Used Car Analysis

By Tenzing Palden

Dataset

We are analyzing the Used car data set from kaggle.

This contains features such as Name, year, selling price, miles driven, Fuel, seller type, transmission, owner, engine, max power, torque and seats.

Data types within this set contains strings, integers but no time component.

Perfect for regression analysis.

Project Description- Why used cars?

Our analysis of used car data can be important for a number of reasons. For one, it can help car buyers make informed decisions about which used car to purchase.

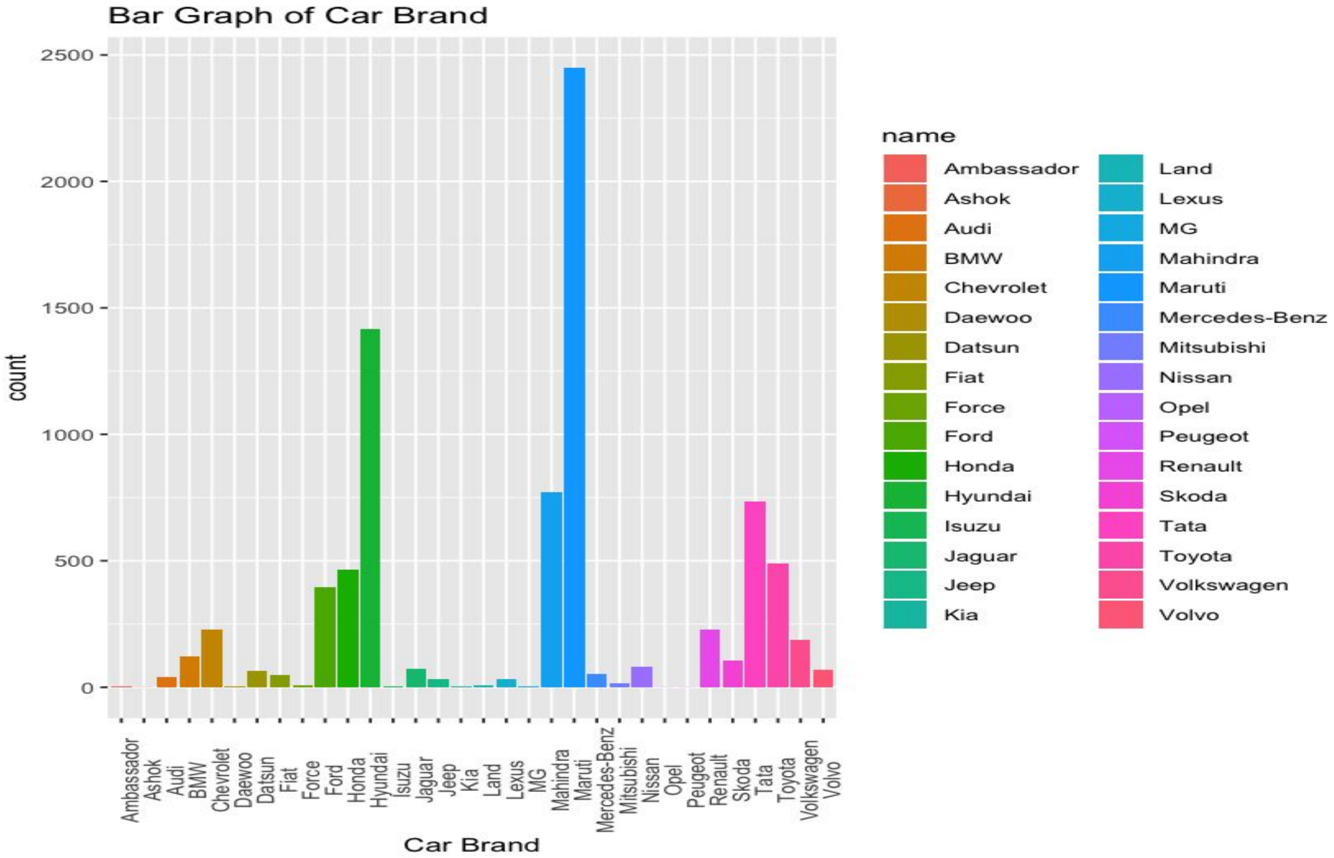
By analyzing data such as a car's make, model, age, and mileage, buyers can better understand the condition and value of a particular car.

Analyzing used car data can help car sellers determine the best price at which to sell their car, and can even help car manufacturers and dealerships understand consumer behavior and preferences, which can inform their future business decisions.

