FINAL PROJECT

PROBABILITY AND STATISTICS ASSIGNMENT

By:

AKASH RAJ,

HARSHIT GAUR,

VARUN SINGH

MASTER OF PROFESSIONAL STUDIES IN ANALYTICS

ALY 6010: PROBABILITY THEORY AND INTRODUCTORY STATISTICS

DECEMBER 12, 2021

To: **PROF. AMIN KARIMPOUR**

Table of Contents

Introduction

Data analytics is the process of analyzing raw data and generating actionable insights. It comprises of the processes, tools and techniques of data analysis and management, including the collection, organization, and storage of data. Organizations use data analytics to gain competitive advantage by improving their performance and operational efficiency. Data analytics is performed on a variety of big data sets, like transactions, click streams, server logs, electronic health records, insurance claims, etc. Different analytical techniques and algorithms can be applied on these data sets to accomplish different objectives. These different types of analytical techniques are colloquially called:-

1. **Descriptive analytics** - Summarizing the data to understand past events and performance
2. **Diagnostic analytics** - Investigating the root cause of certain events
3. **Predictive analytics** - Predicting the future for planning
4. **Prescriptive analytics** - Recommending the optimal outcomes

Irrespective of the type of analytics being performed, the basis of every method or algorithm in data analytics is descriptive/inferential statistics and machine learning. In this analysis report, we will leverage descriptive statistics to generate insights from the data.

Problems Statement :

CarDekho.com is India's leading car search venture that helps users buy cars that are right for them. It's website and app carry rich automotive content such as expert reviews, detailed specs and prices, comparisons as well as videos and pictures of all car brands and models available in India.

We have the sales data of all the cars sold during the time frame of 1983 to 2020. We are going to analyse this data set in order to help them expand their business, gain and retain customers, and stand out the competitions they face.

The data set has 8128 data points with 13 features in it related to :

Car Details - *Car name, transmission, fuel type, number of seats, year of manufacturing.*

Engine Details - *Mileage, Engine type, Torque, Maximum power in BHP.*

Sale Details - *Selling price, kilometers driven by the car.*

Later, we will implement learning algorithms and modelling techniques to understand the patterns and achieve high quality, consistent results targeting the following points :

- Identify the right prospects at the right time
- Build customer loyalty.
- Promote efficiency across the departments.
- Marketing expenditures and supply chain management.

Data Analysis

Importing the libraries required for the analysis

```
library(tidyverse)
library(grid)
library(gridExtra)
library(dplyr)
library(ggplot2)
library(reshape2)
library(DT)
library(RColorBrewer)
library(data.table)
library(knitr)
library(caret)
library(stringr)
library(RANN)
library(data.table)
library(vtable)
library(scales)
library(gplots)
library(MASS)
library(ggpubr)
```

Importing the data set for analysis

```
car_sales <- read.csv("CarDekho Sales.csv")
```

Head of data

```
# View the head and summary of the data
DT::datatable(head(car_sales, 5), rownames = FALSE)
```

Show 10 entries Search:

name	year	selling_price	km_driven	fuel	seller_type	transmission	owner	mileage	engine	max_power	torque	seats
Maruti Swift Dzire VDI	2014	450000	145500	Diesel	Individual	Manual	First Owner	23.4 kmpl	1248 CC	74 bhp	190Nm@ 2000rpm	5
Skoda Rapid 1.5 TDI Ambition	2014	370000	120000	Diesel	Individual	Manual	Second Owner	21.14 kmpl	1498 CC	103.52 bhp	250Nm@ 1500-2500rpm	5
Honda City 2017-2020 EXi	2006	158000	140000	Petrol	Individual	Manual	Third Owner	17.7 kmpl	1497 CC	78 bhp	12.7@ 2,700(kgm@ rpm)	5
Hyundai i20 Sportz Diesel	2010	225000	127000	Diesel	Individual	Manual	First Owner	23.0 kmpl	1396 CC	90 bhp	22.4 kgm at 1750-2750rpm	5
Maruti Swift VXI BSIII	2007	130000	120000	Petrol	Individual	Manual	First Owner	16.1 kmpl	1298 CC	88.2 bhp	11.5@ 4,500(kgm@ rpm)	5

Showing 1 to 5 of 5 entries Previous 1 Next

Dimension and summary of data

```
# View the dimension and summary of the data
dim(car_sales)

## [1] 8128   13

summary(car_sales)

##      name                year      selling_price        km_driven
##  Length:8128        Min.   :1983   Min.   :   29999   Min.   :      1
##  Class :character   1st Qu.:2011   1st Qu.:  254999   1st Qu.:  35000
##  Mode  :character   Median :2015   Median :  450000   Median :  60000
##                     Mean   :2014   Mean   :  638272   Mean   :  69820
##                     3rd Qu.:2017   3rd Qu.:  675000   3rd Qu.:  98000
##                     Max.   :2020   Max.   :10000000   Max.   :2360457
##
##      fuel           seller_type        transmission         owner
##  Length:8128        Length:8128        Length:8128        Length:8128
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##    mileage            engine           max_power            torque
##  Length:8128        Length:8128        Length:8128        Length:8128
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      seats
##  Min.   : 2.000
##  1st Qu.: 5.000
##  Median : 5.000
##  Mean   : 5.417
##  3rd Qu.: 5.000
##  Max.   :14.000
##  NA's   :221
```

Observation - There are 8128 observations with 13 features in the data set.

Data Cleaning

Extract value from the 'Mileage' column

The feature 'mileage' contains nominal categorical values suffixed with unit measurements of fuel efficiency. We need to clean this feature by removing the measurements (like kmpl, km/kg, etc.) and convert the data type of this feature from categorical to numeric for further utilization in analysis. We are using 'gsub' function of the 'base' package to perform this operation.

```
# Replace the 'kmpl, km/kg' string from data points in the 'mileage' feature. Convert to numeric data type.
car_sales$mileage <- as.numeric(gsub('[a-zA-Z/ ]', '', car_sales$mileage))
```

Extract value from the 'Engine' column

The feature 'engine' contains nominal categorical values suffixed with unit measurements of capacity. We need to clean this feature by removing the measurements (like CC) and convert the data type of this feature from categorical to numeric for further utilization in analysis. We are using 'gsub' function of the 'base' package to perform this operation.

```
# Replace the 'CC' string from data points in the 'engine' feature. Convert to numeric data type.
car_sales$engine <- as.numeric(gsub('[a-zA-Z/ ]', '', car_sales$engine))
```

Extract value from the 'Max Power' column

The feature 'max_power' contains nominal categorical values suffixed with unit measurements of Brake Horse Power (BHP). We need to clean this feature by removing these measurements (like bhp) and convert the data type of this feature from categorical to numeric for further utilization in analysis. We used the 'str_remove' function from the 'stringr' package to perform this operation.

```
car_sales$max_power <- as.numeric(str_remove(car_sales$max_power, " [a-z]+"))
```

Extract torque value from the 'Torque' column

The feature 'torque' contains nominal categorical values suffixed with imperial measurements of Newton meters along with 'Revolutions per minute'. We need to extract the torque values from this feature into another feature which will be further utilized in analysis and have the data type of this new feature as numeric. There were 2 values in the 'torque' feature from which we needed to extract the 1st part of the numeric value as 'Torque' and the latter part as 'RPM'. The functions 'str_sub' and 'str_locate' form the package 'stringr' were used to perform this operation.

```
car_sales$torque_val <- as.numeric(str_sub(car_sales$torque, rep(1, nrow(car_sales)),
                                           str_locate(car_sales$torque, "\\D+")[,1]-1))
```

Extract RPM value from the torque column

The feature 'torque' contains nominal categorical values suffixed with imperial measurements of Newton meters along with 'Revolutions per minute'. We need to extract the RPM values from this 'torque' feature into another feature which will be further utilized in analysis and have the data type of this new feature as numeric. We extracted the first part of torque as 'Torque Value' and needed to extract the second part as 'RPM' value. The functions of 'str_sub' and 'gsub' were used to perform these operations.

```
num_length <- str_length(gsub("\\D+", "", car_sales$torque))
car_sales$rpm <- as.numeric(str_sub(as.numeric(gsub("\\D+", "", car_sales$tor
que)),num_length-3, num_length))
```

Extract brand from the car name column

The feature 'name' contains the name of the car, its manufacturer, and the model. We will extract the Brand name from this feature into another feature which will be further utilized in analysis. Now, there are several data points where the brand names contain single word and multiple words. We formulated the extraction to take out the words corresponding to the brand names these cars belong to.

```
# Extract the brand name (first word) into another feature from the 'name' fe
ature.
car_sales$brand <- word(car_sales$name, start = 1, end = 1)

# Extract the brand name (first 2 words) into another feature from the 'name'
feature.
car_sales$brand[car_sales$brand == "Ashok"] <- word(car_sales$name[car_sales$
brand == "Ashok"], start = 1, end = 2)
car_sales$brand[car_sales$brand == "Land"] <- word(car_sales$name[car_sales$b
rand == "Land"], start = 1, end = 2)
car_sales$brand[car_sales$brand == "Range"] <- word(car_sales$name[car_sales$
brand == "Range"], start = 1, end = 2)
```

Filter out the values for CNG and LPG

The feature 'fuel' contains 4 different types namely Diesel, Petrol, CNG, LPG. We will eliminate those records which belong to CNG and LPG as their cumulative data points account for only 1.5% of the total records in the data set.

```
car_sales <- filter(car_sales, !fuel %in% c("LPG", "CNG"))
```

Identify blank and NA values in the data

We further checked the data set for NA and blank values in all of its features.

```
colSums(is.na(car_sales))

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

```
##             0             0             0           214           214
##     max_power        torque         seats    torque_val           rpm
##           208             0           214           214           214
##         brand
##             0
```

Remove unnecessary columns in the dataframe

We removed some of the unnecessary columns from the data set as either informative values were extracted from them or were of no use.

```
car_sales$name <- NULL
car_sales$torque <- NULL
names(car_sales)[12] <- "torque"
```

Observation - We found 214 records with NA values in most of the numeric features. These features need to be either removed or imputed by mean, median, etc.

Impute missing and NA values in the data using kNN

The data set contained few data points which contained missing or NA values in some of their features. We implemented the kNN algorithm to impute these missing and NA values with means of the features. This way we wouldn't need to eliminate these records and can utilize them as well for our analysis. The functions 'preProcess' and 'predict' from the 'caret' package were used to perform these operations.

```
preProcValues <- preProcess(car_sales %>%
                              dplyr::select(year, selling_price, km_driven, f
uel, seller_type,
                                            transmission, owner, mileage, eng
ine, max_power, seats, torque, rpm),
                            method = c("knnImpute"),
                            k = 20,
                            knnSummary = mean)

impute_cars_info <- predict(preProcValues, car_sales, na.action = na.pass)

procNames <- data.frame(col = names(preProcValues$mean), mean = preProcValues
$mean, sd = preProcValues$std)

for(i in procNames$col){
  car_sales[i] <- impute_cars_info[i]*preProcValues$std[i]+preProcValues$mean
[i]
}

colSums(is.na(car_sales))
```

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

```
##         seats        torque           rpm         brand
##             0             0             0             0
```

Observation - No NA or missing values were found after the kNN imputation.

Feature Engineering

Create categorical column to indicate the size of the car

A new feature named 'Car Type' is created to indicate the size of the car. We formulated the feature 'size' to calculate the size of car based on the following rule set :

If number of seats are less than or equal to 5, the type of car is "Small".

If number of seats are greater than 5 and less than equal to 8, the type of car is "Medium".

If number of seats are greater than 8 and less than equal to 14, the type of car is "Large".

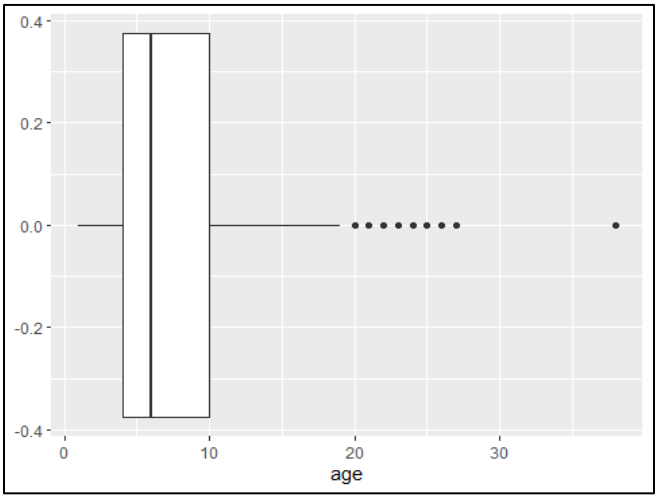
If number of seats don't lie in the above range, we give the type as "Others" to the cars.

```
car_sales$car_type <- ifelse(car_sales$seats <= 5, "Small", ifelse(car_sales$
seats <= 8, "Medium", ifelse(car_sales$seats <= 14, "Large", "Others")))
unique(car_sales$car_type)

## [1] "Small"  "Medium" "Large"
```

Create a numeric column to obtain the age of the car

We created another feature named 'age' which signifies the age of the car. The feature 'year' was utilised to calculate the age of all the cars.

