Capstone_Project [2]- edX HarvardX

Used Car Price Prediction

(AUDI Brand)

Abedulla M. A. El-Saidy

21/12/2021

Table of Contents

1 Overview ... 3
1.1 Introduction ... 3
1.2 Dataset ... 3
2 Methods and Analysis ... 3
2.1 Data import ... 3
2.2 Training and Validation Partition ... 4
5: Load Data from disc ... 4
2.3 Data charcterstics ... 4
2.4 Check na values ... 5
2.5 Data Exploration ... 5
2.5.1 Price per year ... 5
2.5.2 Price per model ... 7
2.5.3 Price per transmission ... 7
2.5.4 Price per milage ... 8
2.5.5 Price per fuelType ... 9
2.5.6 Price per tax ... 9
2.5.7 Price per mpg ... 10
2.5.8 Price per enginSize ... 10
2.5.9 Numeric parameters correlation ... 11
2.6 Modeling Methods ... 11
2.6.1 Model 1 - Base Line mean only ... 11
2.6.2 Linear models (lm) ... 11
2.6.3 Model 6 (Generalized Additive Model using Splines- gam) ... 14
2.6.4 Model 7 (Partial Least Squares-pls) -Predict price using all ... 14
2.6.5 Model 8 (randomForest) -Predict price using all ... 15
2.6.6 Model 9 (Decision Tree) -Predict price using all ... 16
2.6.7 Model 10 (k-Nearest Neighbors) -Predict price using all ... 16
2.6.8 Model 11 (Multivariate Adaptive Regression Spline- Earth) ... 17
2.6.9 Model 12 (Generalized Linear Model- glm) -Predict price using all ... 17
2.6.10 Model 13 (The lasso) -Predict price using all ... 17
3 Results ... 18
4 Conclusion ... 19
5 References ... 19

1 Overview

For this project “Capstone_Project [2]- edX HarvardX”, I will apply machine learning techniques that go beyond standard linear regression. I will use an available dataset “Audi Used Car” [1] to solve the problem of my choice. In this study, it is aimed to determine the best prediction model by using ML algorithms.

1.1 Introduction

Many countries have a high-volume second-hand car market. Today, used car sales given over the internet have accelerated this market even more. This situation has caused difficulties in determining the most suitable price for the vehicle to be bought or sold. The problem of determining the price of second-hand vehicles causes both buyers and sellers to have difficulties since it contains many variables. [2], the best model will be selected based on RMSE.

1.2 Dataset

Data was downloaded from KAGGLE, and have been separated into files corresponding to each car manufacturer. I chose just the data for Audi brand.

- The cleaned data set contains(Features) Information of:
  - Price £
  - Transmission
  - Mileage
  - Fuel type
  - Road tax
  - Miles per gallon (mpg)
  - Engine size

Data exploration and visualization will be presented in the next section.

2 Methods and Analysis

2.1 Data import

Import the data from disc location after downloading the file from KAGGLE [1].

```
audi_cars =read.csv(“~/R/z/pr2/pr2/Data/audi.csv”,sep = “,”)
```

2.2 Training and Validation Partition

Split the dataset into training and validation

1: row numbers

```
RowNum = createDataPartition(audi_cars$price, p=0.80, list=FALSE)
```

2: Create the training dataset

```
train_data <- audi_cars[RowNum,]
```

3: Create the test dataset

```
test_data <- audi_cars[-RowNum,]
```

4: Take a copy of the train and test data

```
write.csv(train_data,"~/R/z/pr2/Data/train_data.csv")
write.csv(test_data,"~/R/z/pr2/Data/test_data.csv")
```

5: Load Data from disc

```
audi_cars =read.csv("~/R/z/pr2/Data/audi.csv",sep = ",")
train_data =read.csv("~/R/z/pr2/Data/train_data.csv",sep = ",")
test_data =read.csv("~/R/z/pr2/Data/test_data.csv",sep = ",")
```