Goals

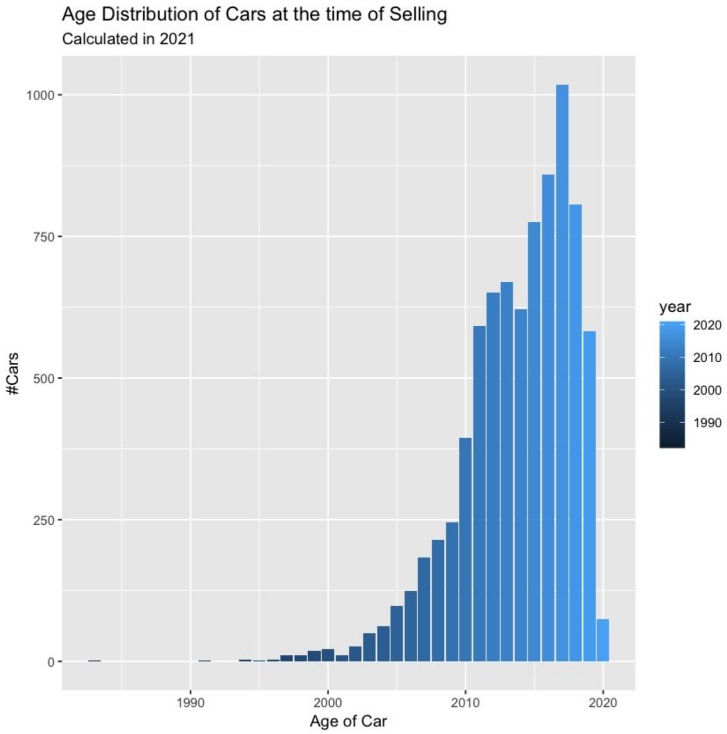
- Determine the factors that are most influential in determining a used car's price.
- Identify any potential outliers or anomalies in the data.
- Develop a regression model that can accurately predict the price of a used car based on its characteristics.
- Compare the performance of different regression algorithms on the dataset and select the most effective one.
- Evaluate the reliability and robustness of the regression model by testing it on unseen data.

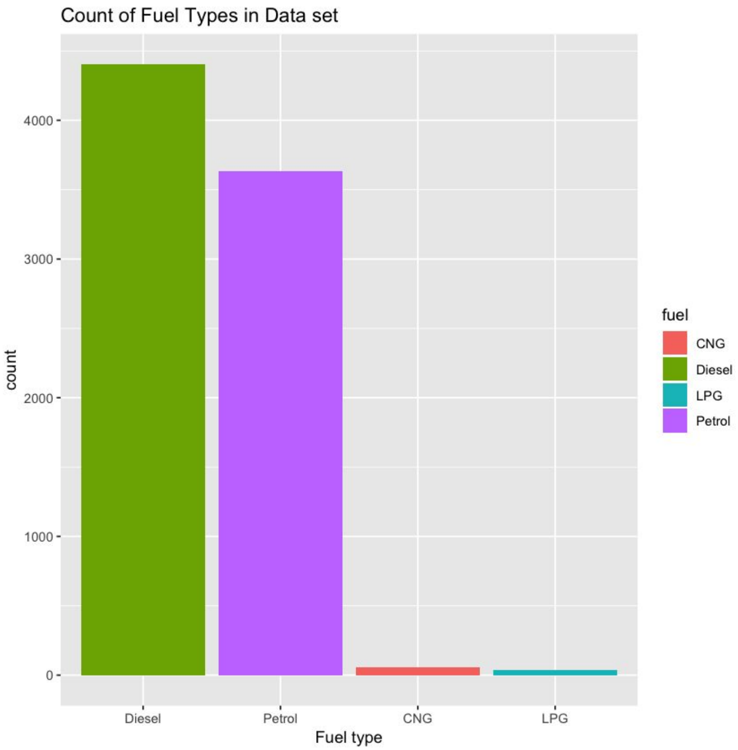
First steps... Basic Visuals



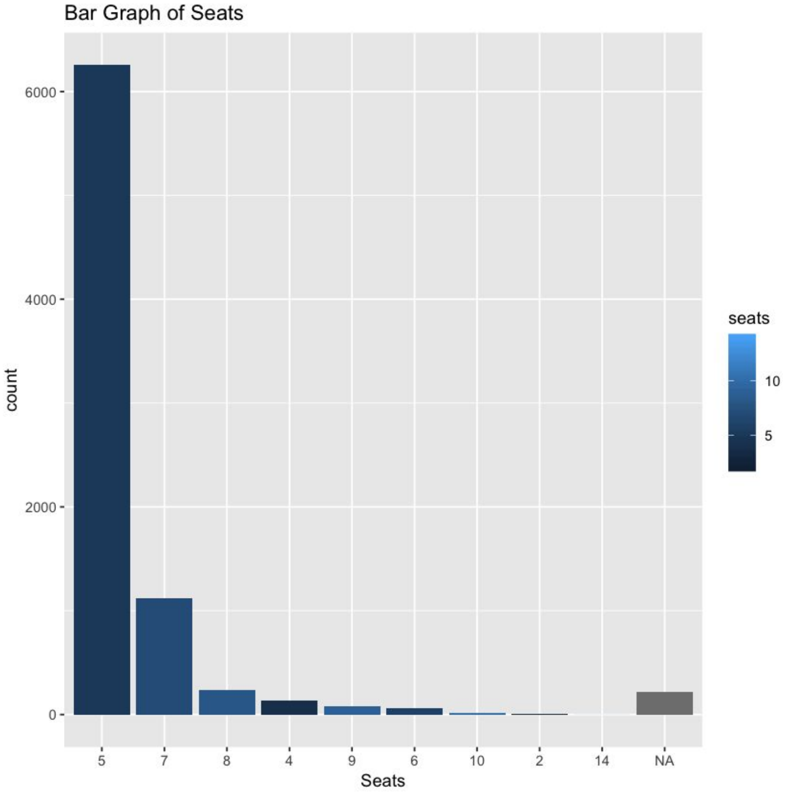
Basic Visuals cont...