```
car_sales$age <- as.numeric(format(Sys.Date(), "%Y")) - car_sales$year
ggplot(car_sales) +
  geom_boxplot(mapping = aes(age))
```



Unit Modification (Changing 1000s to 1s) of two features

The values in the features 'Selling Price' and 'KMs driven' are in hundreds of thousands. We reduced these values from 1000s to Ones in order for proper analysis and better scaling of visualisations.

```
car_sales$selling_price <- round( (car_sales$selling_price / 1000), 2)
car_sales$km_driven <- round( (car_sales$km_driven / 1000), 2)

#car_sales[sapply(car_sales, as.character)] <- lapply(car_sales[sapply(car_sa
les, as.character)], as.factor)
```

Changing Character data type to Factor data type

The features belonging to character data types need to be changed into factor data type for efficient text manipulations.

```
car_sales$fuel <- as.factor(car_sales$fuel)
car_sales$seller_type <- as.factor(car_sales$seller_type)
car_sales$transmission <- as.factor(car_sales$transmission)
car_sales$owner <- as.factor(car_sales$owner)
car_sales$brand <- as.factor(car_sales$brand)
car_sales$car_type <- as.factor(car_sales$car_type)

#car_sales[sapply(car_sales, as.character)] <- lapply(car_sales[sapply(car_sa
les, as.character)], as.factor)
```

Exploratory Data Analysis

Descriptive Statistics of the data set

```
# Descriptive Statistics of the data set.
mul_fun <- function(x) {
  c(mean(x), sd(x), median(x), min(x), max(x), max(x)-min(x), quantile(x, 0.2
5),
    quantile(x, 0.5), quantile(x, 0.75))
}

var_names <- c("Mean", "Std Dev", "Median", "Min", "Max", "Range", "Percentil
e 25", "Percentile 50",
               "Percentile 75")
num_cols <- c("selling_price", "km_driven", "mileage", "engine", "max_power",
"torque", "rpm", "age")
desc_stats <- as.data.frame(round(sapply(car_sales[num_cols], mul_fun), 2), r
ow.names = var_names)
DT::datatable(desc_stats)
```

Show 10 entries Search:

	selling_price	km_driven	mileage	engine	max_power	torque	rpm	age
Mean	642.74	69.74	19.34	1458.82	91.44	167.72	3055.79	7.18
Std Dev	809.86	56.64	3.97	500.69	35.6	96.92	917.43	4.03
Median	450	60	19.16	1248	82	154	3000	6
Min	30	1	0	624	0	4	400	1
Max	10000	2360.46	42	3604	400	789	5300	38
Range	9970	2359.46	42	2980	400	785	4900	37
Percentile 25	260	35	16.78	1197	68.07	101	2400	4
Percentile 50	450	60	19.16	1248	82	154	3000	6
Percentile 75	680	98	22.15	1582	102	202	4000	10

Showing 1 to 9 of 9 entries Previous 1 Next

Observation - From the statistics table, we found the below points :

The minimum value of mileage is **0**. On further analysis, we found that some of the cars contain *0.0 Kmpl* values as their mileage. We would need to impute these values in our second phase of analysis.

The minimum value of max power also came out to be **0**. We found that some of the cars contain *0 bhp* values as their max power. On further analysis, we would need to impute these values as well.

Since, the cars are manufactured in the time frame of 1983 to 2020, the attributes (mileage, max power, engine, torque, and rpm) are very dispersed.

Mileage has the least skewed distribution as its mean and median are almost same for it.

Identify the Outliers in numeric variables

```
num_cols <- c("selling_price", "km_driven", "mileage", "engine", "max_power",
"torque", "rpm", "age")
outlier_val <- sapply(car_sales[num_cols], function(x) boxplot.stats(x)$out)
min_outlier <- data.frame('Count of Outliers' = sapply(outlier_val, length),
check.names = FALSE)
DT::datatable(min_outlier)
```

Show 10 entries Search:

	Count of Outliers
selling_price	600
km_driven	168
mileage	18
engine	1186
max_power	573
torque	405
rpm	0
age	78

Showing 1 to 8 of 8 entries Previous 1 Next

Observation - The count of outliers corresponding to their features have been tabulated in the above table. The count of outliers in the feature 'Selling Type' is 600 and 'engine' has outliers in north of 1000. We found that 'RPM' has no outliers in the data set.

These outliers will need to be dealt with before building the regression models.

Univariate Analysis

Frequency plot of the categorical variables

We have various categorical features which can provide insightful information about the data set. The features for which we have formulated the frequency tables are :

Fuel Type

Seller Type

Transmission Type

Owner Type

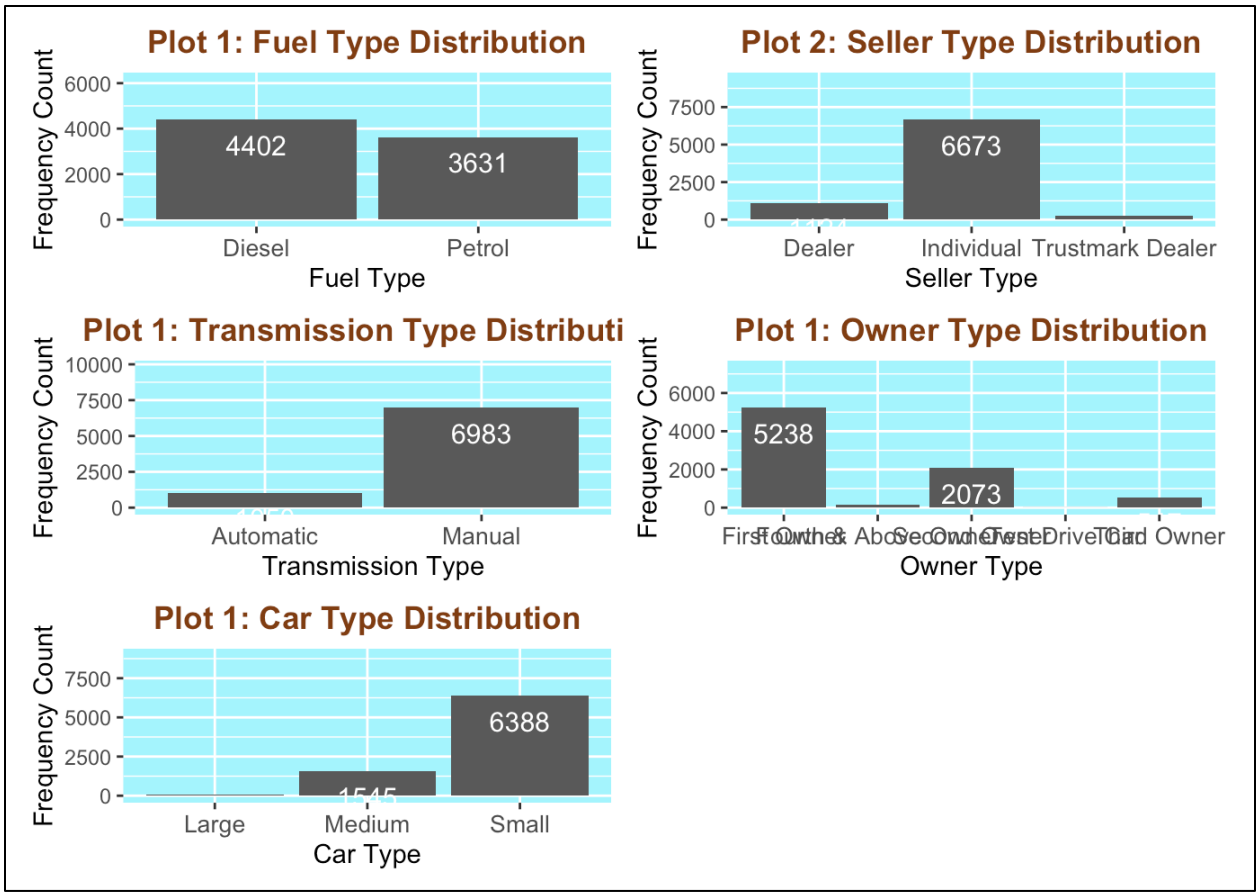
Car Type

```
# Frequency Table of Brands
freqTable_fuel <- dplyr::count(car_sales, car_sales$fuel)
colnames(freqTable_fuel) <- c("Fuel Type", "Frequency")

freqTable_sellerType <- dplyr::count(car_sales, car_sales$seller_type)
colnames(freqTable_sellerType) <- c("Seller Type", "Frequency")

freqTable_transmission <- dplyr::count(car_sales, car_sales$transmission)
colnames(freqTable_transmission) <- c("Transmission", "Frequency")

freqTable_owner <- dplyr::count(car_sales, car_sales$owner)
colnames(freqTable_owner) <- c("Owner Type", "Frequency")

freqTable_carType <- dplyr::count(car_sales, car_sales$car_type)
colnames(freqTable_carType) <- c("Car Type", "Frequency")

bar_fuel <- ggplot(freqTable_fuel)+
  geom_bar(
    mapping = aes(
      x = `Fuel Type`,
      y = Frequency
    ),
    stat="identity"
  ) +
  theme(panel.background = element_rect("#a7f4fc")) +
  labs(
```

```
    title = "Plot 1: Fuel Type Distribution",
    x = "Fuel Type",
    y = "Frequency Count"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5, colour = "#7F3D17", face = "bold")
,
    axis.text.x = element_text(vjust = 1, size = 10)
  ) +
  scale_y_continuous(labels = comma) +
  ylim(0, max(freqTable_fuel$Frequency) * 1.4) +
  geom_text(aes(x = `Fuel Type`, y = Frequency, label = Frequency), vjust = 1
.8, color = "WHITE", size = 4)

bar_sellerType <- ggplot(freqTable_sellerType)+
  geom_bar(
    mapping = aes(
      x = `Seller Type`,
      y = Frequency
    ),
    stat="identity"
  ) +
  theme(panel.background = element_rect("#a7f4fc")) +
  labs(
    title = "Plot 2: Seller Type Distribution",
    x = "Seller Type",
    y = "Frequency Count"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5, colour = "#7F3D17", face = "bold")
,
    axis.text.x = element_text(vjust = 1, size = 10)
  ) +
  scale_y_continuous(labels = comma) +
  ylim(0, max(freqTable_sellerType$Frequency) * 1.4) +
  geom_text(aes(x = `Seller Type`, y = Frequency, label = Frequency), vjust =
1.8, color = "WHITE", size = 4)

bar_transmission <- ggplot(freqTable_transmission)+
  geom_bar(
    mapping = aes(
      x = Transmission,
      y = Frequency
    ),
    stat="identity"
  ) +
  theme(panel.background = element_rect("#a7f4fc")) +
  labs(
    title = "Plot 1: Transmission Type Distribution",
    x = "Transmission Type",
```

```
    y = "Frequency Count"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5, colour = "#7F3D17", face = "bold")
,
    axis.text.x = element_text(vjust = 1, size = 10)
  ) +
  scale_y_continuous(labels = comma) +
  ylim(0, max(freqTable_transmission$Frequency) * 1.4) +
  geom_text(aes(x = Transmission, y = Frequency, label = Frequency), vjust =
1.8, color = "WHITE", size = 4)

bar_owner <- ggplot(freqTable_owner)+
  geom_bar(
    mapping = aes(
      x = `Owner Type`,
      y = Frequency
    ),
    stat="identity"
  ) +
  theme(panel.background = element_rect("#a7f4fc")) +
  labs(
    title = "Plot 1: Owner Type Distribution",
    x = "Owner Type",
    y = "Frequency Count"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5, colour = "#7F3D17", face = "bold")
,
    axis.text.x = element_text(vjust = 1, size = 10, angle = 0)
  ) +
  scale_y_continuous(labels = comma) +
  ylim(0, max(freqTable_owner$Frequency) * 1.4) +
  geom_text(aes(x = `Owner Type`, y = Frequency, label = Frequency), vjust =
1.8, color = "WHITE", size = 4)

bar_carType <- ggplot(freqTable_carType)+
  geom_bar(
    mapping = aes(
      x = `Car Type`,
      y = Frequency
    ),
    stat="identity"
  ) +
  theme(panel.background = element_rect("#a7f4fc")) +
  labs(
    title = "Plot 1: Car Type Distribution",
    x = "Car Type",
    y = "Frequency Count"
  ) +
```