2.3 Data charcterstics

```
summary(audi_cars)

##    model               year          price         transmission
##  Length:10667       Min.   :1997   Min.   :  1490   Length:10667
##  Class :character   1st Qu.:2016   1st Qu.: 15120   Class :character
##  Mode  :character   Median :2017   Median : 20200   Mode  :character
##                     Mean   :2017   Mean   : 22896
##                     3rd Qu.:2019   3rd Qu.: 27990
##                     Max.   :2020   Max.   :145000
##     mileage         fuelType             tax           mpg
##  Min.   :     1   Length:10667       Min.   :  0   Min.   : 18.90
##  1st Qu.:  5964   Class :character   1st Qu.:125   1st Qu.: 40.90
##  Median : 19000   Mode  :character   Median :145   Median : 49.60
##  Mean   : 24824                      Mean   :126   Mean   : 50.77
##  3rd Qu.: 36462                      3rd Qu.:145   3rd Qu.: 58.90
##  Max.   :323000                      Max.   :580   Max.   :188.30
##    engineSize
##  Min.   :0.000
##  1st Qu.:1.500
##  Median :2.000
##  Mean   :1.931
##  3rd Qu.:2.000
##  Max.   :6.300
```

```
glimpse(audi_cars)

## Rows: 10,667
## Columns: 9
## $ model        <chr> " A1", " A6", " A1", " A4", " A3", " A1", " A6", " A4", "~
## $ year         <int> 2017, 2016, 2016, 2017, 2019, 2016, 2016, 2016, 2015, 201~
## $ price        <int> 12500, 16500, 11000, 16800, 17300, 13900, 13250, 11750, 1~
## $ transmission <chr> "Manual", "Automatic", "Manual", "Automatic", "Manual", "~
## $ mileage      <int> 15735, 36203, 29946, 25952, 1998, 32260, 76788, 75185, 46~
## $ fuelType     <chr> "Petrol", "Diesel", "Petrol", "Diesel", "Petrol", "Petrol~
## $ tax          <int> 150, 20, 30, 145, 145, 30, 30, 20, 20, 30, 145, 125, 145,~
## $ mpg          <dbl> 55.4, 64.2, 55.4, 67.3, 49.6, 58.9, 61.4, 70.6, 60.1, 55.~
## $ engineSize   <dbl> 1.4, 2.0, 1.4, 2.0, 1.0, 1.4, 2.0, 2.0, 1.4, 1.4, 1.4, 2.~

dim(audi_cars)

## [1] 10667     9
```

2.4 Check na values

```
sum(is.na(audi_cars))

## [1] 0
```

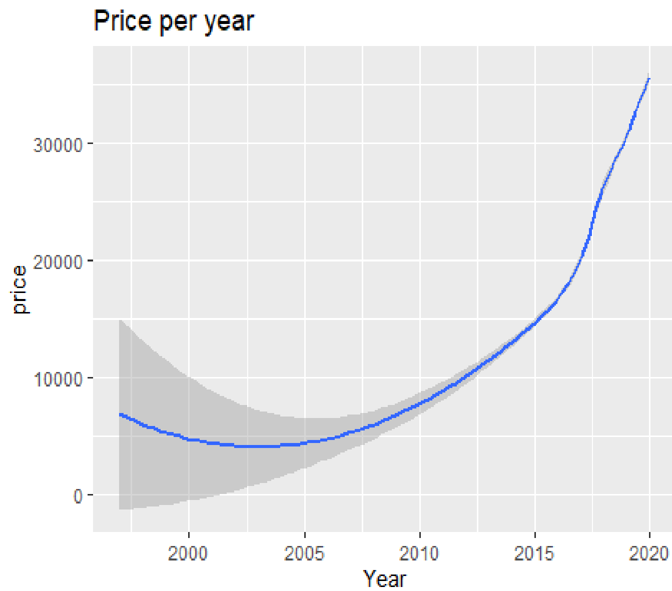
No NA values

2.5 Data Exploration

Visualizations to understand the data

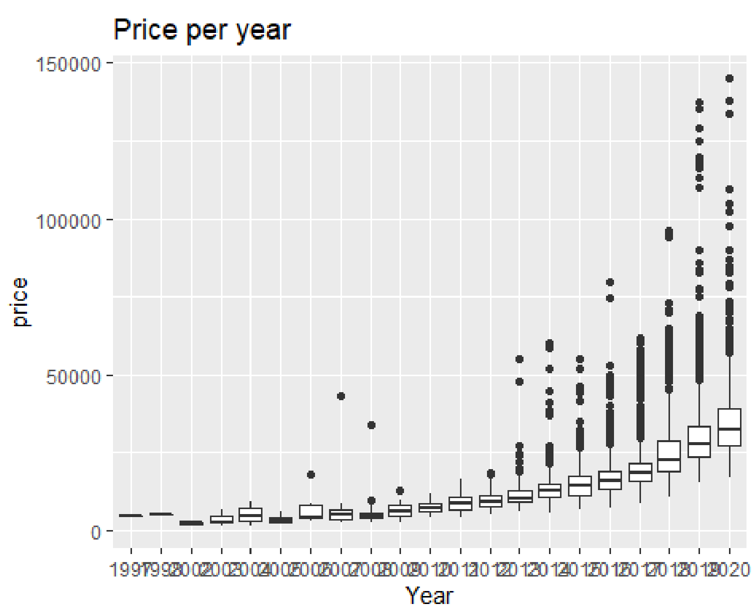
2.5.1 Price per year

```
audi_cars %>% mutate(Year = as.numeric(year)) %>%
  ggplot() +   geom_smooth(aes(x=Year, y=price),method="loess") +
  ggtitle("Price per year")

## `geom_smooth()` using formula 'y ~ x'
```