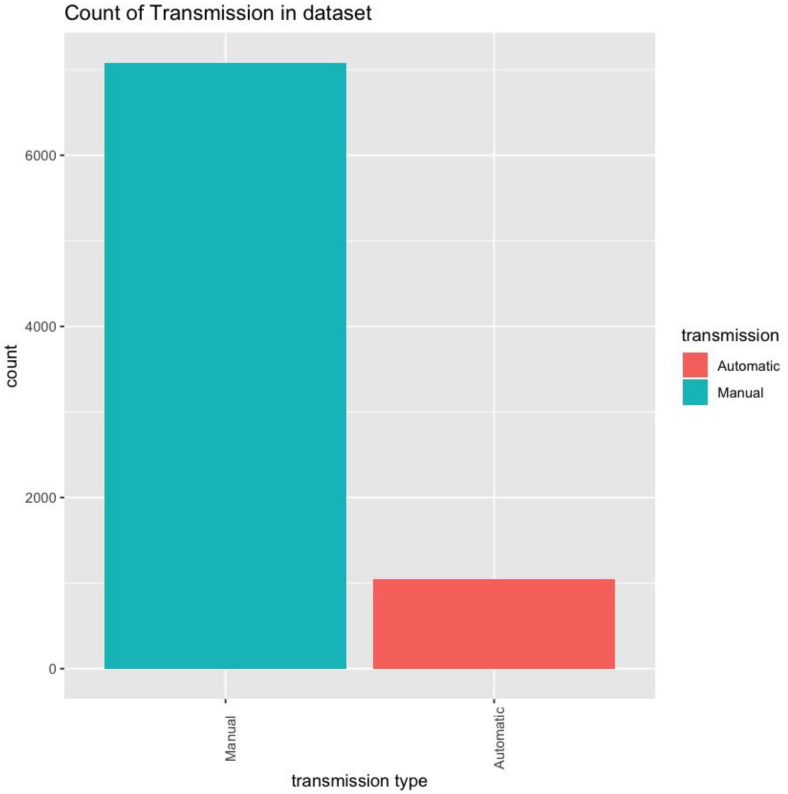
```
   0    1    2    3
4402 3631   57   38
```



