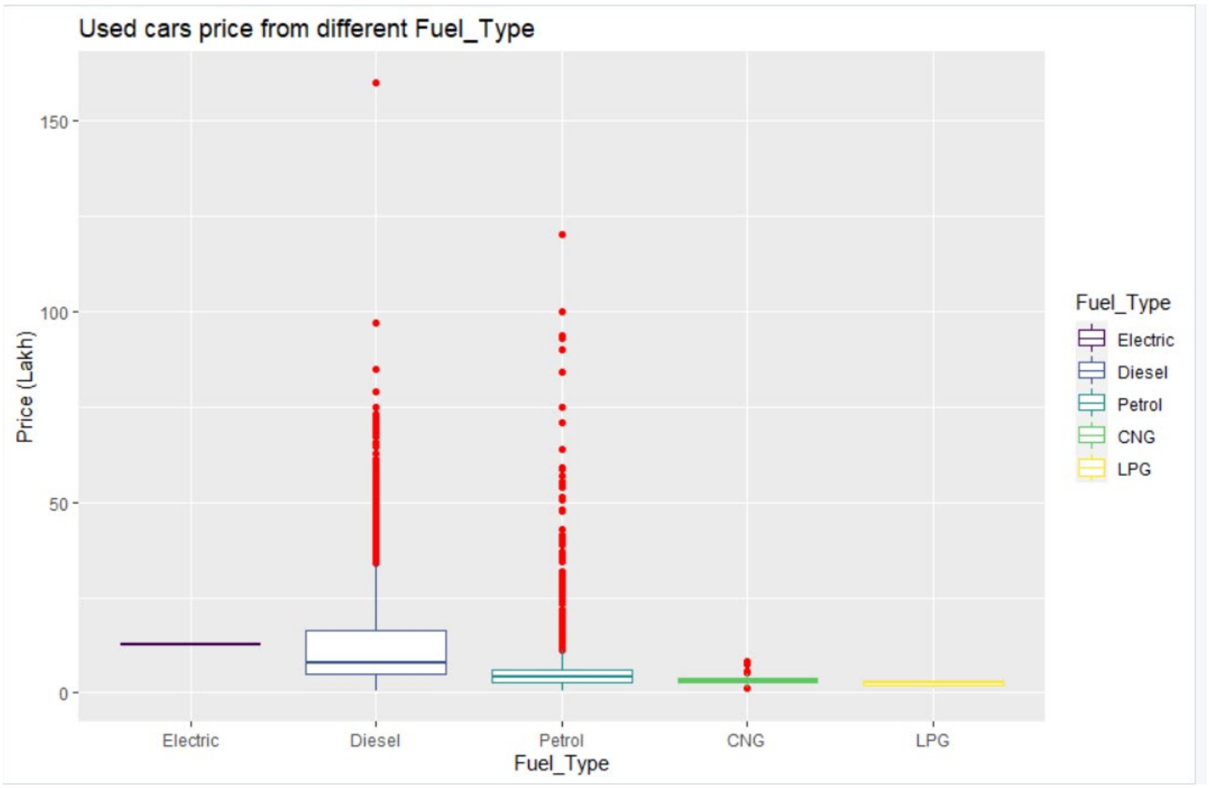
The plot of Price vs. Year shows that the used cars price is lower when it is older. Truly, it is the same as our expectation, even there are few of outliers have a higher price, might because of their different brand or other parameter, But as a complete unit, the used cars price goes down year by year.

Price vs. Kilometers_Driven



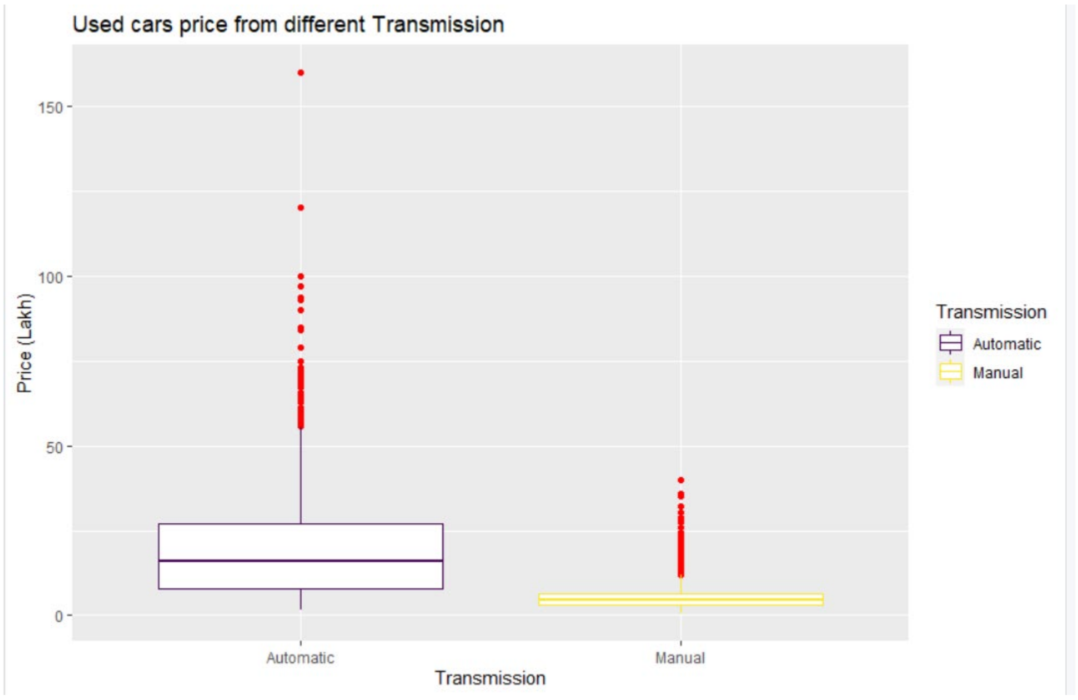
In order to have a clearer observation, the plot of the Price vs. Kilometers_Driven, we limited the price by the top boundary of 0.75 quantile of the Price. So that, as we can see on the plot, almost all levels of Kilometers_Driven have different price and we cannot see the main difference between each other. And then we conclude that the Kilometers_Driven maybe don't have a highly correlation with the prize. Though as a common sense, the used cars price should be lower when it has a higher Kilometers_Driven, in the plot it doesn't show that trend. However, we still need find more stable evidence to prove that when we build the model.