We can conclude the more the year, the more the price.

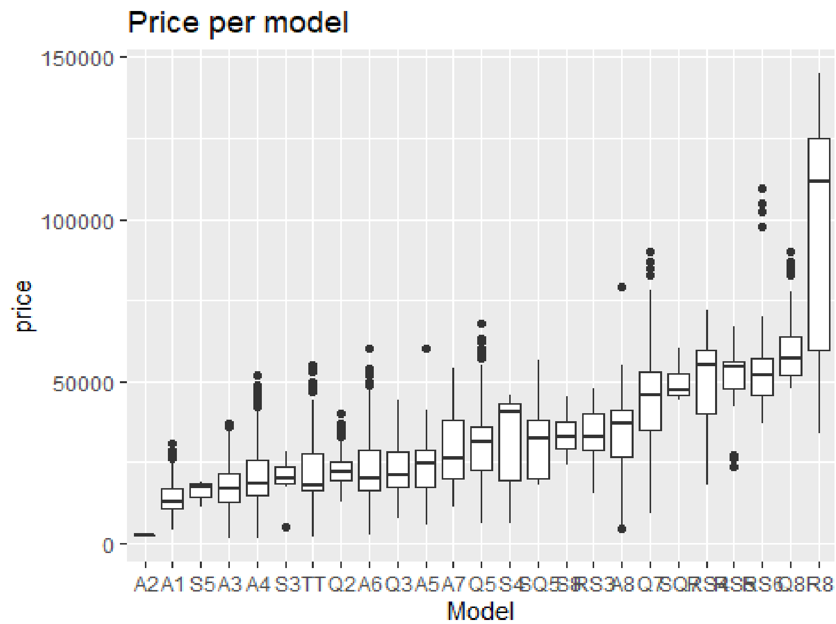
```
audi_cars %>% mutate(Year = as.factor(year)) %>%
  ggplot() +   geom_boxplot(aes(x=Year, y=price)) +
  ggtitle("Price per year")
```



We can conclude the more the year, the more the price.

2.5.2 Price per model

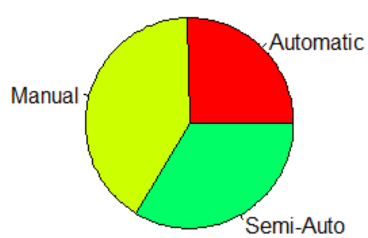
```
audi_cars %>% mutate(Model = (model)) %>%
 ggplot() +  geom_boxplot(aes(x=reorder(Model,price), y=((price)))) +
 labs(x="Model", y="price")+
 ggtitle("Price per model")
```



We can conclude that some models have higher price like R8

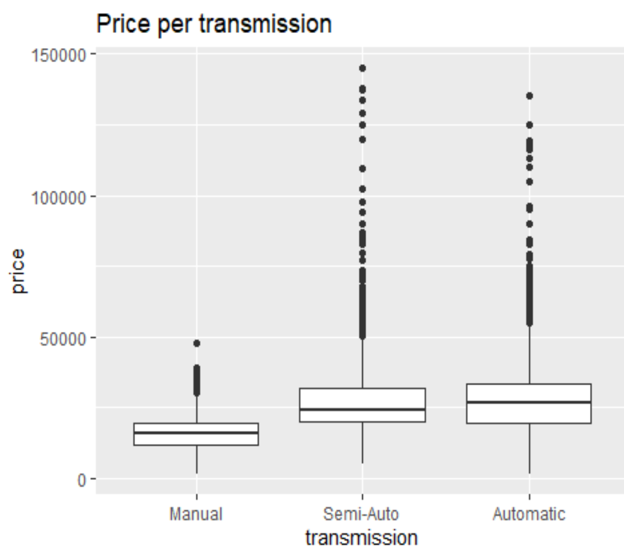
2.5.3 Price per transmission

```
pie( summary(as.factor(audi_cars$transmission)),col=rainbow(5))
```