```
  theme(
    plot.title = element_text(hjust = 0.5, colour = "#7F3D17", face = "bold")
,
    axis.text.x = element_text(vjust = 1, size = 10)
  ) +
  scale_y_continuous(labels = comma) +
  ylim(0, max(freqTable_carType$Frequency) * 1.4) +
  geom_text(aes(x = `Car Type`, y = Frequency, label = Frequency), vjust = 1.
8, color = "WHITE", size = 4)

grid.arrange(bar_fuel, bar_sellerType, bar_transmission, bar_owner, bar_carTy
pe)
```



Observation - From the above graphs, we can figure out the following points:

There is not much difference in between the frequency of diesel and petrol cars.

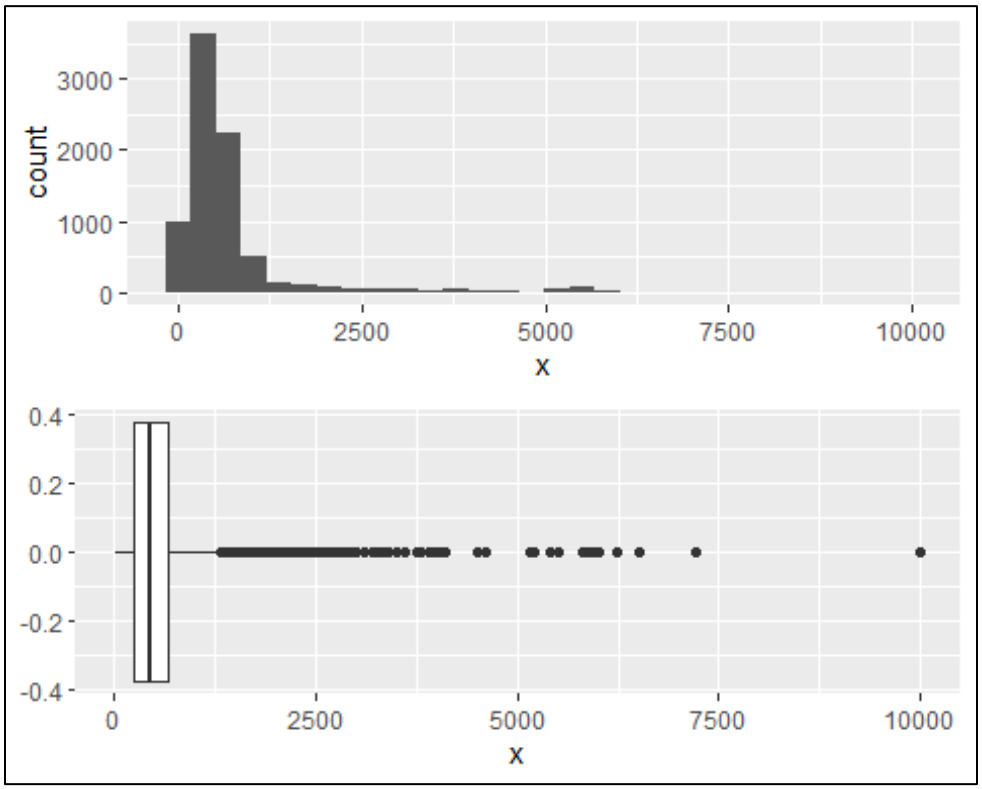
Individual seller type accounts for the most percentage of the cars in the data set. 75% of the cars on CarDekho.com belong to the 'Individual' seller type.

Majority of customers prefer cars of 'Manual' transmission type over 'Automatic' and of 'Small' size.

Histogram and Box plot of numeric variables

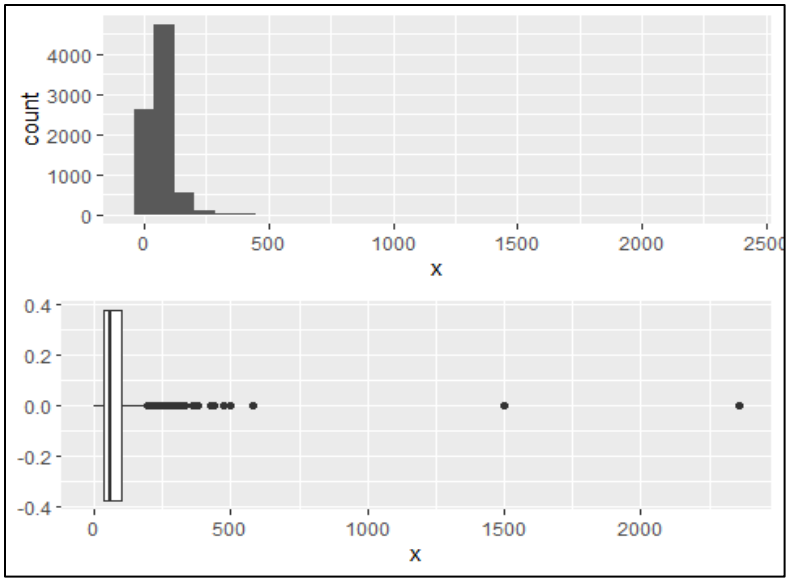
In the following section, we're plotting the histograms and boxplots to understand the univariate distribution of the numeric variables.

```
# Function to plot graph
univariate_plot <- function(x){
  grid.arrange(ggplot(data = car_sales) + geom_histogram(mapping = aes(x)),
               ggplot(data = car_sales) + geom_boxplot(mapping = aes(x)))
}

#Selling Price
univariate_plot(car_sales$selling_price)
```



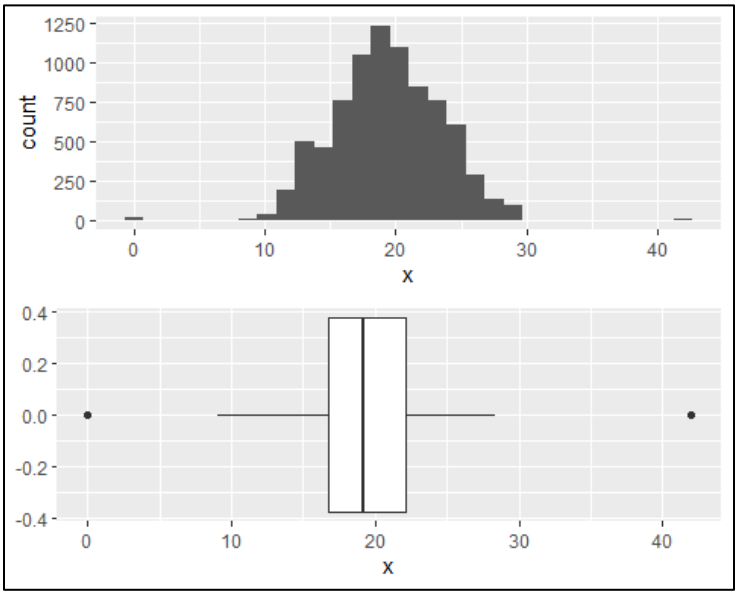
Observation - Most of the data is concentrated in the range of 0 to 1300 and any value above 1325 are outliers.

```
#Kilometer Driven
univariate_plot(car_sales$km_driven)
```



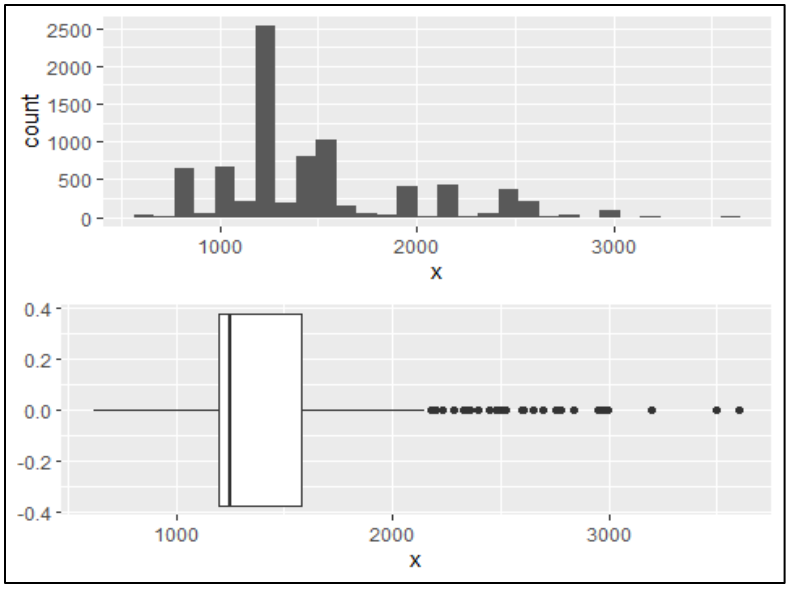
Observation - Mahindra XUV500 W6 2WD and Hyundai i20 Asta 1.2 are outliers to the data set as they have been driven over 150,000 kms.

```
#Car mileage
univariate_plot(car_sales$mileage)
```



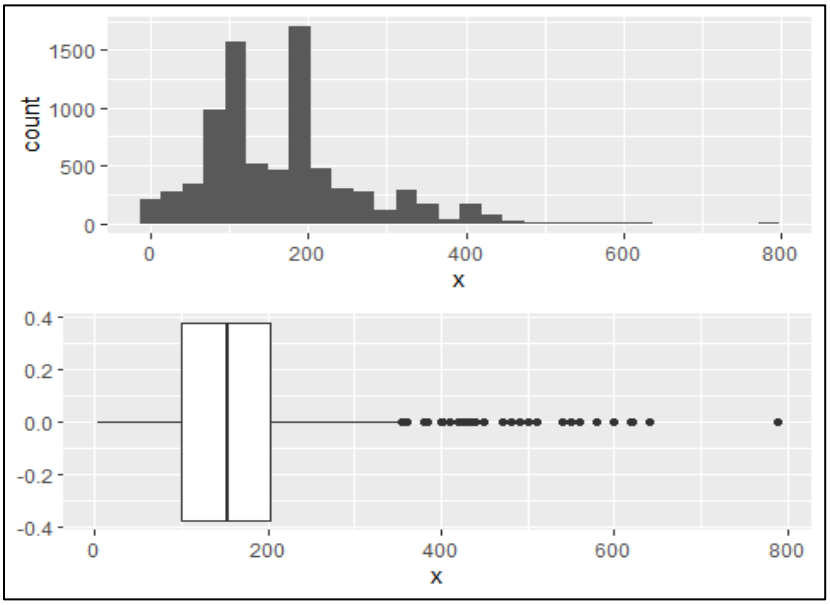
Observation - There are 2 outliers in both of the extremes. Otherwise, the data seems to be normally distributed.

```
#Engine BHP
univariate_plot(car_sales$engine)
```



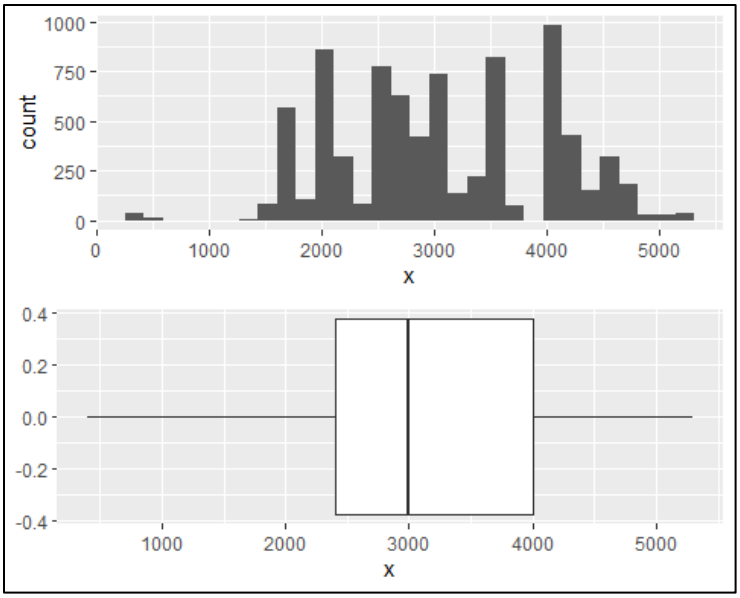
Observation - A fair share of cars belong to the engine types of 1200 CC. Records in north of 2200 CC are outliers to the data set.

```
# Torque
univariate_plot(car_sales$torque)
```



Observation - The data set seems to be normally distributed for the feature 'torque' with the presence of some outliers.