Price vs Fuel_Type



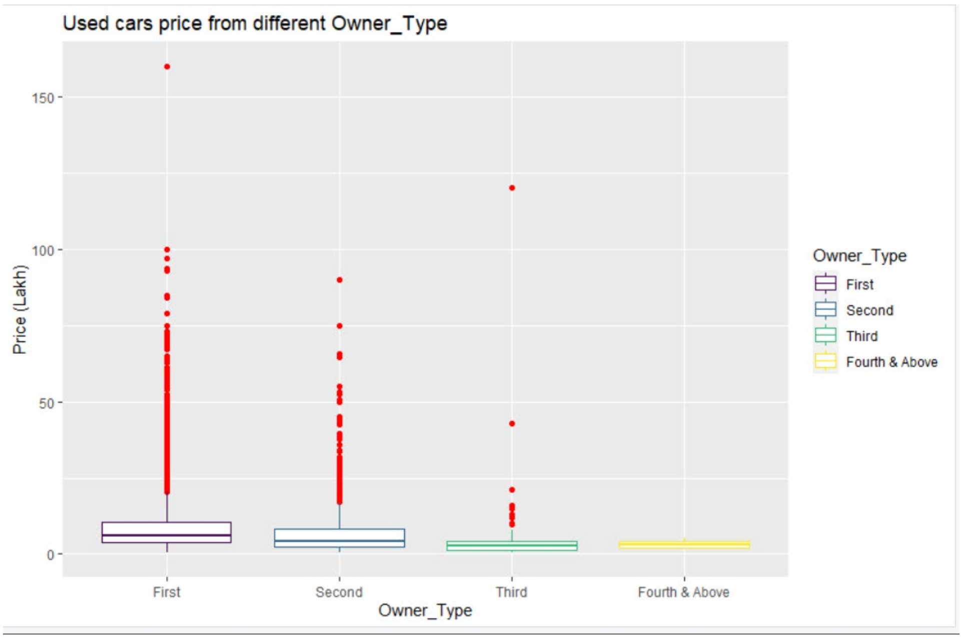
For the plot, the average price of electric cars have the highest price and the LPG have the lowest price. And the Diesel cars have a higher price than Petrol cars which is the same situation when cars are new.

Price vs, Transmission



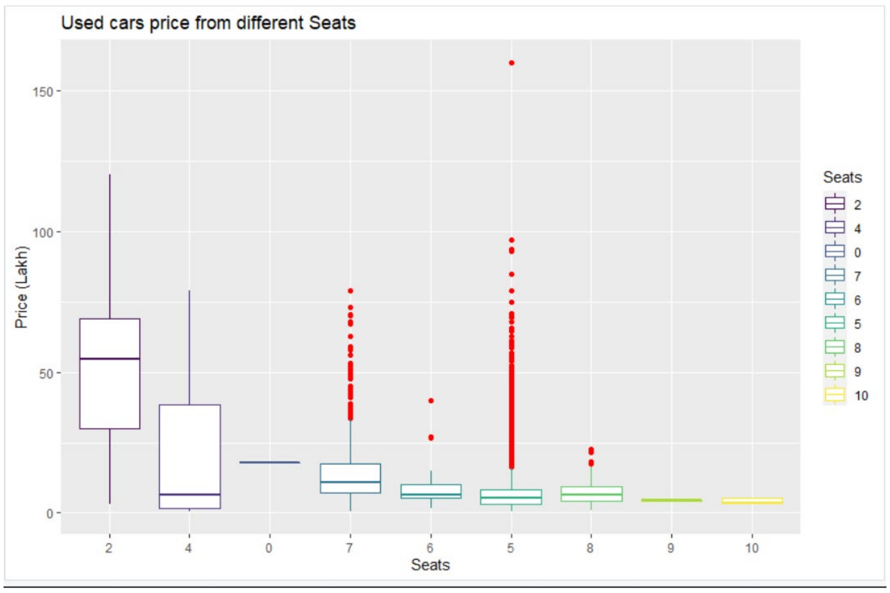
We can easily conclude from the plot, there is no doubt that the automatic used cars have a higher price than the manual used cars. Since new automatic cars are expensive than manual used cars.