Transmission is equally distributed

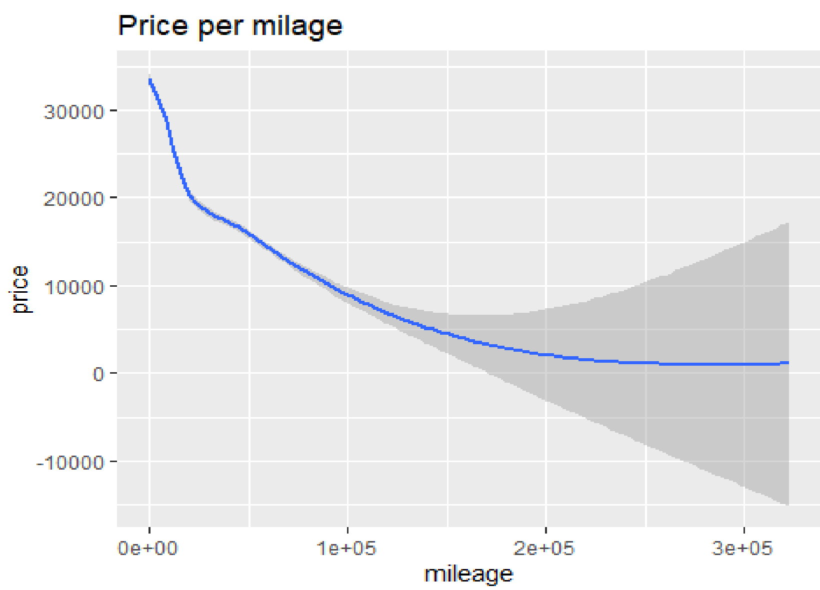
```
audi_cars %>% mutate(TR = (transmission)) %>%
 ggplot() +  geom_boxplot(aes(x=reorder(TR,price), y=((price)))) +  labs(x="transmission",
y="price")+
 ggtitle("Price per transmission")
```



We can conclude that Manual cheaper than automatic.