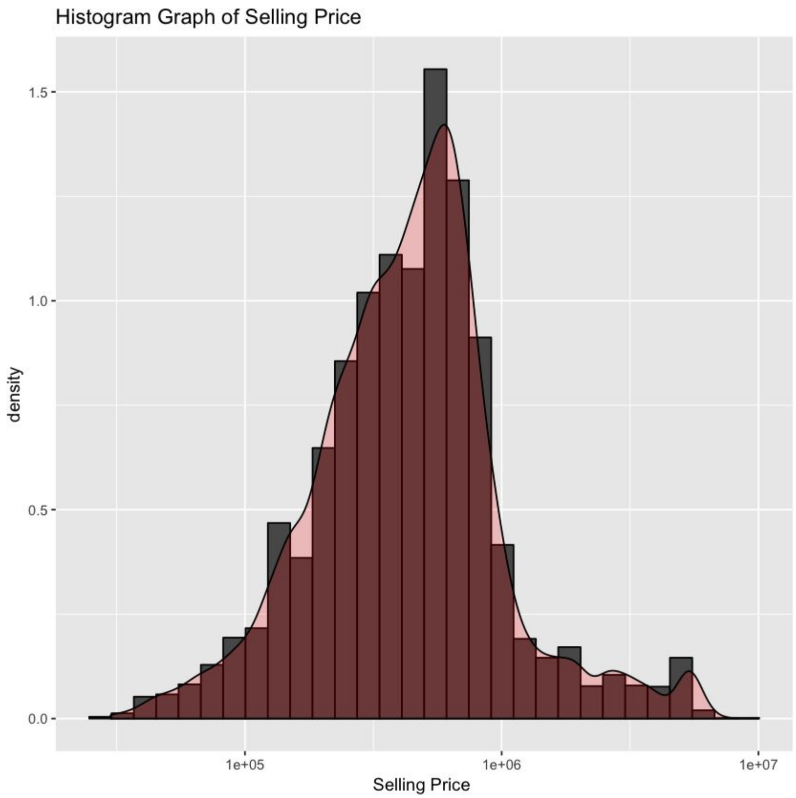


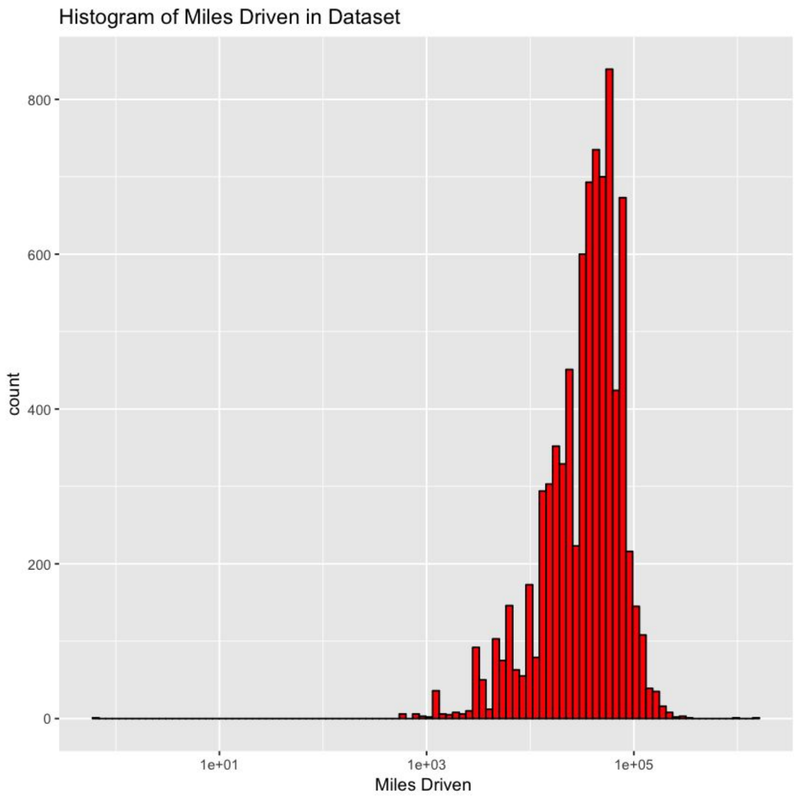
Basic Visuals cont...





Basic Visuals cont...





Data preprocessing and Missing values

```r
df_1$name <- str_replace(df_1$name, 'Maruti', '0')
df_1$name <- str_replace(df_1$name, 'Skoda', '1')
df_1$name <- str_replace(df_1$name, 'Honda', '2')
df_1$name <- str_replace(df_1$name, 'Hyundai', '3')
df_1$name <- str_replace(df_1$name, 'Toyota', '4')
df_1$name <- str_replace(df_1$name, 'Ford', '5')
df_1$name <- str_replace(df_1$name, 'Renault', '6')
df_1$name <- str_replace(df_1$name, 'Mahindra', '7')
df_1$name <- str_replace(df_1$name, 'Tata', '8')
df_1$name <- str_replace(df_1$name, 'Chevrolet', '9')
df_1$name <- str_replace(df_1$name, 'Fiat', '10')
df_1$name <- str_replace(df_1$name, 'Datsun', '11')
df_1$name <- str_replace(df_1$name, 'Jeep', '12')
df_1$name <- str_replace(df_1$name, 'Mercedes-Benz', '13')
df_1$name <- str_replace(df_1$name, 'Mitsubishi', '14')
df_1$name <- str_replace(df_1$name, 'Audi', '15')
df_1$name <- str_replace(df_1$name, 'Volkswagen', '16')
df_1$name <- str_replace(df_1$name, 'BMW', '17')
df_1$name <- str_replace(df_1$name, 'Nissan', '18')
df_1$name <- str_replace(df_1$name, 'Lexus', '19')
df_1$name <- str_replace(df_1$name, 'Jaguar', '20')
df_1$name <- str_replace(df_1$name, 'Land', '21')
df_1$name <- str_replace(df_1$name, 'MG', '22')
df_1$name <- str_replace(df_1$name, 'Volvo', '23')
df_1$name <- str_replace(df_1$name, 'Daewoo', '24')
df_1$name <- str_replace(df_1$name, 'Kia', '25')
df_1$name <- str_replace(df_1$name, 'Force', '26')
df_1$name <- str_replace(df_1$name, 'Ambassador', '27')
df_1$name <- str_replace(df_1$name, 'Ashok', '28')
df_1$name <- str_replace(df_1$name, 'Isuzu', '29')
df_1$name <- str_replace(df_1$name, 'Opel', '30')
df_1$name <- str_replace(df_1$name, 'Peugeot', '31')

#Converting car name from categorical to numerical value

df_1$name <- as.numeric(df_1$name)
table(df_1$name)
```

```r
# Checking for missing values
sapply(df_1, function(x) sum(is.na(x)))
```

```
name
        0
year
        0
selling_price
        0
km_driven
        0
fuel
        0
seller_type
        0
transmission
        0
owner
        0
mileage
        0
engine
        0
max_power
        0
seats
        0
```

We needed to change all string types to integer representation

We mapped each string in the dataset to a binary version or a numerical version.