Price vs. Owner_Type



Just as the plot shows that, first-hand cars have the highest price, and with the raise of transaction number, the price becomes lower and lower.

Price vs. Seats

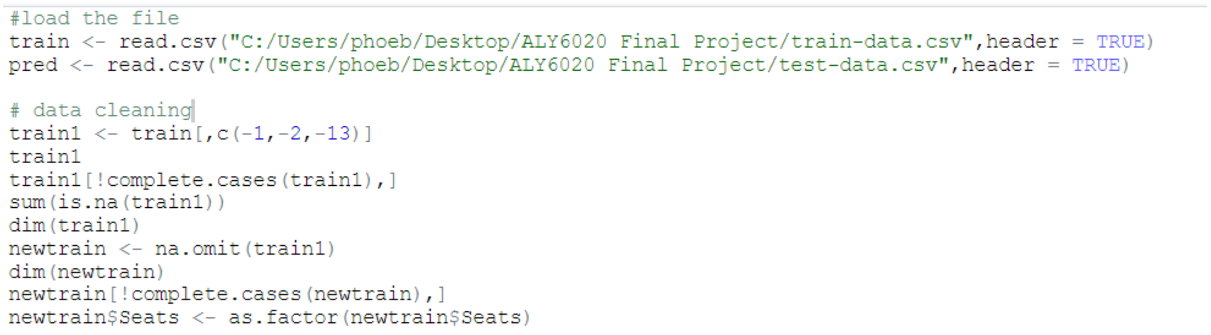


The data from the plot shows that the used cars have two seats have the highest price. Commonly the cars have two seats are sportscar, usually, these cars have a much higher price than normal cars, which is not surprise that used two-seats cars have highest price.

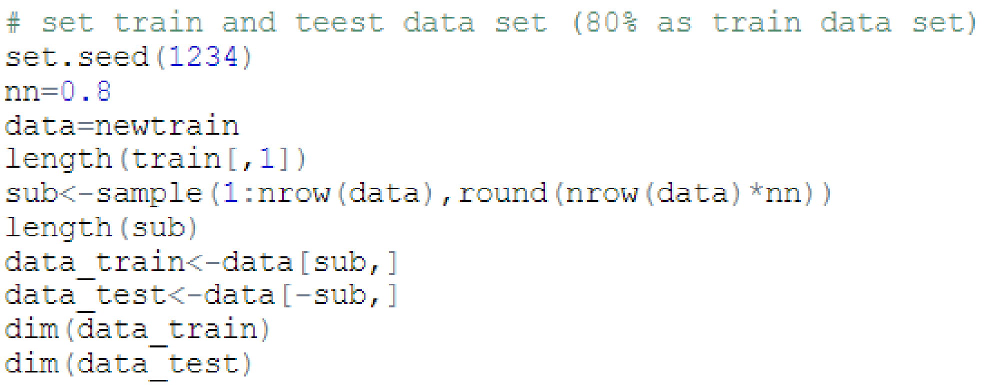
Since we have browsed the basic information about the price and each parameter, next steps, we trying to use different methods to build our prediction model to find out which one is the best way to predict the used cars price.

Model

Before we build the model we need do some preparation, loading the file, data cleaning (delete the missing value),delete unrelative column ,serial number, used cars name, and the new price of the cars(only few rows have new price) and then set up the train and test data set.

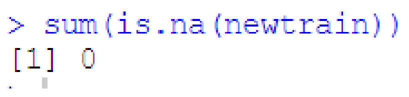


```
#load the file
train <- read.csv("C:/Users/phoeb/Desktop/ALY6020 Final Project/train-data.csv",header = TRUE)
pred <- read.csv("C:/Users/phoeb/Desktop/ALY6020 Final Project/test-data.csv",header = TRUE)

# data cleaning
train1 <- train[,c(-1,-2,-13)]
train1
train1[!complete.cases(train1),]
sum(is.na(train1))
dim(train1)
newtrain <- na.omit(train1)
dim(newtrain)
newtrain[!complete.cases(newtrain),]
newtrain$Seats <- as.factor(newtrain$Seats)
```



```
# set train and teest data set (80% as train data set)
set.seed(1234)
nn=0.8
data=newtrain
length(train[,1])
sub<-sample(1:nrow(data),round(nrow(data)*nn))
length(sub)
data_train<-data[sub,]
data_test<-data[-sub,]
dim(data_train)
dim(data_test)
```

Result:

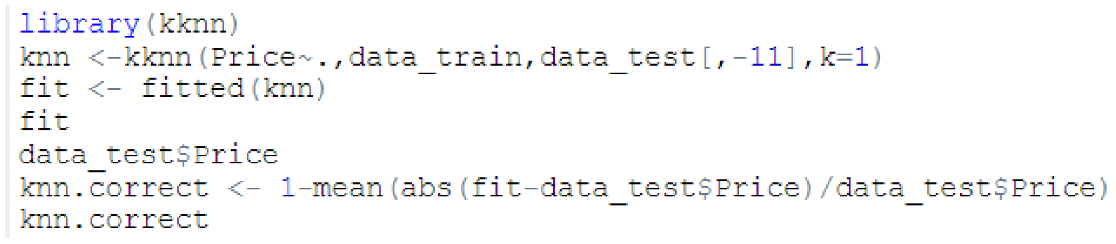


```
> sum(is.na(newtrain))
[1] 0
```

KNN(K-Nearest neighbor:

The first method we choose is KNN algorithm. The KNN algorithm assumes that similar things exist in close proximity, which means the similar things can be classified in one class should be near to each other. *KNN is a model that classifies data points based on the points that are most similar to it. It uses test data to make an "educated guess" on what an unclassified point should be classified as.(Madison).*

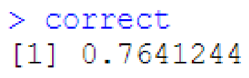
R code:



```r
library(kknn)
knn <-kknn(Price~.,data_train,data_test[,-11],k=1)
fit <- fitted(knn)
fit
data_test$Price
knn.correct <- 1-mean(abs(fit-data_test$Price)/data_test$Price)
knn.correct
```

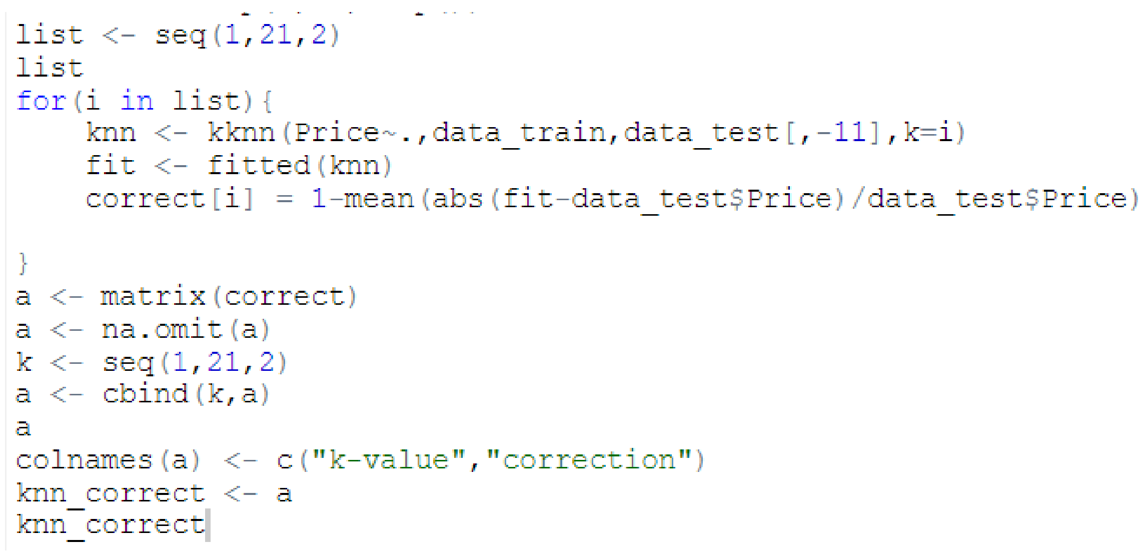
The formula we use is Price ~. which means all the other variables are included in the formula. And because the result of the prediction is value instead of class so we use deviation to show the correction.

For the first knn model, the k-value we choose K=1 and the result of that is 0.764



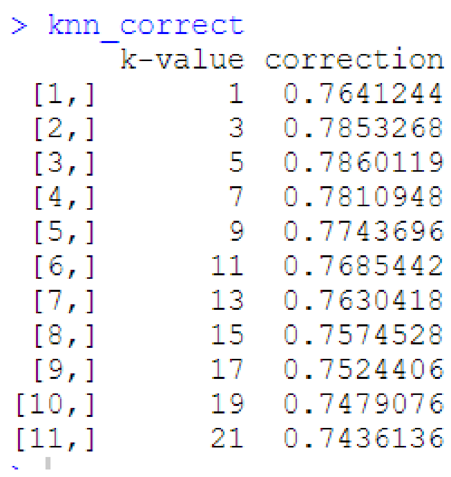
```
> correct
[1] 0.7641244
```

Then we change the K-value to build other knn models. Because the principle of the KNN algorithm, we believe K-value should be odd number, if K-value is even number it might meet two neighbors have the same distance. In case we change K-value from 1 to 21 and step 2



```
list <- seq(1,21,2)
list
for(i in list){
    knn <- kknn(Price~.,data_train,data_test[,-11],k=i)
    fit <- fitted(knn)
    correct[i] = 1-mean(abs(fit-data_test$Price)/data_test$Price)

}
a <- matrix(correct)
a <- na.omit(a)
k <- seq(1,21,2)
a <- cbind(k,a)
a
colnames(a) <- c("k-value","correction")
knn_correct <- a
knn_correct
```