```
# RPM
univariate_plot(car_sales$rpm)
```



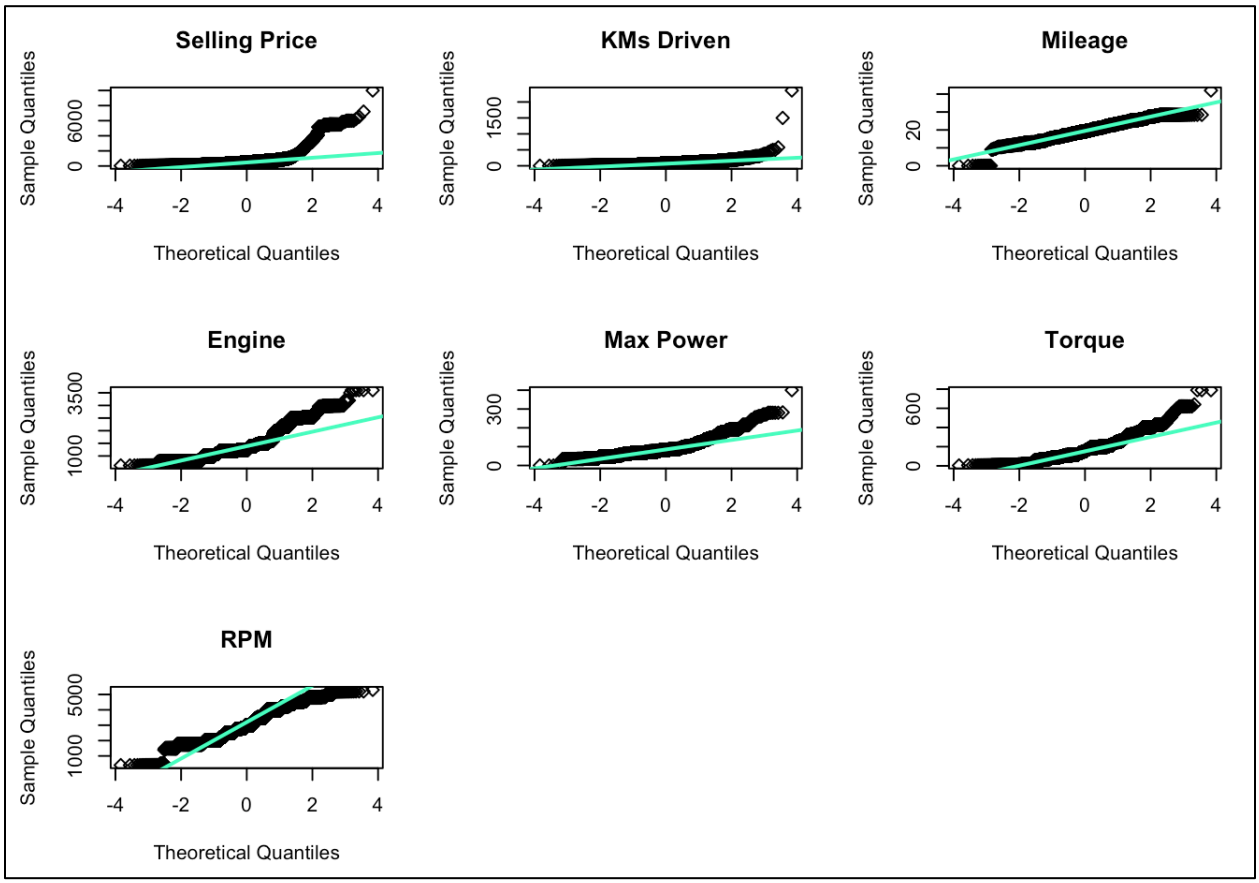
Observation - There are no outliers in the data set for the feature 'rpm' and no legible distribution can be figured out from the data.

Check for the normality of numeric variables

Q-Q Plot is used to visualise the normality of the numeric data. If the 'Sample' and 'Theoretical' quantiles fall in the same line, those features can be considered normally distributed.

```
# Check Normality using Shapiro-Wilks Test
#shapiro.test(car_sales$selling_price)

# Function to plot graph
qq_plot <- function(numeric_feature, mainTitle) {
  qqnorm(numeric_feature, pch = 5, frame = TRUE, main = mainTitle)
  qqline(numeric_feature, col = "#52ffbf", lwd = 2)
}

# Changing Plot Matrix Size to 3x2.
par(mfrow = c(3,3))

# Check Normality using Q-Q Plot of 'Selling Price' Feature.
qq_plot(car_sales$selling_price, "Selling Price")

# Check Normality using Q-Q Plot of 'KMs Driven' Feature.
qq_plot(car_sales$km_driven, "KMs Driven")

# Check Normality using Q-Q Plot of 'Mileage' Feature.
qq_plot(car_sales$mileage, "Mileage")
```

```
# Check Normality using Q-Q Plot of 'Engine' Feature.
qq_plot(car_sales$engine, "Engine")

# Check Normality using Q-Q Plot of 'Max Power' Feature.
qq_plot(car_sales$max_power, "Max Power")

# Check Normality using Q-Q Plot of 'Torque' Feature.
qq_plot(car_sales$torque, "Torque")

# Check Normality using Q-Q Plot of 'RPM' Feature.
qq_plot(car_sales$rpm, "RPM")

# Resetting Plot Matrix Size to 1x1.
par(mfrow = c(1,1))
```



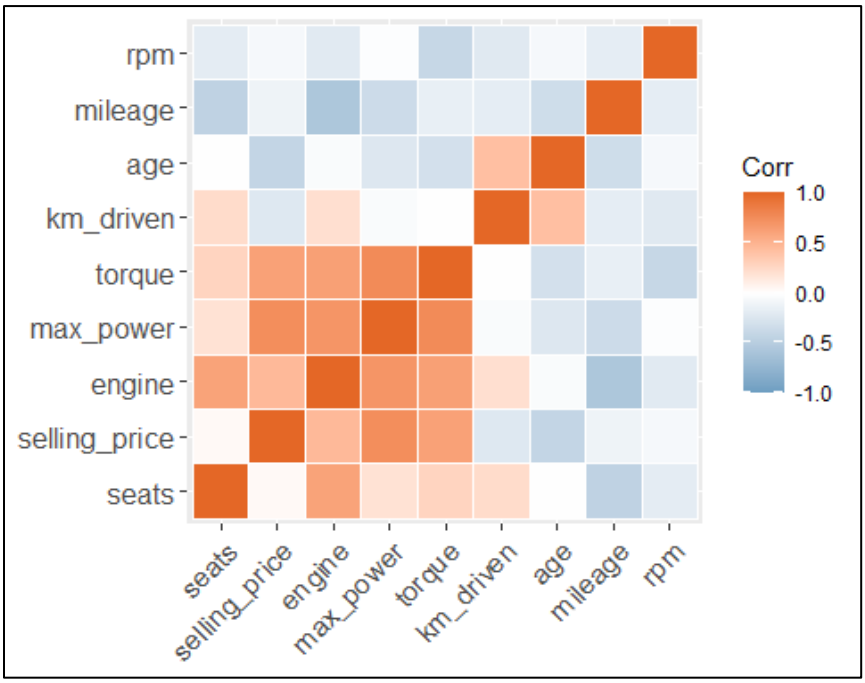
Observation - The features 'mileage' and 'KMs driven' can be considered as normally distributed and other features will need to be treated to make them normally distributed.

Multivariate Analysis

Correlation matrix of numeric variables

Understanding the correlation between the numeric features of the data set.

```
#install.packages("ggcorrplot")
library(ggcorrplot)

data_corr <- car_sales[, c("selling_price", "km_driven", "mileage", "engine",
                           "max_power", "seats", "torque", "rpm", "age")]
corr <- round(cor(data_corr), 2)

ggcorrplot(corr, hc.order = TRUE, outline.col = "white",
           ggtheme = ggplot2::theme_gray, colors = c("#6D9EC1", "white", "#E4
6726"))
```

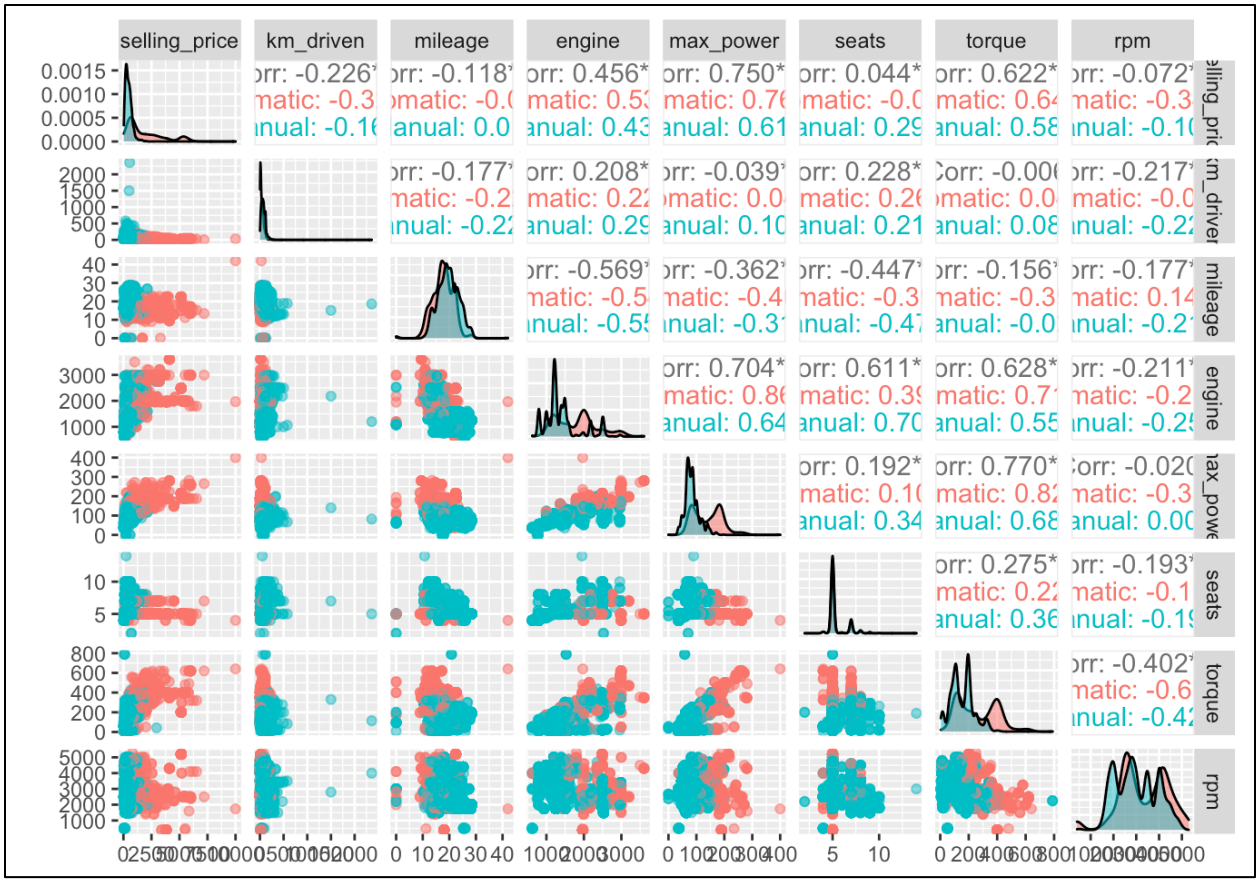


Observation - 'Max Power' seems to be highly correlated with 'Torque' and 'Selling Price'.

Pairplot of all the numeric variables

Using pairplots, we can figure out the correlations (Positive, Negative, No Correlation) between the attributes (features) of the data set.

```
#install.packages("GGally")
library(GGally)
ggpairs(car_sales, columns = c(2, 3, 8:13), aes(color = transmission, alpha =
0.5))
```



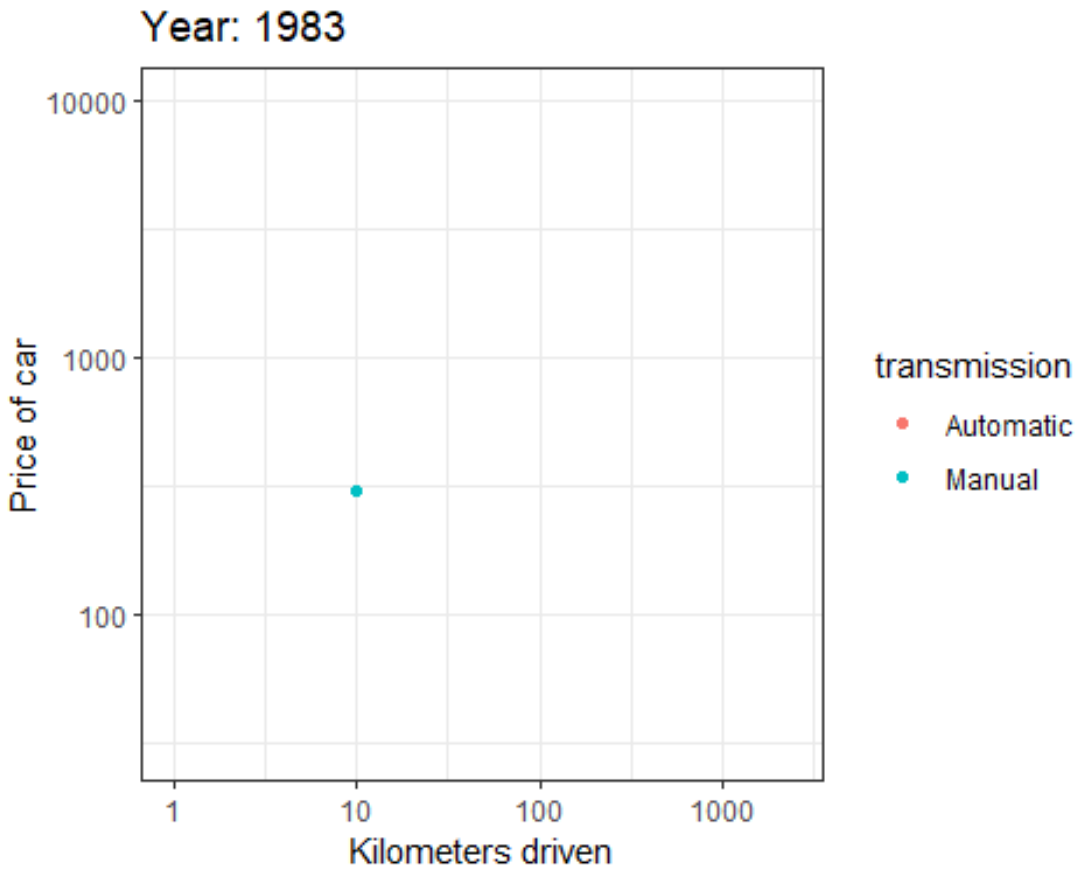
Observation - There are positive correlations between 'Max Power' and 'Engine'. No correlations can be found between 'Kms Driven' and other attributes.

Yearly trend of price

Scatterplot between Kms driven and Selling price over the year. The plot has been shown in the form of GIF instead of static plot.