Ex. Automatic and manual were converted to 1 and 0.

There were no missing values in the dataset.

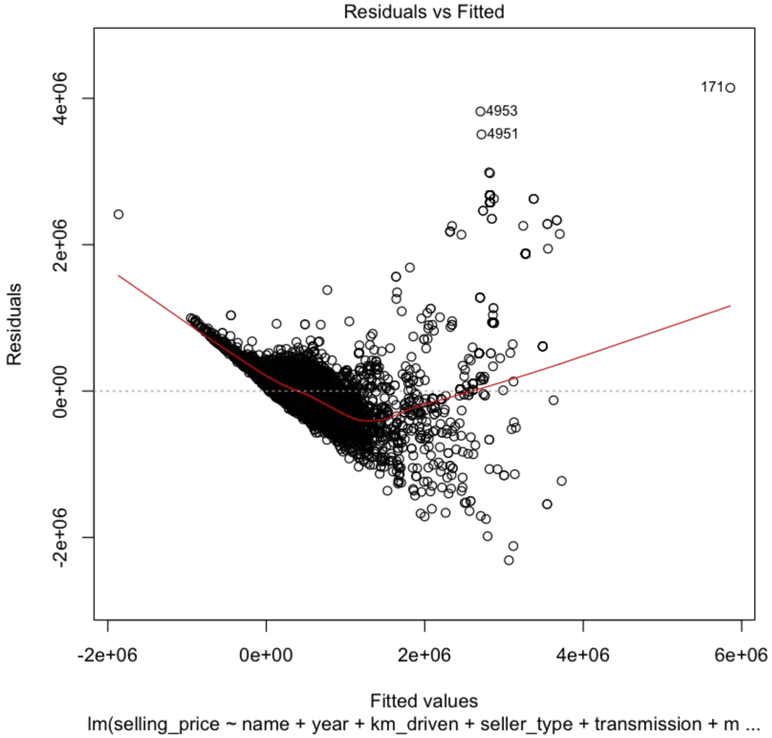
Linear Regression

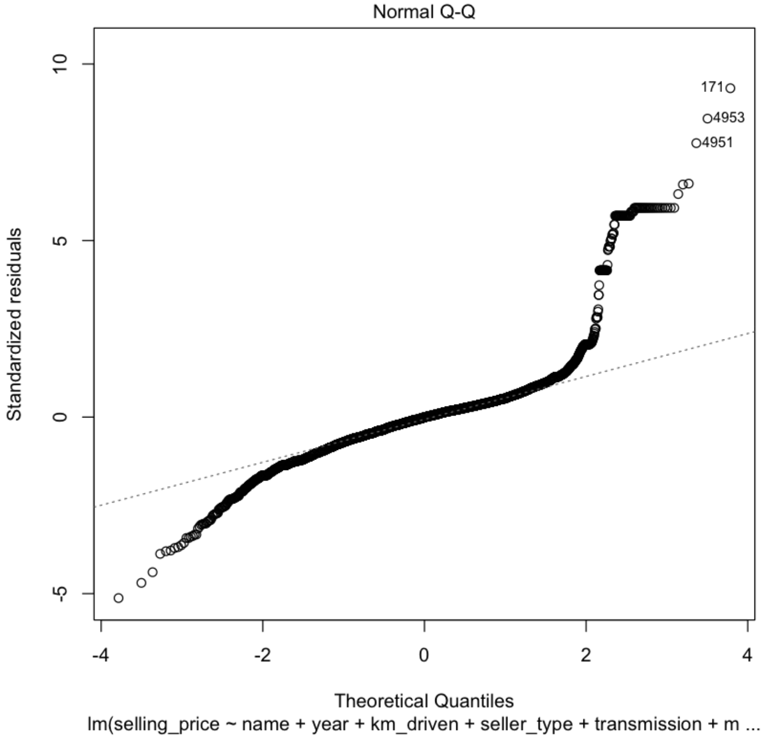
```
Call:
lm(formula = selling_price ~ ., data = trainSet)

Residuals:
     Min       1Q   Median       3Q      Max 
-2319582  -213499    -4783   157083  4187376 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -6.798e+07  3.859e+06 -17.618  < 2e-16 ***
name          2.489e+04  1.346e+03  18.486  < 2e-16 ***
year          3.340e+04  1.929e+03  17.312  < 2e-16 ***
km_driven    -8.877e-01  1.129e-01  -7.860 4.48e-15 ***
fuel         -5.309e+02  1.417e+04  -0.037    0.970    
seller_type  -1.135e+05  1.337e+04  -8.491  < 2e-16 ***
transmission  4.291e+05  2.189e+04  19.600  < 2e-16 ***
owner        -7.013e+03  9.128e+03  -0.768    0.442    
mileage       1.868e+04  2.233e+03   8.366  < 2e-16 ***
engine        3.172e+01  2.521e+01   1.258    0.208    
max_power     1.221e+04  2.836e+02  43.053  < 2e-16 ***
seats        -1.158e+04  8.797e+03  -1.316    0.188    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 452400 on 6490 degrees of freedom
Multiple R-squared:  0.6842,	Adjusted R-squared:  0.6836 
F-statistic:  1278 on 11 and 6490 DF,  p-value: < 2.2e-16
```

```
Call:
lm(formula = selling_price ~ name + year + km_driven + seller_type + 
    transmission + mileage + max_power, data = trainSet)

Residuals:
     Min       1Q   Median       3Q      Max 
-2311732  -214276    -2576   156140  4143775 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -6.824e+07  3.425e+06 -19.927  < 2e-16 ***
name          2.519e+04  1.296e+03  19.444  < 2e-16 ***
year          3.351e+04  1.707e+03  19.632  < 2e-16 ***
km_driven    -8.818e-01  1.074e-01  -8.209 2.66e-16 ***
seller_type  -1.164e+05  1.322e+04  -8.805  < 2e-16 ***
transmission  4.307e+05  2.132e+04  20.200  < 2e-16 ***
mileage       1.874e+04  1.698e+03  11.034  < 2e-16 ***
max_power     1.244e+04  2.217e+02  56.107  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 452300 on 6494 degrees of freedom
Multiple R-squared:  0.684,	Adjusted R-squared:  0.6837 
F-statistic:  2008 on 7 and 6494 DF,  p-value: < 2.2e-16
```