Result:



```
> knn_correct
      k-value correction
 [1,]       1  0.7641244
 [2,]       3  0.7853268
 [3,]       5  0.7860119
 [4,]       7  0.7810948
 [5,]       9  0.7743696
 [6,]      11  0.7685442
 [7,]      13  0.7630418
 [8,]      15  0.7574528
 [9,]      17  0.7524406
[10,]      19  0.7479076
[11,]      21  0.7436136
```

As we can see when K-value equals 5, the model has the highest correction. However the accuracy is only 0.786, is under our standard, we expect that the model can have at least 0.8 accuracy. So that we try other approaches.

Logistic Regression

The main function of logistic regression model is to classify two different class. So that to fit the model, we set up a new class, we calculate the mean of the price, and let the price low than the average as the “low” level and the price higher than the average as the “high” level.

```
log <- newtrain
mean <- mean(log$Price)
log$Price[which(log$Price < mean)] <- "low"
log$Price <- as.numeric(log$Price)
log$Price[which(log$Price > mean)] <- "high"
log$Price[is.na(log$Price)] <- "low"
log$Price <- as.factor(log$Price)
log$Price

set.seed(1234)
nn=0.8
data=log
length(log[,1])
sub<-sample(1:nrow(data),round(nrow(data)*nn))
length(sub)
log_train<-data[sub,]
log_test<-data[-sub,]
dim(log_train)
dim(log_test)
str(log_train)
```

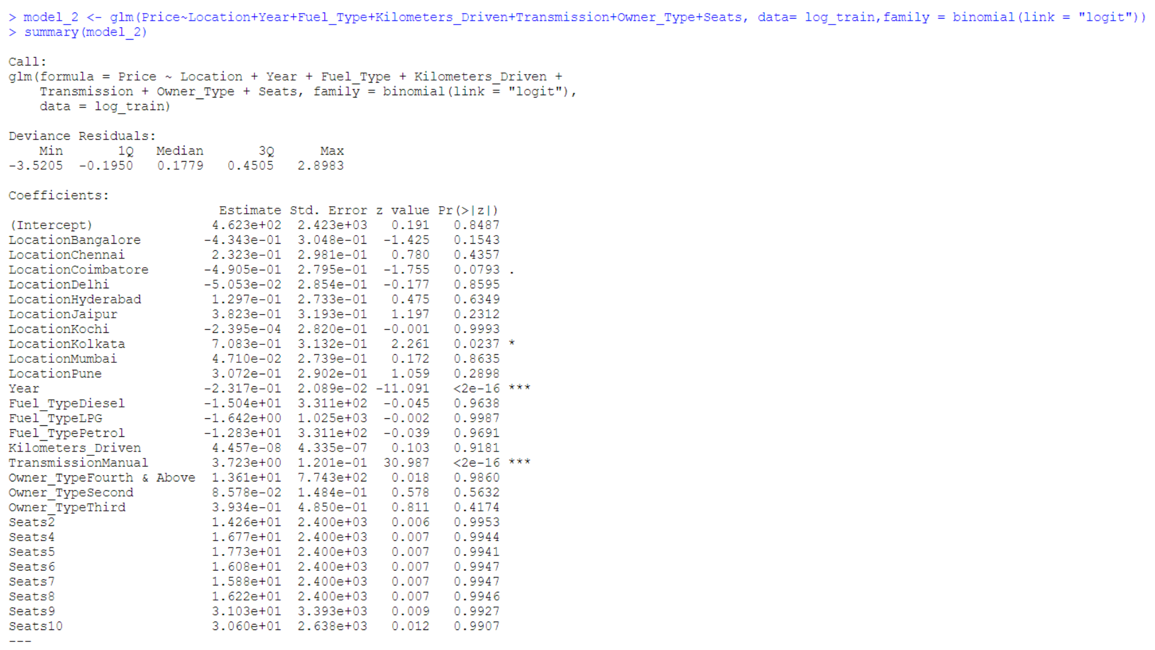
Using the new training data set, we set up the model as following:

```
model_1 <- glm(Price~.,data = log_train,family = binomial(link = "logit"))
summary(model_1)
```

The result shows that there are too many irrelevant variables in the model, so we change our formula to build model_2

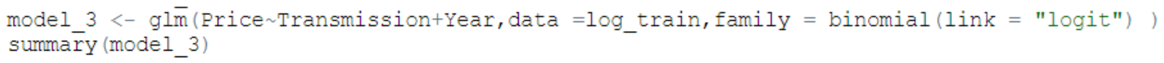
```
model_2 <- glm(Price~Location+Year+Fuel_Type+Kilometers_Driven+Transmission+Owner_Type+Seats, data= log_train,family = binomial(link = "logit"))
summary(model_2)
```