```
#install.packages("gganimate")
#install.packages("gifski")
#install.packages("av")
library(ggplot2)
library(gganimate)

# Make a ggplot, but add frame=year: one image per year
gif1 <- ggplot(car_sales, aes(km_driven, selling_price, color = transmission)
) +
  geom_point() +
  scale_x_log10() +
  scale_y_log10() +
  theme_bw() +
  # gganimate specific bits:
```

```
  labs(title = 'Year: {frame_time}', x = 'Kilometers driven', y = 'Price of c
ar') +
  transition_time(as.integer(year)) +
  ease_aes('linear')

# Save as GIF:
animate(gif1, nframes = 100, fps = 5, end_pause = 20, renderer=gifski_rendere
r("test.gif"))
```



Observation -

1. The number of cars sold from 1983 till 2000 were very less and most of them were manual cars.

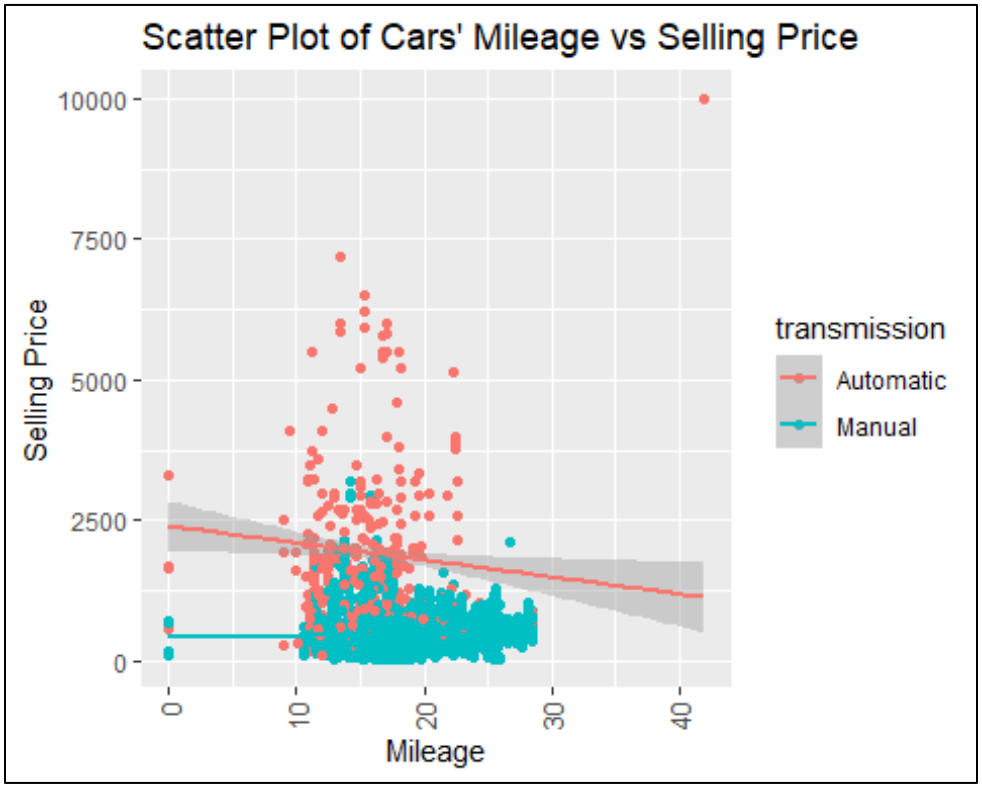
2. After 2005, the demand of automatic cars increased which peaked in 2015 overtaking manual cars.

Relationships between Attributes

Relationships between the attributes in our data set can be studied using scatterplots amongst them. The pattern of distribution of data set and the correlation between the variables can also be decoded from regression lines in the plot using GEOM_SMOOTH() function.

Scatter plot between 'Selling Price' and 'Mileage' distributed by transmission type

```
# Make a ggplot (Scatter plot) of variables 'Mileage' and 'Selling Price'.
ggplot(data = car_sales, aes(x = mileage, y = selling_price, color = transmis
sion)) +
  geom_point() +
  labs(title = "Scatter Plot of Cars' Mileage vs Selling Price", x = "Mileage
", y = 'Selling Price') +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 1
0)) +
  geom_smooth(method = "lm")
```

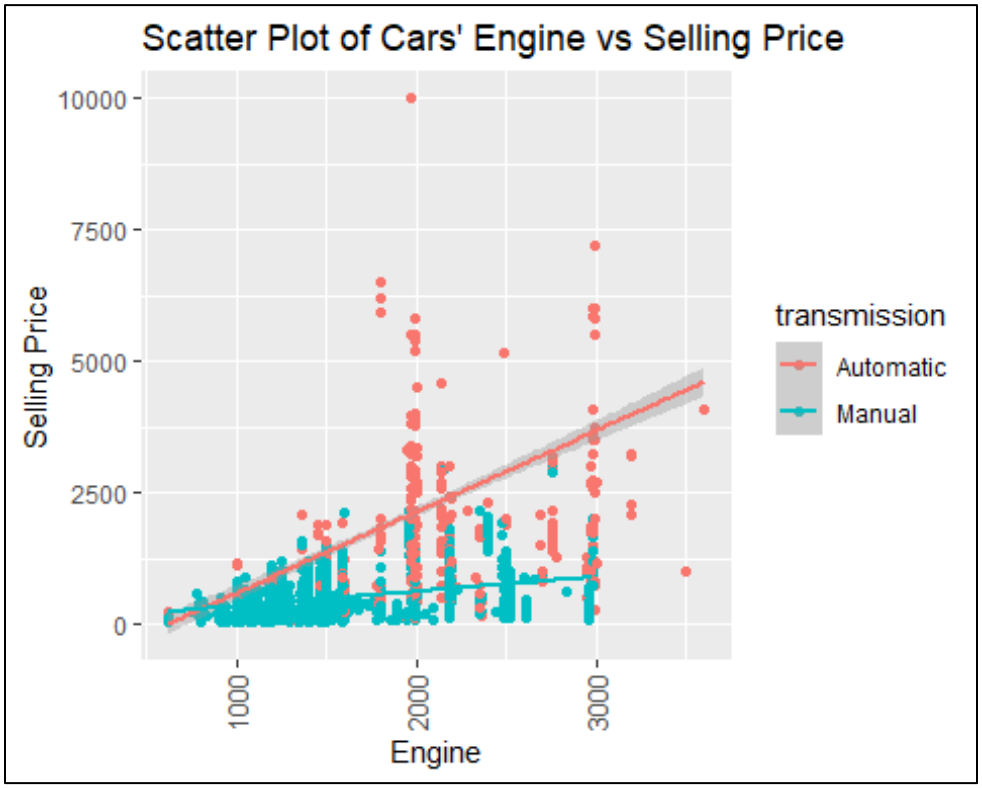



```
summary(lm(car_sales$selling_price ~ car_sales$mileage))

##
## Call:
## lm(formula = car_sales$selling_price ~ car_sales$mileage)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -1002.5  -377.9  -171.2    71.1  9901.7
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1107.550     44.651   24.80   <2e-16 ***
## car_sales$mileage  -24.029      2.261  -10.63   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 804.3 on 8031 degrees of freedom
## Multiple R-squared:  0.01387,    Adjusted R-squared:  0.01374
## F-statistic: 112.9 on 1 and 8031 DF,  p-value: < 2.2e-16
```

Scatter plot between 'Engine' and 'Selling Price' distributed by transmission type

```
# Make a ggplot (Scatter plot) of variables 'Engine' and 'Selling Price'.
ggplot(data = car_sales, aes(x = engine, y = selling_price, color = transmiss
ion)) +
  geom_point() +
  labs(title = "Scatter Plot of Cars' Engine vs Selling Price", x = "Engine",
y = 'Selling Price') +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 1
0)) +
  geom_smooth(method = "lm")
```

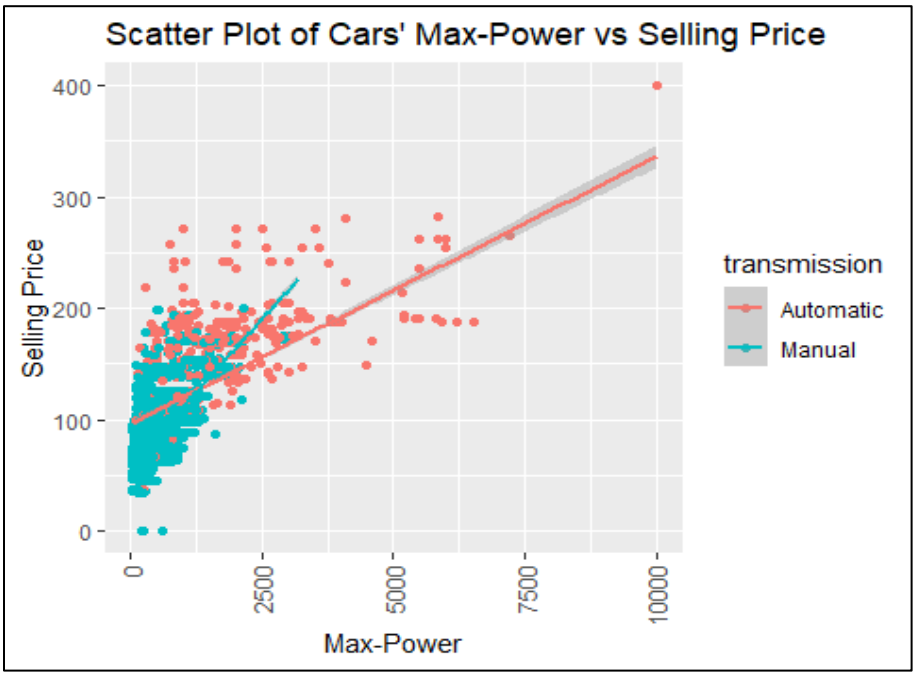



```
summary(lm(car_sales$selling_price ~ car_sales$engine))
```

```
##
## Call:
## lm(formula = car_sales$selling_price ~ car_sales$engine)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1650.8  -277.3   -51.1   125.3  8981.0
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -433.08399   24.77613  -17.48   <2e-16 ***
## car_sales$engine    0.73746    0.01606   45.91   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 720.8 on 8031 degrees of freedom
## Multiple R-squared:  0.2079, Adjusted R-squared:  0.2078
## F-statistic:  2108 on 1 and 8031 DF,  p-value: < 2.2e-16
```

Scatter plot between 'Max Power' and 'Selling Price' distributed by transmission type

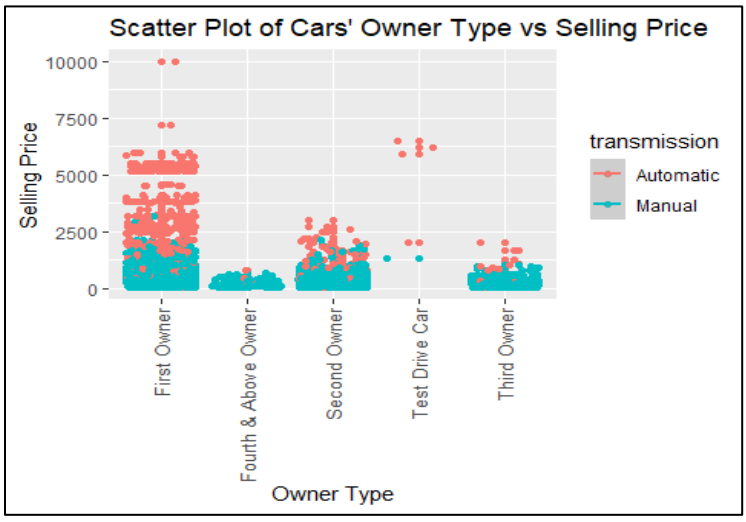
```
# Make a ggplot (Scatter plot) of variables 'Max Power' and 'Selling Price'.
ggplot(data = car_sales, aes(x = selling_price, y = max_power, color = transm
ission)) +
  geom_point() +
  labs(title = "Scatter Plot of Cars' Max-Power vs Selling Price", x = "Max-P
ower", y = 'Selling Price') +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 1
0)) +
  geom_smooth(method = "lm")
```



```
summary(lm(car_sales$selling_price ~ car_sales$max_power))

##
## Call:
## lm(formula = car_sales$selling_price ~ car_sales$max_power)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2733.0  -196.6     3.5   184.0  4238.1
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -916.502     16.485   -55.6   <2e-16 ***
## car_sales$max_power     17.052      0.168   101.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 536 on 8031 degrees of freedom
## Multiple R-squared:  0.562,  Adjusted R-squared:  0.5619
## F-statistic: 1.03e+04 on 1 and 8031 DF,  p-value: < 2.2e-16
```

Scatter plot between 'Selling Price' and 'Owner Type' distributed by transmission type

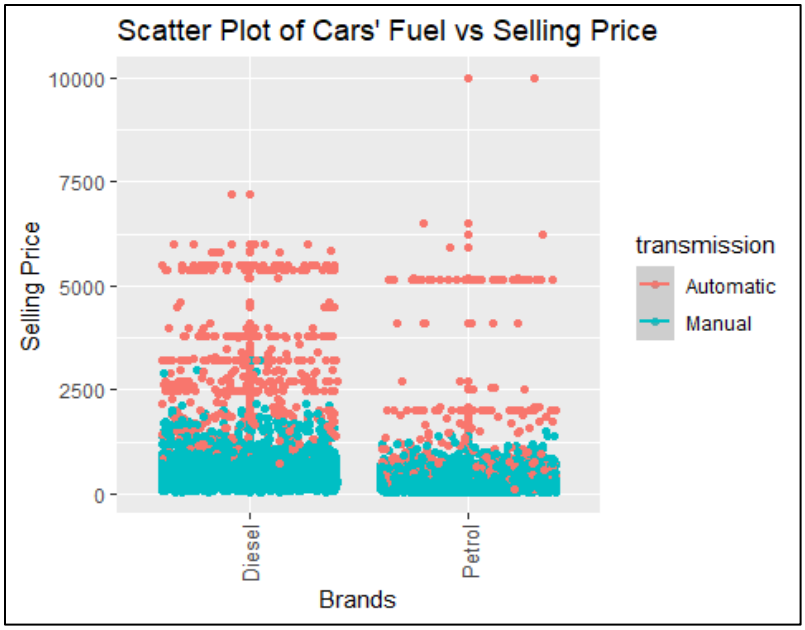
```
# Make a ggplot (Scatter plot) of variables and 'Owner Type' and 'Selling Pri
ce'.
ggplot(data = car_sales, aes(x = owner, y = selling_price, color = transmissi
on)) +
  geom_point() +
  geom_jitter() +
  labs(title = "Scatter Plot of Cars' Owner Type vs Selling Price", x = "Owne
r Type", y = 'Selling Price') +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 1
0)) +
  geom_smooth(method = "lm")
```



```
summary(lm(car_sales$selling_price ~ car_sales$owner))

##
## Call:
## lm(formula = car_sales$selling_price ~ car_sales$owner)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3053.8  -337.6  -162.6    34.3  9212.4
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                          787.60      10.75  73.246   <2e-16 *
**
## car_sales$ownerFourth & Above Owner  -560.29      60.65  -9.238   <2e-16 *
**
## car_sales$ownerSecond Owner          -391.70      20.19 -19.397   <2e-16 *
**
## car_sales$ownerTest Drive Car        3616.20     348.20  10.385   <2e-16 *
**
## car_sales$ownerThird Owner           -501.86      34.97 -14.352   <2e-16 *
**
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 778.2 on 8028 degrees of freedom
## Multiple R-squared:  0.07707,    Adjusted R-squared:  0.07661
## F-statistic: 167.6 on 4 and 8028 DF,  p-value: < 2.2e-16
```

Scatter plot between 'Fuel' and 'Selling Price' distributed by transmission type

```
# Make a ggplot (Scatter plot) of variables 'Fuel' and 'Selling Price'.
ggplot(data = car_sales, aes(x = fuel, y = selling_price, color = transmissio
n)) +
  geom_point() +
  geom_jitter() +
  labs(title = "Scatter Plot of Cars' Fuel vs Selling Price", x = "Brands", y
= 'Selling Price') +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 1
0)) +
  geom_smooth(method = "lm")
```

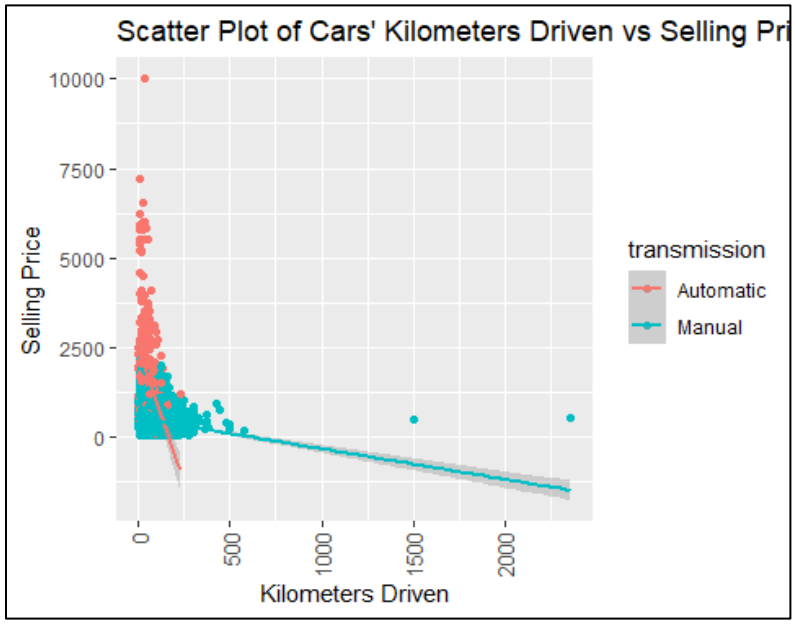



```
summary(lm(car_sales$selling_price ~ car_sales$fuel))

##
## Call:
## lm(formula = car_sales$selling_price ~ car_sales$fuel)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -751.5 -342.4 -172.4   58.5 9537.6
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             791.45      11.95    66.2   <2e-16 ***
## car_sales$fuelPetrol  -329.01      17.78   -18.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 793.2 on 8031 degrees of freedom
## Multiple R-squared:  0.04089,    Adjusted R-squared:  0.04077
## F-statistic: 342.4 on 1 and 8031 DF,  p-value: < 2.2e-16
```

Scatter plot between 'Kilometers Driven' and 'Selling Price' distributed by transmission type

```
# Make a ggplot (Scatter plot) of variables 'Kilometers Driven' and 'Selling
Price'.
ggplot(data = car_sales, aes(x = km_driven, y = selling_price, color = transm
ission)) +
  geom_point() +
  geom_jitter() +
```

```
  labs(title = "Scatter Plot of Cars' Kilometers Driven vs Selling Price", x
= "Kilometers Driven", y = 'Selling Price') +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 1
0)) +
  geom_smooth(method = "lm")
```




```
summary(lm(car_sales$selling_price ~ car_sales$km_driven))

##
## Call:
## lm(formula = car_sales$selling_price ~ car_sales$km_driven)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -806.5 -369.6 -185.1   38.1 9229.1
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         867.6437    13.9642   62.13   <2e-16 ***
## car_sales$km_driven  -3.2250     0.1554  -20.75   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 789 on 8031 degrees of freedom
## Multiple R-squared:  0.05088,    Adjusted R-squared:  0.05076
## F-statistic: 430.5 on 1 and 8031 DF,  p-value: < 2.2e-16
```

Hypothesis Testing

Hypothesis testing is used to validate an assumption regarding the population parameter, which can be generalized. Based on the sample size, distribution and the sample statistics know, different types of hypothesis tests can be employed. The general steps involved in hypothesis testing are:- 1. Formulate the NULL and alternative hypothesis 2. Plan the test to be performed and decide the critical value 3. Perform the test and obtain the test statistics 4. Reject the NULL hypothesis or state that the null hypothesis is plausible

One Sample t-Tests

One Sample t-Test of the kilometers driven by 'Individual' Seller-type

NULL HYPOTHESIS, H0 : True Mean of kilometers driven by 'Individual' seller-type is equal to the overall average.
ALTERNATE HYPOTHESIS, H1 : True Mean of kilometers driven by 'Individual' seller-type is greater than the overall average.
ALTERNATIVE : Greater

```
# Average of 'Km_driven' across the data set.
mean_km_driven <- mean(car_sales$km_driven)

# One-Sample t-test for kilometers driven by 'Individual' Seller type with it
s mean.
ttest <- t.test(car_sales$km_driven[car_sales$seller_type == "Individual"], m
u = mean_km_driven, alternative = "greater", conf.level = .95)

ttest

##
##  One Sample t-test
##
## data:  car_sales$km_driven[car_sales$seller_type == "Individual"]
## t = 7.2395, df = 6672, p-value = 2.503e-13
## alternative hypothesis: true mean is greater than 69.73881
## 95 percent confidence interval:
##  73.79173      Inf
## sample estimates:
## mean of x
##  74.98352

format(ttest$p.value, scientific = FALSE)

## [1] "0.0000000000002503328"
```

DEGREE OF FREEDOM : 6672
P-VALUE : 2.5033283^{-13}

Observation - Since, the p-value of our One-Sample t-Test is 2.5033283^{-13}, which is less than our alpha = 0.05, we reject the Null Hypothesis of the test. This means that we

have sufficient evidence to say that Population True Mean of the kilometers driven by 'Individual' seller-type in the data set is greater than the True Mean of kilometers driven by all seller-types in the data set.

One Sample t-Test of the selling price of cars sold by 'Dealer' Seller-type

NULL HYPOTHESIS, H0 : True Mean of selling price of cars sold by 'Dealer' seller-type is equal to the overall average.
ALTERNATE HYPOTHESIS, H1 : True Mean of selling price of cars sold by 'Dealer' seller-type is greater than the overall average.
ALTERNATIVE : Greater

```
# Average of 'selling_price' across the data set.
mean_selling_price <- mean(car_sales$selling_price)

# One-Sample t-test for selling price of cars sold by 'Dealer' Seller type wi
th its mean.
ttest <- t.test(car_sales$selling_price[car_sales$seller_type == "Dealer"], m
u = mean_selling_price, alternative = "greater", conf.level = .95)

ttest

##
##  One Sample t-test
##
## data:  car_sales$selling_price[car_sales$seller_type == "Dealer"]
## t = 17.151, df = 1123, p-value < 2.2e-16
## alternative hypothesis: true mean is greater than 642.7362
## 95 percent confidence interval:
##  1368.685      Inf
## sample estimates:
## mean of x
##  1445.764

format(ttest$p.value, scientific = FALSE)

## [1] "0.00000000000000000000000000000000000000000000000000000000000004835655"
```

DEGREE OF FREEDOM : 1123
P-VALUE : 4.8356552^{-59}

Observation - Since, the p-value of our One-Sample t-Test is 4.8356552^{-59}, which is less than our alpha = 0.05, we reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that Population True Mean of the selling price of cars sold by 'Dealer' seller-type in the data set is greater than the True Mean of selling price of cars sold by all seller-types in the data set.

The feature 'Owner' contains values like 'First Owner, Second Owner, Third Owner, Fourth Owner, Test Drive Cars'. We removed the 'Test Drive' cars from the data set to make it

more consumer centric and also because there record quantity is very less when compared with others in the data set.

```
# Remove cars belonging to 'Test Drive' owners.
car_sales <- filter(car_sales, !owner %in% c("Test Drive Car"))
```

One Sample t-Test of the kilometers driven by 'First Owners' of the cars

NULL HYPOTHESIS, H0 : True Mean of kilometers driven by 'First Owner' owner-type is equal to the overall average.
ALTERNATE HYPOTHESIS, H1 : True Mean of kilometers driven by 'First Owner' owner-type is less than the overall average.
ALTERNATIVE : Less

```
# Average of 'Km_driven' across the data set.
mean_km_driven <- mean(car_sales$km_driven)

# One-Sample t-test for kilometers driven by 'First Owners' of the cars with
its mean.
ttest <- t.test(car_sales$km_driven[car_sales$owner == "First Owner"], mu = m
ean_km_driven, alternative = "less", conf.level = .95)

ttest

##
##  One Sample t-test
##
## data:  car_sales$km_driven[car_sales$owner == "First Owner"]
## t = -18.941, df = 5237, p-value < 2.2e-16
## alternative hypothesis: true mean is less than 69.77313
## 95 percent confidence interval:
##      -Inf 58.43108
## sample estimates:
## mean of x
##  57.35225

format(ttest$p.value, scientific = FALSE)

## [1] "0.0000000000000000000000000000000000000000000000000000000000000000000
00000000009682691"
```

DEGREE OF FREEDOM : 5237
P-VALUE : 9.6826906^{-78}

Observation - Since, the p-value of our One-Sample t-Test is 9.6826906^{-78}, which is less than our alpha = 0.05, we reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that Population True Mean of the kilometers driven by 'First Owner' owner-type in the data set is less than the True Mean of kilometers driven by 'First Owner' owner-types in the data set.

One Sample t-Test of the selling price of cars sold owned by 'First Owners' of the cars

NULL HYPOTHESIS, H0 : True Mean of selling price of cars sold by 'First Owner' owner-type is equal to the overall average.
ALTERNATE HYPOTHESIS, H1 : True Mean of selling price of cars sold by 'First Owner' owner-type is greater than the overall average.
ALTERNATIVE : Greater

```
# Average of 'selling_price' across the data set.
mean_selling_price <- mean(car_sales$selling_price)

# One-Sample t-test for selling price of cars sold owned by 'First Owners' of
the cars with its mean.
ttest <- t.test(car_sales$selling_price[car_sales$owner == "First Owner"], mu
= mean_selling_price, alternative = "greater", conf.level = .95)

ttest

##
##  One Sample t-test
##
## data:  car_sales$selling_price[car_sales$owner == "First Owner"]
## t = 11.39, df = 5237, p-value < 2.2e-16
## alternative hypothesis: true mean is greater than 640.3937
## 95 percent confidence interval:
##  766.3377      Inf
## sample estimates:
## mean of x
##  787.5987

print(paste("P-value :", format(ttest$p.value, scientific = FALSE)))

## [1] "P-value : 0.000000000000000000000000000005204048"
```

DEGREE OF FREEDOM : 5237
P-VALUE : 5.2040483^{-30}

Observation - Since, the p-value of our One-Sample t-Test is 5.2040483^{-30}, which is less than our alpha = 0.05, we reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that Population True Mean of the selling price of cars sold by 'First Owner' owner-type in the data set is greater than the True Mean of selling price of cars sold by all owner-type in the data set.

Two Sample t-Tests

Two Sample t-Test to test equality of the kilometers driven by 'Small & Medium' Car-types

NULL HYPOTHESIS, H0 : Mean kilometers driven for small cars (car_type = small) is equal to the kilometers driven for medium cars. mean(small car_type) = mean(medium car_type)
ALTERNATE HYPOTHESIS, H1 : Mean kilometers driven for small cars (car_type = small) is not equal to the kilometers driven for medium cars. mean(small car_type) !=

mean(medium car_type)
ALTERNATIVE : Two.Sided

```
# Two-Sample t-test to test equality of the kilometers driven by 'Small & Med
ium' Car-types.
ttest <- t.test(car_sales$km_driven[car_sales$car_type == "Small"], car_sales
$km_driven[car_sales$car_type == "Medium"], alternative = "two.sided", conf.l
evel = .95)

ttest

##
##  Welch Two Sample t-test
##
## data:  car_sales$km_driven[car_sales$car_type == "Small"] and car_sales$km
_driven[car_sales$car_type == "Medium"]
## t = -16.422, df = 1998.3, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -33.83369 -26.61471
## sample estimates:
## mean of x mean of y
##  63.33297  93.55717

format(ttest$p.value, scientific = FALSE)

## [1] "0.000000000000000000000000000000000000000000000000000006046862"
```

DEGREE OF FREEDOM : 1998.3096793
P-VALUE : 6.0468621^{-57}

Observation - Since, the p-value of our Two-Sample t-Test is 6.0468621^{-57}, which is less than our alpha = 0.05, we reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that Mean kilometers driven for small cars (car_type = small) is not equal to the kilometers driven for medium cars in the data set.

Two Sample t-Test to test equality of the kilometers driven by 'Dealer & Trustmark Dealer' Seller-types

NULL HYPOTHESIS, H0 : Mean kilometers driven for dealers (seller_type = dealer) is equal to the kilometers driven for trustmark dealer. mean(dealer seller_type) = mean(trustmark dealer seller_type)
ALTERNATE HYPOTHESIS, H1 : Mean kilometers driven for dealers (seller_type = dealer) is not equal to the kilometers driven for trustmark dealer. mean(dealer seller_type) != mean(trustmark dealer seller_type)
ALTERNATIVE : Two.Sided

```
# Two-Sample t-Test to test equality of the kilometers driven by 'Dealer & Tr
ustmark Dealer' Seller-types.
ttest <- t.test(car_sales$km_driven[car_sales$seller_type == "Dealer"], car_s
ales$km_driven[car_sales$seller_type == "Trustmark Dealer"], alternative = "t
```

```
wo.sided", conf.level = .95)

ttest

##
##  Welch Two Sample t-test
##
## data:  car_sales$km_driven[car_sales$seller_type == "Dealer"] and car_sale
s$km_driven[car_sales$seller_type == "Trustmark Dealer"]
## t = 1.1005, df = 377.67, p-value = 0.2718
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.767033  6.258865
## sample estimates:
## mean of x mean of y
##  44.50460  42.25869

format(ttest$p.value, scientific = FALSE)

## [1] "0.2718353"
```

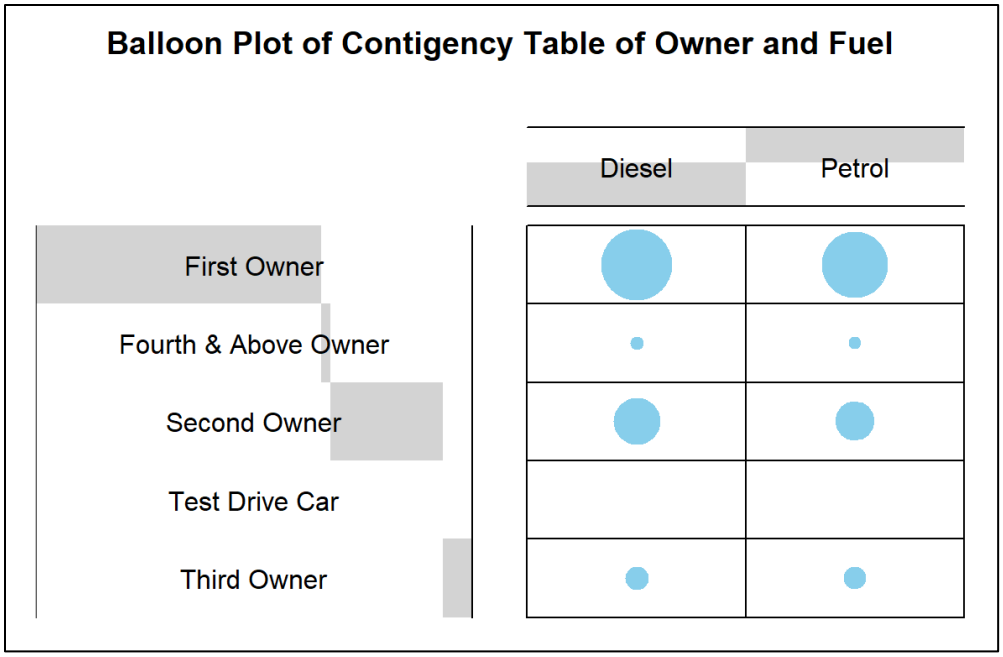
DEGREE OF FREEDOM : 377.6718623
P-VALUE : 0.2718353

Observation - Since, the p-value of our Two-Sample t-Test is 0.2718353, which is not less than our alpha = 0.05, we cannot reject the Null Hypothesis of the test. This means that we do not have sufficient evidence to say that Mean kilometers driven for dealers (seller_type = dealer) is not equal to the kilometers driven for trustmark dealer in the data set.

Chi Square Test

Chi Square test to identify association between variables 'Owner Type & Fuel'

```
# Chi Square test to verify the association between the variables Owner Type
& Fuel
#chisq_test <-
#format(chisq_test$p.value, scientific = FALSE)


# 1. convert the data as a table
car_sales_tab <- table(car_sales$owner, car_sales$fuel)
dt <- as.table(as.matrix(car_sales_tab))

# 2. Graph
balloonplot(t(dt), main ="Balloon Plot of Contigency Table of Owner and Fuel"
, xlab ="", ylab="",
            label = FALSE, show.margins = FALSE)
```



NULL HYPOTHESIS, H0 : There is no relationship between Owner-Type and Fuel-Type of cars. The variables are independent.
ALTERNATE HYPOTHESIS, H1 : There exists some relationship between Owner-Type and Fuel-Type of cars. The variables are dependent.

```
# Chi Square test to verify the association between the variables Owner Type
& Fuel
chisq_test <- chisq.test(car_sales_tab)
chisq_test

##
##  Pearson's Chi-squared test
##
## data:  car_sales_tab
## X-squared = NaN, df = 4, p-value = NA

format(chisq_test$p.value, scientific = FALSE)

## [1] "NaN"
```

DEGREE OF FREEDOM : 4
P-VALUE : NaN

One Way ANOVA test

One Way ANOVA between categorical Owner and Fuel and other continuous variables

Categorical variables:

- fuel
- owner

Continuous variables:

- max_power
- engine
- mileage
- km_driven

NULL HYPOTHESIS, H0 : True Means of the Fuel and Max-Power groups are same.
ALTERNATE HYPOTHESIS, H1 : True Means of the Fuel and Max-Power groups are not same.

```
# One way ANOVA test to test the association between fuel and max_power
one.way1 <- aov(max_power ~ fuel, data = car_sales)
summary(one.way1)

##               Df  Sum Sq Mean Sq F value Pr(>F)
## fuel           1  904406  904406   785.1 <2e-16 ***
## Residuals   8026 9246178    1152
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Observation - Since, the p-value of our ANOVA test is less than our alpha = 0.05, we can reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that True Means of the Fuel and Max-Power groups are not same in the data set.

NULL HYPOTHESIS, H0 : True Means of the Fuel and Engine groups are same.
ALTERNATE HYPOTHESIS, H1 : True Means of the Fuel and Engine groups are not same.

```
# One way ANOVA test to test the association between fuel and engine
one.way2 <- aov(engine ~ fuel, data = car_sales)
summary(one.way2)

##               Df    Sum Sq   Mean Sq F value Pr(>F)
## fuel           1 5.038e+08 503807702    2679 <2e-16 ***
## Residuals   8026 1.509e+09    188053
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Observation - Since, the p-value of our ANOVA test is less than our alpha = 0.05, we can reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that True Means of the Fuel and Engine groups are not same in the data set.

NULL HYPOTHESIS, H0 : True Means of the Fuel and Mileage groups are same.
ALTERNATE HYPOTHESIS, H1 : True Means of the Fuel and Mileage groups are not same.

```
# One way ANOVA test to test the association between fuel and mileage
one.way3 <- aov(mileage ~ fuel, data = car_sales)
summary(one.way3)

##               Df Sum Sq Mean Sq F value   Pr(>F)
## fuel           1    666   665.7   42.48 7.58e-11 ***
## Residuals   8026 125775    15.7
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Observation - Since, the p-value of our ANOVA test is less than our alpha = 0.05, we can reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that True Means of the Fuel and Mileage groups are not same in the data set.

NULL HYPOTHESIS, H0 : True Means of the Fuel and Kilometers-Driven groups are same.
ALTERNATE HYPOTHESIS, H1 : True Means of the Fuel & Kms-Driven are not same.

```
# One way ANOVA test to test the association between fuel and km_driven
one.way4 <- aov(km_driven ~ fuel, data = car_sales)
summary(one.way4)

##               Df   Sum Sq Mean Sq F value Pr(>F)
## fuel           1  1888007 1888007   634.9 <2e-16 ***
## Residuals   8026 23867156    2974
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Observation - Since, the p-value of our ANOVA test is less than our alpha = 0.05, we can reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that True Means of the Fuel and Kilometers-Driven groups are not same in the data set.

NULL HYPOTHESIS, H0 : True Means of the Owner and Max-Power groups are same.
ALTERNATE HYPOTHESIS, H1 : True Means of the Owner and Max-Power groups are not same.

```
# One way ANOVA test to test the association between owner and max_power
one.way5 <- aov(max_power ~ owner, data = car_sales)
summary(one.way5)

##               Df  Sum Sq Mean Sq F value Pr(>F)
## owner          3  155302   51767   41.56 <2e-16 ***
## Residuals   8024 9995283    1246
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Observation - Since, the p-value of our ANOVA test is less than our alpha = 0.05, we can reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that True Means of the Owner and Max-Power groups are not same in the data set.

NULL HYPOTHESIS, H0 : True Means of the Owner and Engine groups are same.
ALTERNATE HYPOTHESIS, H1 : True Means of the Owner and Engine groups are not same.

```
# One way ANOVA test to test the association between owner and engine
one.way6 <- aov(engine ~ owner, data = car_sales)
summary(one.way6)

##               Df    Sum Sq Mean Sq F value Pr(>F)
## owner          3 9.721e+05  324021   1.292  0.275
## Residuals   8024 2.012e+09  250766
```

Observation - Since, the p-value of our ANOVA test is not less than our alpha = 0.05, we can not reject the Null Hypothesis of the test. This means that we do not have sufficient evidence to say that True Means of the Owner and Engine groups are not same in the data set.

NULL HYPOTHESIS, H0 : True Means of the Owner and Mileage groups are same.
ALTERNATE HYPOTHESIS, H1 : True Means of the Owner and Mileage groups are not same.

```
# One way ANOVA test to test the association between owner and mileage
one.way7 <- aov(mileage ~ owner, data = car_sales)
summary(one.way7)

##               Df Sum Sq Mean Sq F value Pr(>F)
## owner          3   4347  1449.1   95.23 <2e-16 ***
## Residuals   8024 122093    15.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Observation - Since, the p-value of our ANOVA test is less than our alpha = 0.05, we can reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that True Means of the Owner and Mileage groups are not same in the data set.

NULL HYPOTHESIS, H0 : True Means of the Owner and Kms-Driven groups are same.
ALTERNATE HYPOTHESIS, H1 : True Means of the Owner and Km-Driven are not same.

```
# One way ANOVA test to test the association between owner and km_driven
one.way8 <- aov(km_driven ~ owner, data = car_sales)
summary(one.way8)

##               Df   Sum Sq Mean Sq F value Pr(>F)
## owner          3  2406771  802257   275.7 <2e-16 ***
## Residuals   8024 23348392    2910
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Observation - Since, the p-value of our ANOVA test is less than our alpha = 0.05, we can reject the Null Hypothesis of the test. This means that we have sufficient evidence to say that True Means of the Owner and Kilometers-Driven groups are not same in the data set.

Regression Model

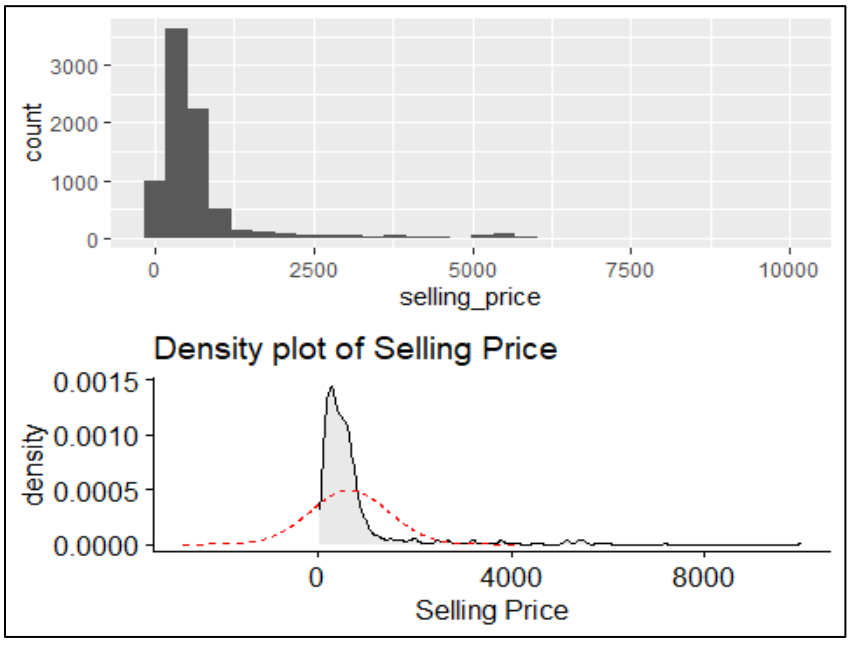
Linear regression models the relationship between two variables by fitting a linear equation to observed data. One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable. Linear regression uses the equation y = B0 + B1*x to model the data. We are going to model the price of the homes/apartments listed on Airbnb with the other variables.

Assumptions of Linear Model

Before building a linear regression model, these assumptions need to be verified: 1. Linear relationship: There exists a linear relationship between the independent variable, x, and the dependent variable, y. 2. Independence: The residuals are independent. In particular, there is no correlation between consecutive residuals in time series data. 3. Homoscedasticity: The residuals have constant variance at every level of x. 4. Normality: The residuals of the model are normally distributed.

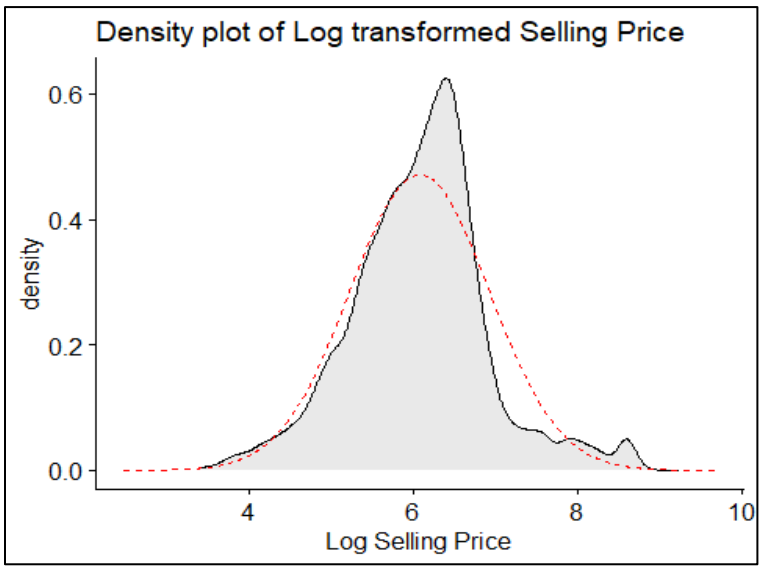
We have already tested the linearity and dependence of the variables. We are going to test the normality of the data now, before building the linear regression model.

```
p1 <- ggplot(data = car_sales) +
  geom_histogram(mapping = aes(selling_price))

p2 <- ggdensity(car_sales$selling_price,  fill = 'lightgray') +
  stat_overlay_normal_density(color = 'red', linetype = 'dashed') +
  xlab(label = 'Selling Price') +
  labs(title = 'Density plot of Selling Price')

gridExtra::grid.arrange(p1, p2)
```



The Selling price variable needs to be log transformed to convert it into normal distribution.

```
ggdensity(log(car_sales$selling_price),  fill = 'lightgray') +
  stat_overlay_normal_density(color = 'red', linetype = 'dashed') +
  xlab(label = 'Log Selling Price') +
  labs(title = 'Density plot of Log transformed Selling Price')
```



Transformed the selling price

```
car_sales$log_selling_price <- log(car_sales$selling_price)

names_vec <- c('log_selling_price', 'fuel', 'seller_type', 'transmission', 'o
wner', 'engine', 'max_power', 'torque', 'age', 'brand')
car_sales_df <- car_sales[, names_vec]
```

Building Linear Regression model

The lm() function is used to build the linear regression model. However, to obtain the best model, we have used an approach called stepwise linear regression. Stepwise linear regression is a method of regressing multiple variables while simultaneously removing those that aren't important. Stepwise regression essentially does multiple regression a number of times, each time removing the weakest correlated variable. At the end we are left with the variables that explain the distribution best.

```
# Fit the full model
full.model <- lm(log_selling_price ~., data = car_sales_df)
# Stepwise regression model
step.model <- stepAIC(full.model, direction = "both",
                      trace = FALSE)
summary(step.model)
```

```
##
## Call:
## lm(formula = log_selling_price ~ fuel + seller_type + transmission +
##     owner + engine + max_power + age + brand, data = car_sales_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2792 -0.1476  0.0110  0.1621  3.6919
##
## Coefficients:
##                               Estimate Std. Error  t value Pr(>|t|)
## (Intercept)                  6.300e+00  1.311e-01   48.048  < 2e-16 ***
## fuelPetrol                  -2.344e-01  7.338e-03  -31.942  < 2e-16 ***
## seller_typeIndividual       -4.371e-02  9.469e-03   -4.616 3.96e-06 ***
## seller_typeTrustmark Dealer -1.794e-02  1.988e-02   -0.902 0.366895
## transmissionManual          -7.663e-02  1.184e-02   -6.475 1.00e-10 ***
## ownerFourth & Above Owner   -1.567e-01  2.098e-02   -7.469 8.92e-14 ***
## ownerSecond Owner           -9.411e-02  7.423e-03  -12.679  < 2e-16 ***
## ownerThird Owner            -1.233e-01  1.256e-02   -9.820  < 2e-16 ***
## engine                       2.120e-04  1.238e-05   17.122  < 2e-16 ***
## max_power                    8.173e-03  1.659e-04   49.257  < 2e-16 ***
## age                         -1.142e-01  8.998e-04 -126.862  < 2e-16 ***
## brandAshok Leyland          -3.912e-01  2.875e-01   -1.361 0.173609
## brandAudi                    8.296e-02  1.370e-01    0.606 0.544704
## brandBMW                     3.314e-01  1.326e-01    2.498 0.012501 *
## brandChevrolet              -5.544e-01  1.301e-01   -4.262 2.05e-05 ***
## brandDaewoo                  1.008e-01  1.968e-01    0.512 0.608598
## brandDatsun                 -5.262e-01  1.332e-01   -3.950 7.88e-05 ***
## brandFiat                   -4.158e-01  1.344e-01   -3.094 0.001979 **
## brandForce                  -3.854e-01  1.664e-01   -2.317 0.020554 *
## brandFord                   -3.092e-01  1.297e-01   -2.384 0.017133 *
## brandHonda                  -1.446e-01  1.298e-01   -1.114 0.265181
## brandHyundai                -2.106e-01  1.294e-01   -1.628 0.103490
## brandIsuzu                  -2.663e-01  1.732e-01   -1.537 0.124229
## brandJaguar                  2.330e-01  1.339e-01    1.740 0.081888 .
## brandJeep                   -1.528e-01  1.382e-01   -1.106 0.268913
## brandKia                    -9.918e-02  1.825e-01   -0.543 0.586860
## brandLand Rover              6.371e-01  1.669e-01    3.818 0.000136 ***
## brandLexus                   4.287e-01  1.383e-01    3.100 0.001941 **
## brandMahindra               -2.666e-01  1.292e-01   -2.064 0.039090 *
## brandMaruti                 -1.581e-01  1.292e-01   -1.223 0.221325
## brandMercedes-Benz           2.177e-01  1.347e-01    1.616 0.106111
## brandMG                      1.498e-01  1.975e-01    0.759 0.448033
## brandMitsubishi             -3.897e-02  1.461e-01   -0.267 0.789595
## brandNissan                 -2.540e-01  1.322e-01   -1.922 0.054663 .
## brandOpel                    5.507e-02  2.875e-01    0.192 0.848119
## brandPeugeot                 1.465e-01  2.876e-01    0.509 0.610621
## brandRenault                -2.865e-01  1.303e-01   -2.198 0.027957 *
## brandSkoda                  -2.496e-01  1.315e-01   -1.898 0.057768 .
## brandTata                   -6.239e-01  1.293e-01   -4.824 1.43e-06 ***
## brandToyota                  2.534e-02  1.295e-01    0.196 0.844883
## brandVolkswagen             -2.798e-01  1.305e-01   -2.144 0.032066 *
## brandVolvo                   2.295e-01  1.340e-01    1.713 0.086750 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2569 on 7986 degrees of freedom
## Multiple R-squared:  0.9065, Adjusted R-squared:  0.906
## F-statistic:  1889 on 41 and 7986 DF,  p-value: < 2.2e-16
```

Conclusion

The data set of car sales by CarDekho.com has provided various insights about the types of cars sold in the car industry and the patterns between them. The data set contains 8128 data points along with 13 features related to car details, engine details, and sale details.

Data cleaning and data pre-processing are important to prepare the data in the correct format before building the regression model. We extracted numerical values from text fields like engine, mileage, torque, along with brand information from car name.

Unnecessary data that skew the results were also filtered out.

To ensure that the numerical variables are not skewed, it's crucial to remove or impute the missing values and outliers. We imputed the missing values using kNN imputation method.

Feature engineering and exploratory data analysis were performed to gather more meaningful information from the data.

Derived variables were created using the existing features and skewed variables were scaled.

Apart from this, various data visualization, like box plot, frequency plot, histogram, pair plot, correlation matrix and scatter plot were created to understand the uni-variate distribution and multi-variate relationship of the data.

Once the data cleaning and exploratory analysis is performed, hypothesis testing is performed to validate certain assumptions on the sample data.

In this assignment, we used one sample t-test and two sample t-test to compare the variables 'km_driven' and 'selling_price' with the overall sample average and compare across two groups respectively. Based on the exploratory analysis performed earlier, we wanted to validate the following hypotheis using t-test:

1. True Mean of kilometers driven by 'Individual' seller-type is greater than the overall average
2. True Mean of selling price of cars sold by 'Dealer' seller-type is greater than the overall average
3. True Mean of kilometers driven by 'First Owner' owner-type is less than the overall average
4. True Mean of selling price of cars sold by 'First Owner' owner-type is greater than the overall average
5. Mean kilometers driven for small cars is not equal to the kilometers driven for medium cars
6. Mean kilometers driven for dealers is not equal to the kilometers driven for trustmark dealer

In the first five test, based on the t-statistic and p-value we obtained sufficient evidence to reject the NULL hypothesis, however in the last test, we did not obtain enough evidence to reject the NULL hypothesis.

We also performed Chi-Square Test and One-way ANOVA test on categorical and continuous variables in our data set. The p-value of all the tests except the one where groups of Owner and Engine were taken came out to be very less than our assumed alpha = 0.05. Therefore, we were able to reject the Null Hypotheses in all, but one tests successfully.

Lastly, we performed Regression Analysis on our data set to model the prices of cars listed on CarDekho.com. We log transformed the Selling price variable to convert it into normal distribution and used lm() function to build the linear regression model.

Bibliography

Vehicle dataset. (2020, October 24). Kaggle. https://www.kaggle.com/nehalbirla/vehicle-dataset-from-cardekho

Xie, Y. C. D. (2021, October 7). R Markdown Cookbook. R Markdown. Retrieved October 30, 2021, from https://bookdown.org/yihui/rmarkdown-cookbook/

Bluman, A. (2017). Elementary Statistics: A Step By Step Approach (10th ed.). McGraw-Hill Education.

Kabacoff, R., I. (2022). R in Action, Third Edition. Manning.

CarDekho. (2021). About Us | CarDekho.com. https://www.cardekho.com/info/about_us

F. (2021, April 2). tidyverse in r – Complete Tutorial. R-Bloggers. Retrieved October 30, 2021, from https://www.r-bloggers.com/2021/04/tidyverse-in-r-complete-tutorial/

D. (2021a, March 22). Data Analytics for Car Dealers. Automated Metrics. https://www.automatedmetrics.io/data-analytics-for-car-dealers/

Swaminathan, S. (2019, January 18). *Linear Regression — Detailed View - Towards Data Science*. Medium. Retrieved December 13, 2021, from https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86

Appendix

The RMD file of the analysis is included with the analysis report.

Final_Project_Akash_Harshit_Varun.Rmd