2.5.4 Price per milage

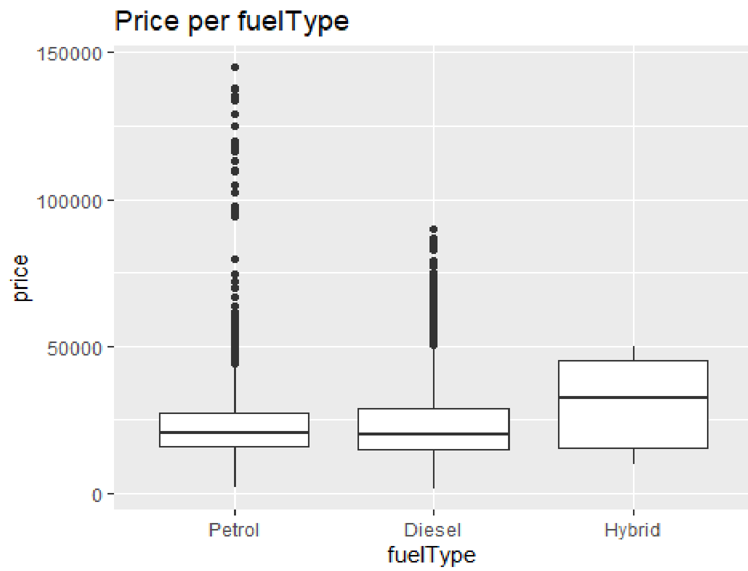
```
audi_cars %>% ggplot() +   geom_smooth(aes(x=mileage, y=price)) +
  ggtitle("Price per milage")

## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



We can conclude the more milage the little price.

2.5.5 Price per fuel type

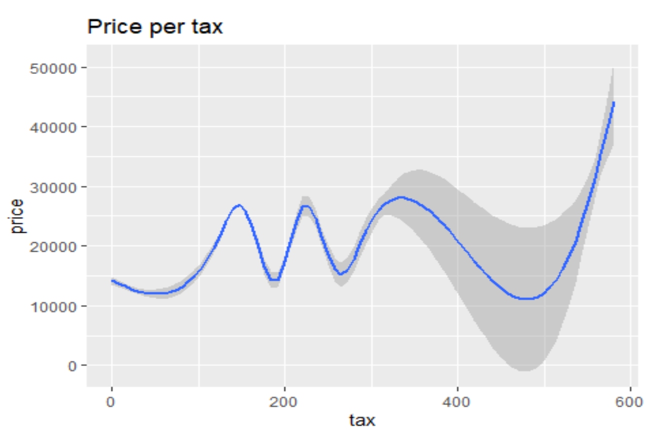
```
audi_cars %>% mutate(fuel = (fuelType)) %>%
 ggplot() +   geom_boxplot(aes(x=reorder(fuel,price), y=((price)))) +   labs(x="fuelType", y=
"price")+
 ggtitle("Price per fuelType")
```



We can conclude that Hybrid is more expensive.

2.5.6 Price per tax

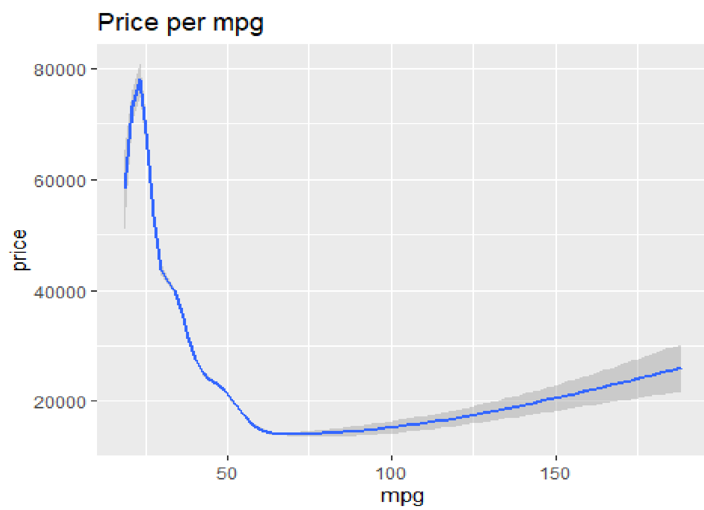
```
audi_cars %>% ggplot() +   geom_smooth(aes(x=tax, y=price)) +
 ggtitle("Price per tax")

## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



We can conclude no obvious relation between price and tax.

2.5.7 Price per mpg

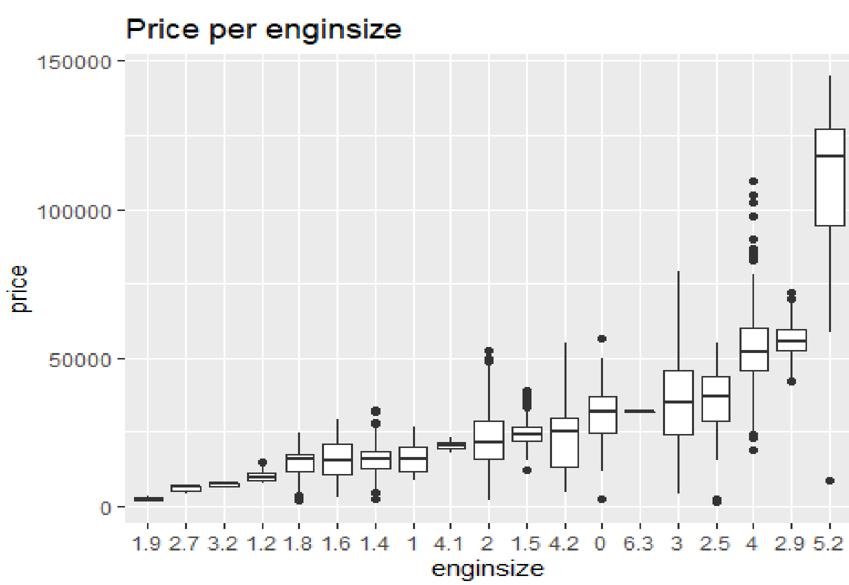
```
audi_cars %>% ggplot() +   geom_smooth(aes(x=mpg, y=price)) +
 ggtitle("Price per mpg")

## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



We can conclude inverse relation between price and mpg

2.5.8 Price per enginSize

```
audi_cars %>% mutate(enginsize = (engineSize)) %>%
 ggplot() +   geom_boxplot(aes(x=reorder(enginsize,price), y=((price)))) +
 labs(x="enginsize", y="price")+
 ggtitle("Price per enginsize")
```



We can conclude that high engine size costs a lot.

2.5.9 Numeric parameters correlation