Result:



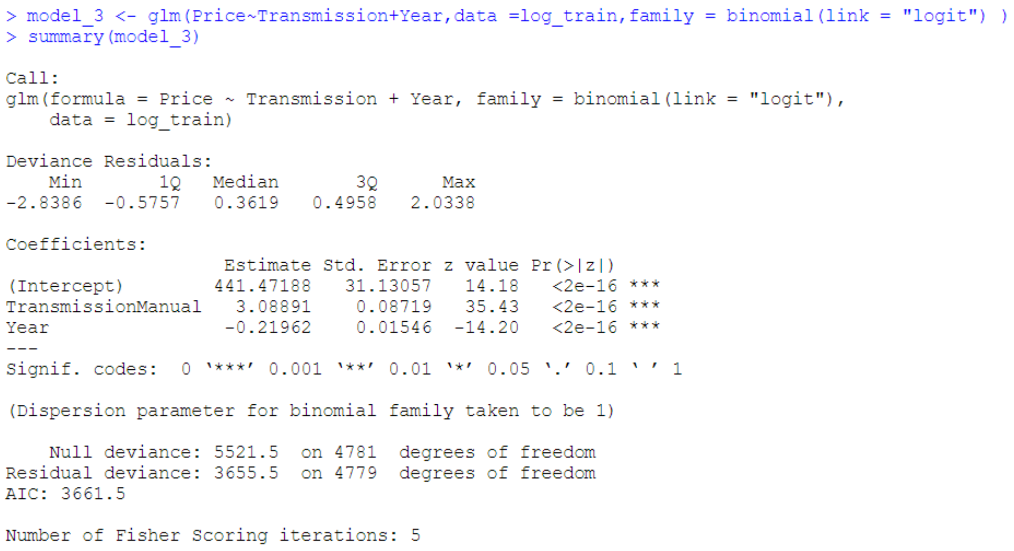
```
> model_2 <- glm(Price~Location+Year+Fuel_Type+Kilometers_Driven+Transmission+Owner_Type+Seats, data= log_train,family = binomial(link = "logit"))
> summary(model_2)

Call:
glm(formula = Price ~ Location + Year + Fuel_Type + Kilometers_Driven +
    Transmission + Owner_Type + Seats, family = binomial(link = "logit"),
    data = log_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.5205  -0.1950   0.1779   0.4505   2.8983

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)               4.623e+02  2.423e+03   0.191   0.8487
LocationBangalore        -4.343e-01  3.048e-01  -1.425   0.1543
LocationChennai           2.323e-01  2.981e-01   0.780   0.4357
LocationCoimbatore       -4.905e-01  2.795e-01  -1.755   0.0793 .
LocationDelhi            -5.053e-02  2.854e-01  -0.177   0.8595
LocationHyderabad         1.297e-01  2.733e-01   0.475   0.6349
LocationJaipur            3.823e-01  3.193e-01   1.197   0.2312
LocationKochi            -2.395e-04  2.820e-01  -0.001   0.9993
LocationKolkata           7.083e-01  3.132e-01   2.261   0.0237 *
LocationMumbai            4.710e-02  2.739e-01   0.172   0.8635
LocationPune              3.072e-01  2.902e-01   1.059   0.2898
Year                     -2.317e-01  2.089e-02 -11.091   <2e-16 ***
Fuel_TypeDiesel          -1.504e+01  3.311e+02  -0.045   0.9638
Fuel_TypeLPG             -1.642e+00  1.025e+03  -0.002   0.9987
Fuel_TypePetrol          -1.283e+01  3.311e+02  -0.039   0.9691
Kilometers_Driven         4.457e-08  4.335e-07   0.103   0.9181
TransmissionManual        3.723e+00  1.201e-01  30.987   <2e-16 ***
Owner_TypeFourth & Above  1.361e+01  7.743e+02   0.018   0.9860
Owner_TypeSecond          8.578e-02  1.484e-01   0.578   0.5632
Owner_TypeThird           3.934e-01  4.850e-01   0.811   0.4174
Seats2                    1.426e+01  2.400e+03   0.006   0.9953
Seats4                    1.677e+01  2.400e+03   0.007   0.9944
Seats5                    1.773e+01  2.400e+03   0.007   0.9941
Seats6                    1.608e+01  2.400e+03   0.007   0.9947
Seats7                    1.588e+01  2.400e+03   0.007   0.9947
Seats8                    1.622e+01  2.400e+03   0.007   0.9946
Seats9                    3.103e+01  3.393e+03   0.009   0.9927
Seats10                   3.060e+01  2.638e+03   0.012   0.9907
---
```

As we can see from the result, some variables like Location, Kilmeters_Driven, Seats, these variables have a high p-value which is over 0.05, so we need to optimize our model again.