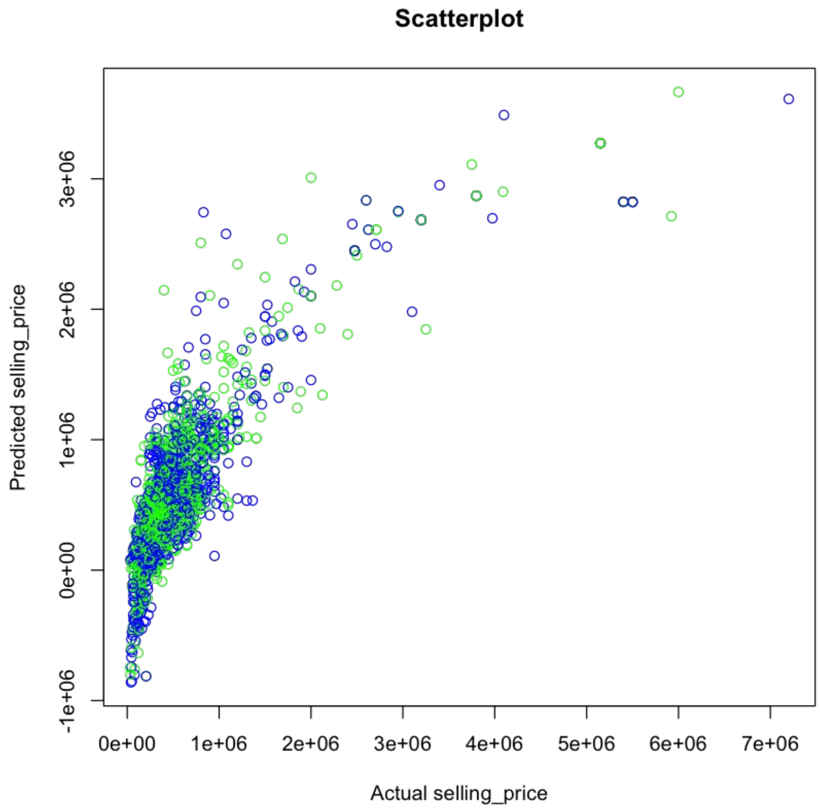
Linear Regression cont...

y=-68240000 + 25190x + 33510x+ -0.8818x + -116400x + 430700x+ 18740x+ 12440x





Linear Regression cont...



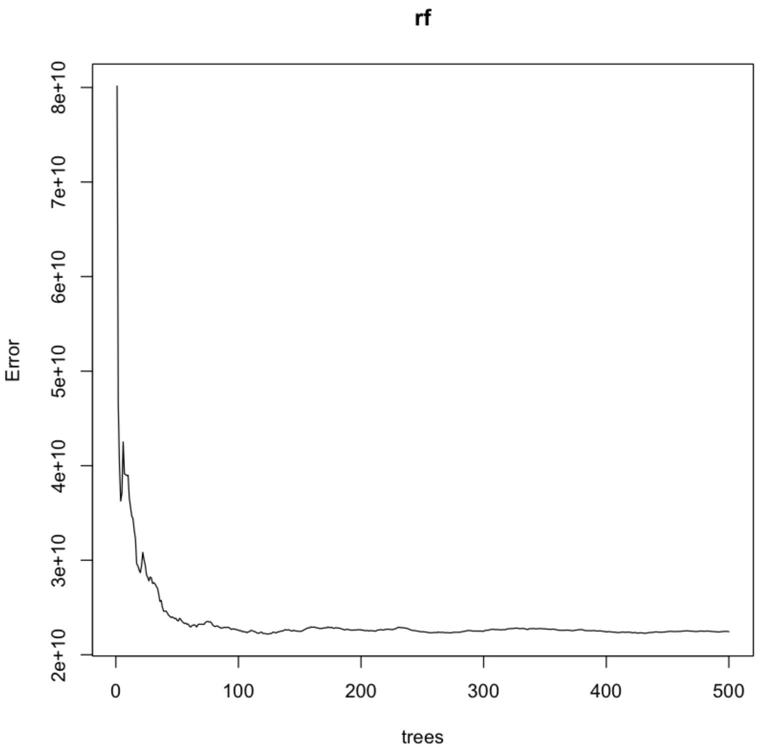
Random Forest

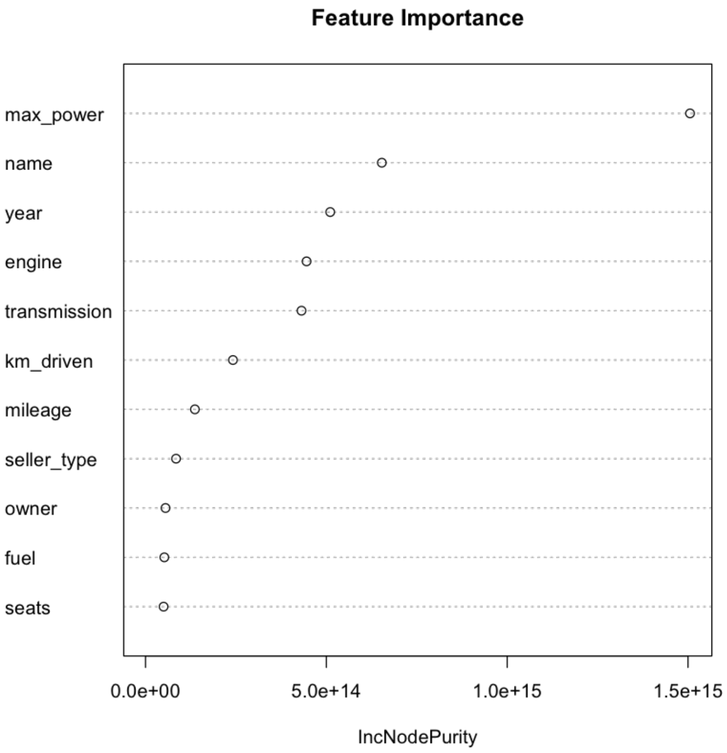
```
rf <- randomForest(selling_price~.,data = trainSet)

rf
```

```
Call:
 randomForest(formula = selling_price ~ ., data = trainSet)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 3

          Mean of squared residuals: 22452106273
                    % Var explained: 96.53
```

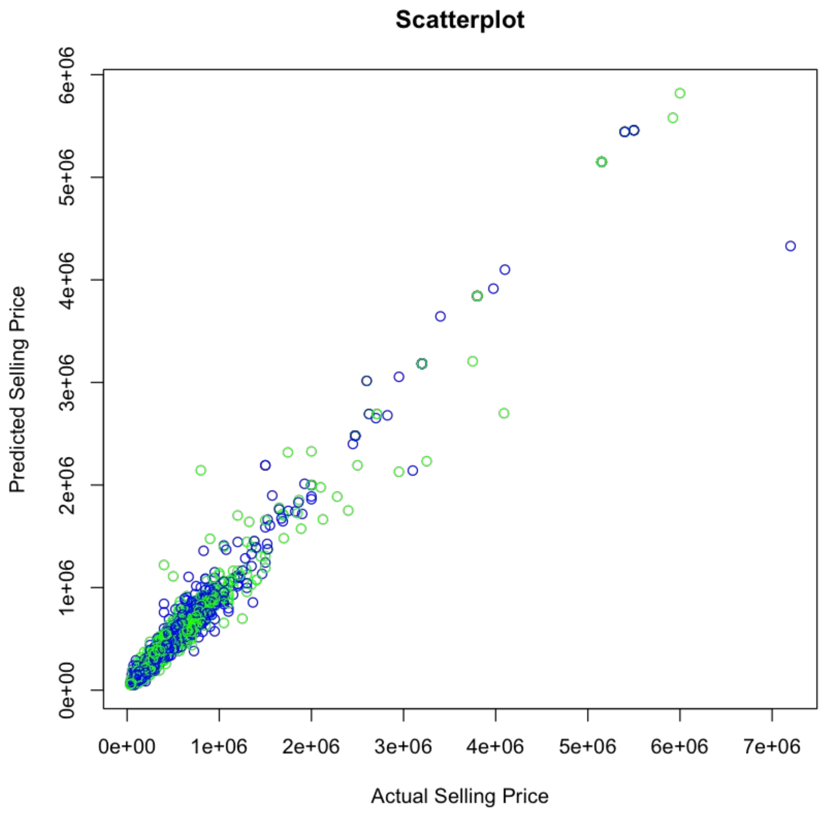
Random Forest cont...

Dotchart of variable importance as measured by a Random Forest



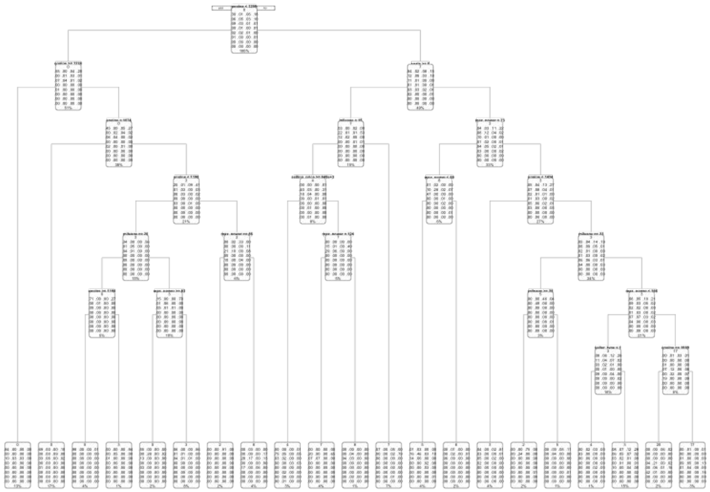


Random Forest cont...

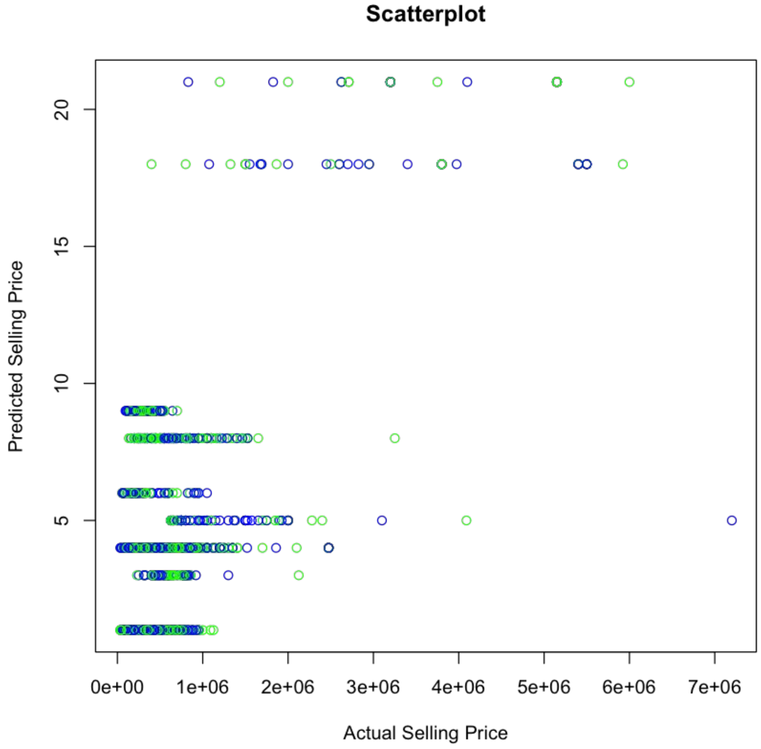


Decision Tree

Decision tree performed on Brand names



66% accuracy but this can be explained by many converging models with the free market.



Conclusion

To determine the best indicators of selling price, one's used car must have a strong max power, and second to that comes name, year, engine and transmission. This all goes along with the leading theory that the most expensive cars are the ones with high powered engines and recognizable brand names.

The Linear model performed the most accurate results, but overall we have seen that torque, fuel, engine and seller type does not matter

So when looking for a car, remember that the quality of the car is not why it is so expensive, it usually is brand and speed. If you just need to get around, shop responsibly!

Struggles and road blocks

- Struggled with working under time crunch, missing group members.

- Struggled with mapping strings with their respective numerical values

- Struggled with changing datasets. Our first idea was very hard to do any form of regression on as it did not contain a predictable goal.

- Struggled with problems with R on the computer.

Future ideas and plans.

Possible to add more features to this dataset.

(ex. Brand popularity, previous accidents, modification, tints)

Aggregate more data in the csv and compare with millions of rows instead of thousands to get a clever picture.

Scrape used car information from websites to get this data instead of having a predetermined set of data.

THANK YOU FOR YOUR TIME

By Tenzing Palden