```
audi_cars %>% select(where(is.numeric)) %>% cor()

##                  year      price     mileage         tax        mpg
## year       1.00000000  0.5927823 -0.78962998  0.09411187 -0.3515896
## price      0.59278233  1.0000000 -0.53553421  0.35631111 -0.6002946
## mileage   -0.78962998 -0.5355342  1.00000000 -0.16767597  0.3954003
## tax        0.09411187  0.3563111 -0.16767597  1.00000000 -0.6361960
## mpg       -0.35158958 -0.6002946  0.39540027 -0.63619597  1.0000000
## engineSize -0.03114220  0.5913199  0.07032809  0.39198678 -0.3653259
##            engineSize
## year      -0.03114220
## price      0.59131991
## mileage    0.07032809
## tax        0.39198678
## mpg       -0.36532588
## engineSize  1.00000000
```

The most effective numeric parameters to price are = year + mileage +mpg + engineSize

2.6 Modeling Methods

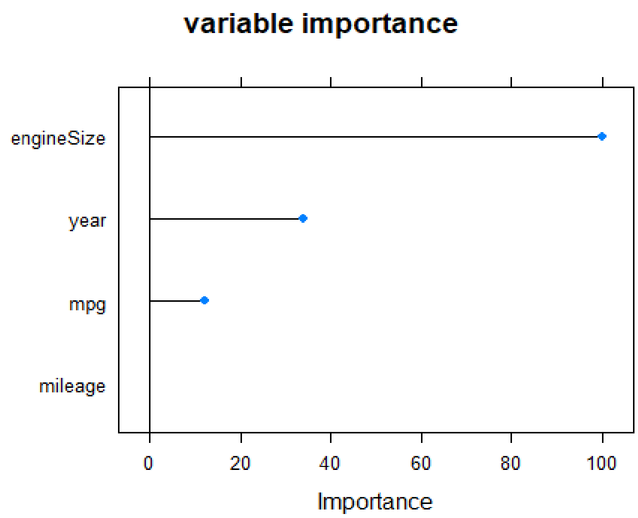
2.6.1 Model 1 - Base Line mean only

```
m1_rm= RMSE(mean(train_data$price),train_data$price)
m1_rm

## [1] 11749.86
```

2.6.2 Linear models (lm)

2.6.2.1 Model 2 (lm) -Predict price using year + mileage + mpg + engine Size

```
cv <- trainControl(  method = "repeatedcv",  number = 10,  repeats = 5)
model_lm = train(price ~ year+mileage+mpg+engineSize , data=train_data, method='lm',trC
ontrol = cv)
fit_lm = predict(model_lm,test_data)
varimp = varImp(model_lm)
plot(varimp, main="variable importance")
```




```
lm2_RM = RMSE(test_data$price, fit_lm )
comparison_lm=head(data.frame(Actual=test_data$price,Predicted=fit_lm))
comparison_lm
```

Actual £	Predicted £
12500	17025.085
16800	20512.245
17300	19276.364
13250	14712.418
10200	9140.296
16400	15884.067

2.6.2.2 Model 3 (lm) -Predict price using engineSize

```
model_lm_engin = train(price ~ engineSize , data=train_data, method='lm')
fit_lm_engin = predict(model_lm_engin,test_data)
lm3_RM = RMSE(test_data$price, fit_lm_engin )
comparison_lm_engin=head(data.frame(Actual=test_data$price,Predicted=fit_lm_engin))
comparison_lm_engin
```

Actual	Predicted
12500	16683.18
16800	23727.41
17300	11987.02
13250	23727.41
10200	16683.18
16400	16683.18

2.6.2.3 Model 4 (lm) -Predict price using year mileage * mpg* engineSize*

```
model_lm_h = train(price ~ year*mileage*mpg*engineSize , data=train_data, method='lm')
fit_lm_h = predict(model_lm_h,test_data)
lm4_RM = RMSE(test_data$price, fit_lm_h )
comparison_lm_h=head(data.frame(Actual=test_data$price,Predicted=fit_lm_h))
comparison_lm_h
```

Actual	Predicted
12500	17305.99
16800	18156.47
17300	20159.19
13250	14230.65
10200	11266.89
16400	16608.50

2.6.2.4 Model 5 (lm) -Predict price using polynomial

```
model_poly = train(price ~ (model)+poly(year,3)+poly(mileage,3)+poly(mpg,3)+poly(engin
eSize,3) , data=train_data, method='lm',trControl = cv)
fit_poly = predict(model_poly,test_data)
lm5_RM = RMSE(test_data$price, fit_poly )
comparison_poly=head(data.frame(Actual=test_data$price,Predicted=fit_poly))
comparison_poly
```

Actual	Predicted
12500	14925.590
16800	18298.613
17300	23620.111
13250	15057.494
10200	9863.857
16400	16072.147

2.6.3 Model 6 (Generalized Additive Model using Splines- gam) -Predict price using year+mileage+mpg+engineSize

```
model_gam = train(price ~ (year+mileage+mpg+engineSize) , data=train_data, method='ga
m',trControl = cv)

fit_gam = predict(model_gam,test_data)
gam6_RM = RMSE(test_data$price, fit_gam )
comparison_gam=head(data.frame(Actual=test_data$price,Predicted=fit_gam))
comparison_gam
```

Actual	Predicted
12500	17274.50
16800	18826.71
17300	21518.15
13250	13868.69
10200	10130.45
16400	16401.80

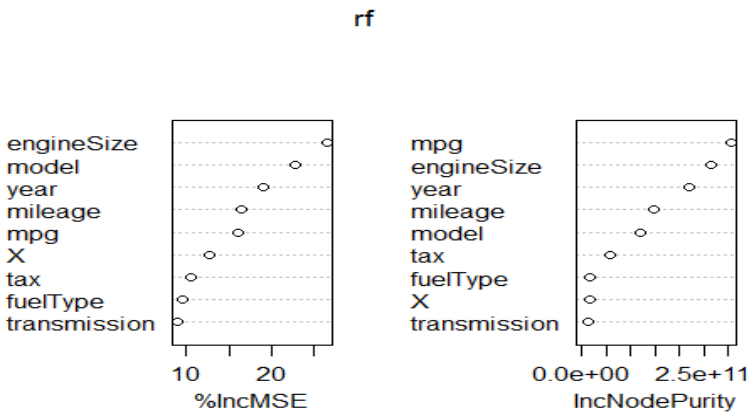
2.6.4 Model 7 (Partial Least Squares-pls) -Predict price using all

```
model_pls = train(price ~ . , data=train_data, method='pls',trControl = cv,tuneLength = 30)
fit_pls = predict(model_pls,test_data)
pls7_RM = RMSE(test_data$price, fit_pls )
comparison_pls=head(data.frame(Actual=test_data$price,Predicted=fit_pls))
comparison_pls
```

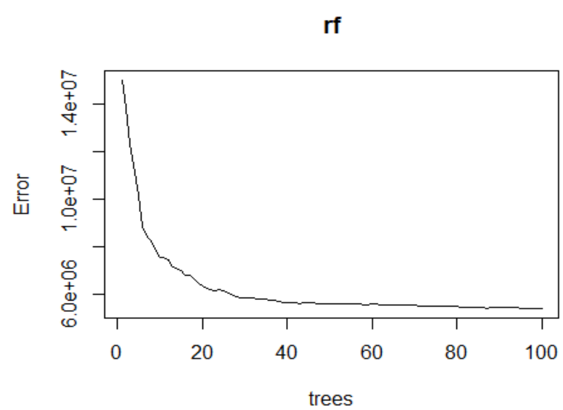
Actual	Predicted
12500	14003.97
16800	16576.42
17300	20043.56
13250	17794.77
10200	11661.30
16400	17341.29

2.6.5 Model 8 (randomForest) -Predict price using all

```
rf <- randomForest(price~., data=train_data, importance=TRUE,ntree=100,trControl = cv)
fit_rf <- predict(rf, test_data)
varImpPlot(rf)
```




```
plot(rf)
```




```
rf8_RM = RMSE(test_data$price, fit_rf )
comparison_rf=head(data.frame(Actual=test_data$price,Predicted=fit_rf))
comparison_rf
```

Actual	Predicted
12500	13915.57
16800	17021.73
17300	18202.93
13250	14787.81
10200	11731.42
16400	13505.40

2.6.6 Model 9 (Decision Tree) -Predict price using all

```
TRE = train(price~., data=train_data, method="rpart",trControl = cv)
fit_TRE = predict(TRE, test_data)
TRE9_RM = RMSE(test_data$price, fit_TRE)
comparison_TRE=head(data.frame(Actual=test_data$price,Predicted=fit_TRE))
comparison_TRE
```

Actual	Predicted
12500	15971.33
16800	15971.33
17300	25525.47
13250	15971.33
10200	15971.33
16400	15971.33

2.6.7 Model 10 (k-Nearest Neighbors) -Predict price using all

```
knn = train(price~., data=train_data, method="knn",trControl = cv)
fit_knn <- predict(knn, test_data)
knn10_rm = RMSE(test_data$price, fit_knn )
comparison_knn=head(data.frame(Actual=test_data$price,Predicted=fit_knn))
comparison_knn
```

Actual	Predicted
12500	17362.56
16800	15229.44
17300	25333.33
13250	12426.33
10200	20096.44
16400	21538.11

2.6.8 Model 11 (Multivariate Adaptive Regression Spline- Earth) -Predict price using all

```
model_earth = train(price ~ ., data=train_data, method='earth',trControl = cv)
fit_earth = predict(model_earth, test_data)
earth11_rm = RMSE(test_data$price, fit_earth )
comparison_earth=head(data.frame(Actual=test_data$price,Predicted=fit_earth))
comparison_earth
```

Actual	Predicted
12500	15942.100
16800	18476.020
17300	21123.842
13250	15067.983
10200	9905.427
16400	14625.430

2.6.9 Model 12 (Generalized Linear Model- glm) -Predict price using all

```
model_glm = train(price ~ . , data=train_data, method='glm',trControl = cv)
fit_glm = predict(model_glm,test_data)
glm12_RM = RMSE(test_data$price, fit_glm )
comparison_glm=head(data.frame(Actual=test_data$price,Predicted=fit_glm))
comparison_glm
```

Actual	Predicted
12500	14013.76
16800	16560.86
17300	20037.35
13250	17793.28
10200	11656.97
16400	17334.74

2.6.10 Model 13 (The lasso) -Predict price using all

```
model_lasso = train(price ~ . , data=train_data, method = 'glmnet', tuneGrid = expand.grid(al
pha = 1, lambda = 1),trControl = cv)
fit_lasso = predict(model_lasso,test_data)
lasso13_RM = RMSE(test_data$price, fit_lasso )
comparison_lasso=head(data.frame(Actual=test_data$price,Predicted=fit_lasso))
comparison_lasso
```

Actual	Predicted
12500	14211.47
16800	16598.48
17300	19988.70
13250	17768.29
10200	11596.49
16400	17275.60

3 Results

```
Rmse_Result=data.frame(method=c("Model 1 - Base Line", "Model 2 (lm)", "Model 3 (lm)", "Model 4 (lm)", "Model 5 (lm)", "Model 6 (Generalized Additive- gam)", "Model 7 (Partial Least Squares-pls)", "Model 8 (randomForest)", "Model 9 (Decision Tree)" , "Model 10 (k-Nearest Neighbors)", "Model 11 (Multivariate Adaptive- Earth)", "Model 12 (Generalized Linear Model- glm)", "Model 13 (The lasso)"),
             RMSE=c(m1_rm, lm2_RM, lm3_RM, lm4_RM, lm5_RM, gam6_RM,
                pls7_RM, rf8_RM, TRE9_RM, knn10_rm, earth11_rm,
                glm12_RM, lasso13_RM))


arrange(Rmse_Result, (RMSE))
```

Final Results

method	RMSE £
Model 8 (randomForest)	2982.375
Model 11 (Multivariate Adaptive- Earth)	3528.726
Model 7 (Partial Least Squares-pls)	3657.707
Model 12 (Generalized Linear Model- glm)	3657.938
Model 13 (The lasso)	3659.353
Model 5 (lm)	3883.765
Model 4 (lm)	4485.540
Model 2 (lm)	5589.255
Model 6 (Generalized Additive- gam)	5950.013
Model 9 (Decision Tree)	8293.511
Model 3 (lm)	9578.438
Model 10 (k-Nearest Neighbors)	9717.522
Model 1 - Base Line	11749.863

4 Conclusion

The goal of this project was to use machine learning models to predict the price of used car particularly Audi brand and understand what features were important. In this report, I tried 13 model, The Best model (Random Forest) obtained an RMSE of 2982.375 £ applied on the test_Data The most important features according to random forest to explain price are car model and engine size.

5 References

1. https://www.kaggle.com/adityadesai13/used-car-dataset-ford-and-mercedes?select=audi.csv
2. http://www.jomude.com/index.php/jomude/article/view/91