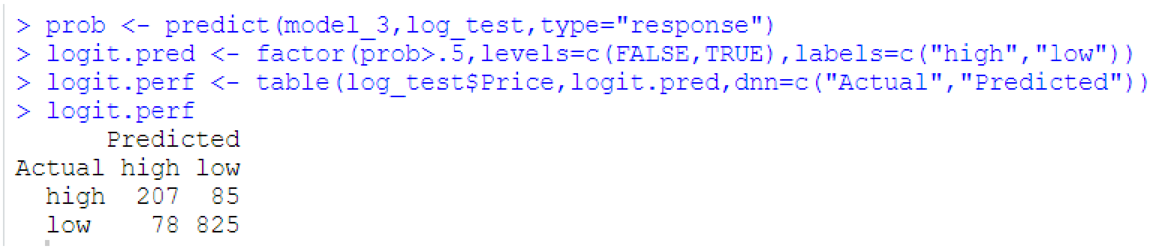
```
model_3 <- glm(Price~Transmission+Year,data =log_train,family = binomial(link = "logit") )
summary(model_3)
```

Results:



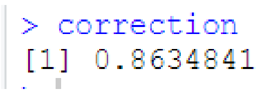
```
> model_3 <- glm(Price~Transmission+Year,data =log_train,family = binomial(link = "logit") )
> summary(model_3)

Call:
glm(formula = Price ~ Transmission + Year, family = binomial(link = "logit"),
    data = log_train)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.8386  -0.5757   0.3619   0.4958   2.0338  

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)    
(Intercept)       441.47188   31.13057   14.18   <2e-16 ***
TransmissionManual   3.08891    0.08719   35.43   <2e-16 ***
Year                -0.21962    0.01546  -14.20   <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: 5521.5  on 4781  degrees of freedom
Residual deviance: 3655.5  on 4779  degrees of freedom
AIC: 3661.5

Number of Fisher Scoring iterations: 5
```

Finally, the model is acceptable, and the parameters are significant. Using this model to do the prediction:



```
> prob <- predict(model_3,log_test,type="response")
> logit.pred <- factor(prob>.5,levels=c(FALSE,TRUE),labels=c("high","low"))
> logit.perf <- table(log_test$Price,logit.pred,dnn=c("Actual","Predicted"))
> logit.perf
      Predicted
Actual high low
  high  207  85
  low    78 825
```

And with the simple calculation we find out the correction is around 0.863

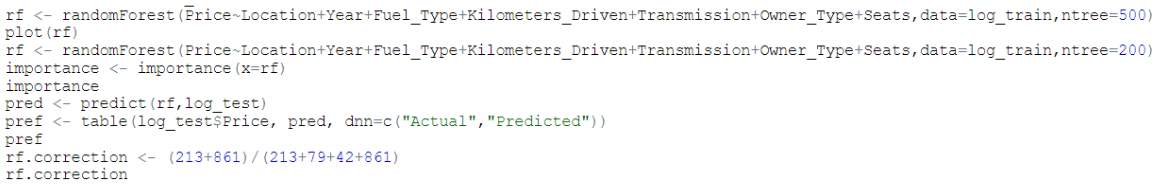


```
> correction
[1] 0.8634841
```

Which is satisfied with our expectation.

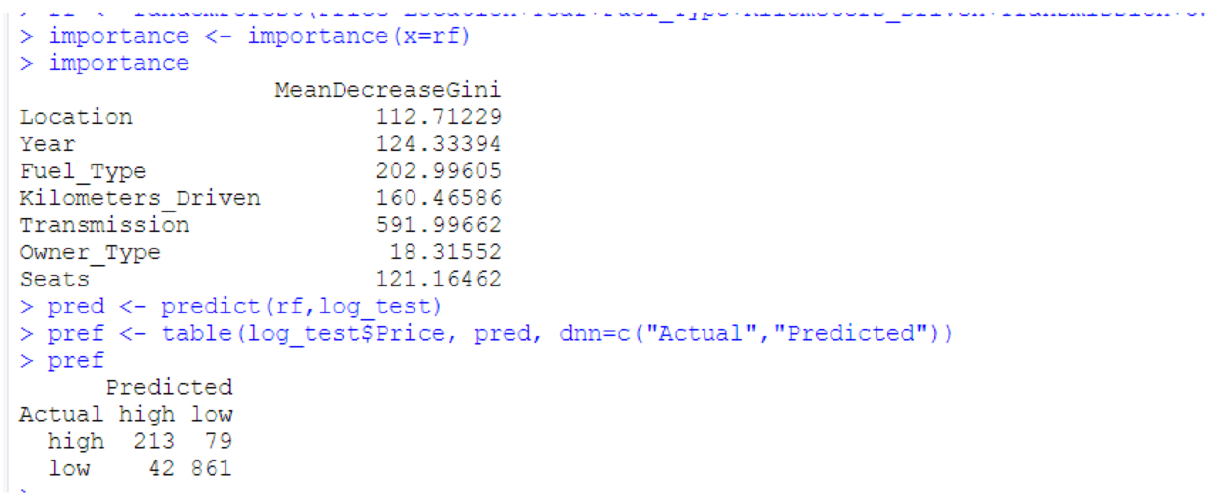
Random forest

First of all, we use the data set contains Price as factor with the level “low” and “high” to build the model



```
rf <- randomForest(Price~Location+Year+Fuel_Type+Kilometers_Driven+Transmission+Owner_Type+Seats,data=log_train,ntree=500)
plot(rf)
rf <- randomForest(Price~Location+Year+Fuel_Type+Kilometers_Driven+Transmission+Owner_Type+Seats,data=log_train,ntree=200)
importance <- importance(x=rf)
importance
pred <- predict(rf,log_test)
pref <- table(log_test$Price, pred, dnn=c("Actual","Predicted"))
pref
rf.correction <- (213+861)/(213+79+42+861)
rf.correction
```

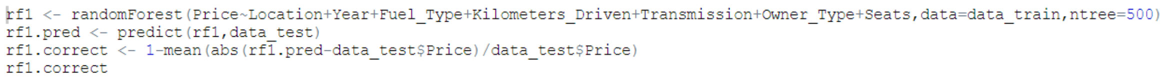
Result:



```
> importance <- importance(x=rf)
> importance
                  MeanDecreaseGini
Location                 112.71229
Year                     124.33394
Fuel_Type                202.99605
Kilometers_Driven        160.46586
Transmission             591.99662
Owner_Type                18.31552
Seats                    121.16462
> pred <- predict(rf,log_test)
> pref <- table(log_test$Price, pred, dnn=c("Actual","Predicted"))
> pref
      Predicted
Actual high low
  high  213  79
  low    42 861
```

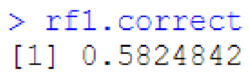
After calculation the correction of the random forest model is 0.898 which is higher than the logistic regression model.

And then we also build the model when price is numeric parameter.



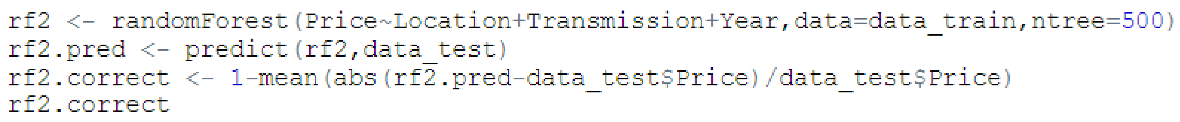
```
rf1 <- randomForest(Price~Location+Year+Fuel_Type+Kilometers_Driven+Transmission+Owner_Type+Seats,data=data_train,ntree=500)
rf1.pred <- predict(rf1,data_test)
rf1.correct <- 1-mean(abs(rf1.pred-data_test$Price)/data_test$Price)
rf1.correct
```

Result



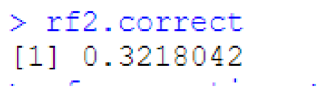
```
> rf1.correct
[1] 0.5824842
```

We believed that the low accuracy might because the irrelevant parameters, so we delete them and build a new model



```
rf2 <- randomForest(Price~Location+Transmission+Year,data=data_train,ntree=500)
rf2.pred <- predict(rf2,data_test)
rf2.correct <- 1-mean(abs(rf2.pred-data_test$Price)/data_test$Price)
rf2.correct
```

Result:



```
> rf2.correct
[1] 0.3218042
```

It becomes lower. There might be some problem when random forest model to analyze the numeric parameters. Or this project is not fit for the random forest model.

After we using three method to build the model, we find some problems and some of them we fixed it but there still exist some issues that might affect the final result of the project.

Problems:

- Multinomial regression model can not use predict function to comes out the numeric result, the error shows that the type of the predict function should be "class" or "probs" but we need "response" to find out the accuracy.
- How to optimize the random forest model? We need to find some solution that can solve the problem of optimize the random forest model.
- How to handle the outliers? From the overview of the data set, we find out that there are few parameters have outliers, what should we do? Delete them or replace them by the boundary?
- Some of the parameters might be related but since the meet some problems of data structure transformation not used in the model. Adding these parameter might improve the accuracy.
- Multiple class. The class of price is just separate into two class, using more detail classification by price range might make result more useful.

Insights:

- From the above work, the kNN model have the best performance in predicting the price, and the random forest have the higher accuracy in predicting the price level.
- We expect that the Kilometers_Driven is an irrelevant parameter of the price prediction and the results shows that it is correct.
- Using the deviation to show the accuracy of the prediction model is the main steps when building the model.

References

Madison Schott(2019) K-Nearest Neighbors (KNN) Algorithm for Machine Learning(Retrieved from https://medium.com/capital-one-tech/k-nearest-neighbors-knn-algorithm-for-machine-learning-e883219c8f26)

Aman Sharma(2020) ML from Scratch-Multinomial Logistic Regression (Retrieved from: https://towardsdatascience.com/ml-from-scratch-multinomial-logistic-regression-6dda9cbacf9d)

Saishruthi Swaminathan(2018) Logistic Regression — Detailed Overview (Retrieved from: https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc)