FINAL PROJECT REPORT

PROJECT ASSIGNMENT 4: DATA ANALYTICS RESEARCH PROJECT

AIT-INFS-580 ANALYTICS – BIG DATA TO INFORMATION

FACULTY: HARRY J FOXWELL

SUBMITTED BY: KETAKI SHAH

GNUMBER: G01396496

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

ABSTRACT

A car price forecast has been a popular study topic since it demands significant effort and expertise from a field specialist. For a dependable and accurate forecast, a large number of unique qualities are analyzed. The dataset's purpose is to allow the development of a linear regression model that can accurately predict car prices based on these attributes for the Chinese automobile company Geely Auto, which aspires to enter the US market by establishing a manufacturing unit there and producing cars locally to compete with their US and European counterparts. The data used for prediction was collected from an automobile consulting company. We want to understand the elements influencing automobile pricing in the United States since they may differ significantly from those in China. The linear regression model is used to model the price of cars with independent variables, allowing management to manipulate design, business strategy, etc. to meet certain price levels.

1. INTRODUCTION

Car price prediction is intriguing, and Geely6 Auto is a Chinese automaker looking to enter the US market. They have hired an automobile consulting firm to help them understand the factors that influence car price in the American market. This study investigates the relationship between car pricing, fuel type, aspirations, wheelbase, and car length.

Proper car price prediction necessitates specialist knowledge because the price is normally determined by a number of different features and circumstances. The most important ones are usually the brand and model, horsepower, and mileage. Because of the frequent variations in the price of gasoline, the fuel type used in the automobile as well as fuel consumption per mile have a significant impact on the price of a car. The price of a car is also affected by factors such as its exterior color, door number, transmission type, dimensions, safety, air conditioning, interior, and whether or not it has GPS. Several of the features stated above are taken into account for analysis. In this article, several strategies and techniques are used to improve the precision of car price forecast. Linear regression methods are used to determine the relationship between car price and independent factors.

This paper is organized in the following manner:
Part 2 includes relevant work in the realm of car price prediction. The data analysis id explained in Section 3, 4 and 5. Part 6 gives the conclusion, part 7 gives the future scope and finally, in section 8, the work is summarized with references.

1.1 DATASET SELECTED

- **Car Price Prediction Database**

An automobile consulting firm gathered this vast dataset of various types of cars across the American market to examine the elements that influence car pricing. The data set offers information about the numerous elements that influence the price of a specific car. There are a total of 26 columns/attributes, including the automobile name, fuel type, engine location, horsepower, peak rpm, city mpg, and so on, as well as the price of the output variable/attribute. There are 205 observations in total.

The column "Price" is the target variable, and the rest of the columns are independent variables. The independent variables are again divided into Categorical and Numerical variables.

Numerical variables: [‘car_ID ‘, ‘wheelbase’, ‘carlength’, ‘carwidth’, ‘carheight’, ‘curbweight’, ‘enginesize’, ‘boreratio','stroke’, ‘compressionratio’, ‘horsepower’, ‘peakrpm’, ‘citympg’, ‘highwaympg’]

Categorical variables: [‘symboling’, ‘CompanyName’, ‘fueltype’, ‘aspiration’, ‘doornumber’, ‘carbody’, ‘drivewheel’, ‘enginelocation’, ‘enginetype’, ‘cylindernumber’, 'fuelsystem', ‘car_name’]

Attributes information and NOIR analysis:

Nominal and Ordinal Datatypes are considered as categorical datatype.

NOMINAL: A group of entities that can be identified by their names or categories and have distinct values, often known as binomial characteristics. The order is unimportant.

ORDINAL: Orderable items, such as levels in a corporate office, whose distinctions cannot be defined. Although they can be arranged, there is no mathematical unit of measurement.

Interval and Ratio datatypes are considered as numeric datatype.

INTERVAL: Ordered items with a mathematical unit of measurement and a quantifiable distance between them, but no absolute zero.

RATIO: Objects with order, a mathematical unit of measurement, and a quantifiable distance with absolute zero between them.

SL NO:	ATTRIBUTE	DESCRIPTION	NOIR DATA TYPE
1	car_ID	Unique id of each observation (Integer)	Nominal
2	symboling	Its assigned insurance risk rating, A value of +3 indicates that the auto is risky, -3 that it is probably pretty safe (Categorical)	Ordinal
3	CompanyName	This is a column derived from CarName(in the original dataset). The car name and company name are split. It denotes the name of the car company (Categorical)	Nominal
4	fueltype	Car fuel type i.e gas or diesel (Categorical)	Nominal
5	aspiration	Aspiration used in a car (Categorical)	Nominal
6	doornumber	Number of doors in a car (Categorical)	Nominal
7	carbody	The body of the car (Categorical)	Nominal
8	drivewheel	Type of drive wheel (Categorical)	Nominal
9	enginelocation	Location of a car engine (Categorical)	Nominal
10	wheelbase	Wheelbase of a car (Numeric)	Ratio
11	carlength	Length of the car (Numeric)	Ratio
12	carwidth	Width of the car (Numeric)	Ratio
13	carheight	Height of car (Numeric)	Ratio

14	curbweight	The weight of a car without occupants or baggage (Numeric)	Ratio
15	enginetype	Type of engine (Categorical)	Nominal
16	cylindernumber	Cylinder placed in the car (Categorical)	Nominal
17	enginesize	Size of the car engine (Numeric)	Ratio
18	fuelsystem	Fuel system of car (Categorical)	Nominal
19	boreratio	Boreratio of a car (Numeric)	Ratio
20	stroke	Stroke or volume inside the engine (Numeric)	Ratio
21	compressionratio	Compression ratio of a car (Numeric)	Ratio
22	horsepower	Horsepower (Numeric)	Ratio
23	peakrpm	Car peak rpm (Numeric)	Ratio
24	citympg	Mileage in the city (Numeric)	Ratio
25	highwaympg	Mileage on highway (Numeric)	Ratio
26	price	Price of the car (Numeric)(Target variable)	Ratio
27	curbweight_interval	curbweight attribute converted from ratio to interval	Interval

Table 1: Attributes information and NOIR analysis

1.2 RESEARCH QUESTIONS RELATED TO THE DATASET

Studying the data set can help us answer a lot of research questions, few of which are mentioned below,

1. Can we predict the price of a car based on its features, such as engine size, mileage, and type of transmission?[1]
2. The relation and variance between city mpg and highway mpg. How it affects the performance of the car?[2]
3. How does the car's body type (e.g., sedan, SUV, hatchback) affect its price and other features?[3]

1.3 THE POTENTIAL BENEFITS OF ANSWERING THE RESEARCH QUESTIONS

The automotive industry is an important and dynamic sector, and datasets like these can provide valuable insights into consumer behavior, product trends, and industry dynamics.

- It will help an individual interested in purchasing a car to make informed decisions based on the features and prices of different models.
- It will help the automotive industry to understand the market, the sales and the public demand to work on their design, production and work on their supply. It can help the companies to compare their car prices to those of their competitors, to identify areas where improvement can be done and pricing strategy can be adjusted to remain competitive in the market.
- It helps the insurance companies to study the public demand and industry supply and introduce their policies which will benefit the customers and themselves. Also, it will help the insurance companies to collaborate with the cars makers to introduce their policies to the customers.

Overall, studying the car price data set can provide valuable insights for both individuals and businesses in the car industry.

2. RELATED WORKS

Predicting price of cars has been studied extensively in various researches. A paper by Kanwal Noor and Sadaqat Jan presents a vehicle price prediction system by using the supervised machine learning technique. The machine learning prediction approach used in the study, multiple linear regression, provided 98% forecast precision. Multiple linear regression employs numerous independent variables but only one dependent variable, the actual and predicted values of which are compared to determine precision of findings. This paper provides a method in which price is a dependent variable that is anticipated and is generated from characteristics such as car model, manufacturer, city, version, color, mileage, alloy rims, and power steering[1].

Another paper by Qilin Li is the "US Auto Production and Price Prediction in the Context of Multiple Regression Analysis". This paper investigates US auto prices based on a multiple-factor regression model, which finds a direct correlation between semiconductor production and the auto production rate. It also follows along with the Consumer Price Index and unemployment rate, providing an approximate prediction of future auto car prices in the US. The results can be used to guide further exploration of stock price variation through multiple factors[2].

Machine learning (ML) methodologies were used in predicting vehicle prices and excellent bargains in the paper "Vehicle Price Classification and Prediction Using Machine Learning in the IoT Smart Manufacturing Age." With the rise of IoT for sustainability, vehicle value prediction has emerged as one of the most important study areas. This is due to the fact that it necessitates noticeable exertion and large field data. We used three ML algorithms to create a model that predicts the pricing of autos (neural network, decision tree, support vector machine, and linear regression). Yet, the strategies mentioned above have been combined to work as a group in a hybrid model. The data used was obtained from an information and computer science school that maintains several datasets. Different displays of several ML approaches were compared to determine which one is most suited for the accessible information index. Many difficulties and concerns with this design have also been highlighted. Furthermore, the model was tested, and 90% precision was obtained. This potential outcome could aid in the provision of exact vehicle deals in the coming Internet of Things (IoT) for the sustainability paradigm[3].

3. DATASET ANALYSIS

3.1 IMPORT DATASET

Before starting the study, we must import the dataset and required libraries. I've used Python, R and SQL for the entire analysis. Majorly used Python.

Import the python libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
```

Figure 1: Import Libraries

Reading the dataset

```
cars_data = pd.read_csv("car_price.csv")
```

Figure 2: Read dataset

Dataset information:

In this dataset we have 205 rows and 26 attributes (columns)

```
cars.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   car_ID            205 non-null    int64
 1   symboling         205 non-null    int64
 2   CarName           205 non-null    object
 3   fueltype          205 non-null    object
 4   aspiration        205 non-null    object
 5   doornumber        205 non-null    object
 6   carbody           205 non-null    object
 7   drivewheel        205 non-null    object
 8   enginelocation    205 non-null    object
 9   wheelbase         205 non-null    float64
 10  carlength         205 non-null    float64
 11  carwidth          205 non-null    float64
 12  carheight         205 non-null    float64
 13  curbweight        205 non-null    int64
 14  enginetype        205 non-null    object
 15  cylindernumber    205 non-null    object
 16  enginesize        205 non-null    int64
 17  fuelsystem        205 non-null    object
 18  boreratio         205 non-null    float64
 19  stroke            205 non-null    float64
 20  compressionratio  205 non-null    float64
 21  horsepower        205 non-null    int64
 22  peakrpm           205 non-null    int64
 23  citympg           205 non-null    int64
 24  highwaympg        205 non-null    int64
 25  price             205 non-null    float64
dtypes: float64(8), int64(8), object(10)
memory usage: 41.8+ KB
```

Figure 3: Data Info

Summary of the distribution of a numerical dataset. Output includes the count of non-null values, the mean, the standard deviation, the minimum value, the 25th percentile, the median (50th percentile), the 75th percentile, and the maximum value.

```
cars.describe()
```

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	boreratio	stroke	compressionratio	horsepowe
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000
mean	103.000000	0.834146	98.756585	174.049268	65.907805	53.724878	2555.565854	126.907317	3.329756	3.255415	10.142537	104.117073
std	59.322565	1.245307	6.021776	12.337289	2.145204	2.443522	520.680204	41.642693	0.270844	0.313597	3.972040	39.544167
min	1.000000	-2.000000	86.600000	141.100000	60.300000	47.800000	1488.000000	61.000000	2.540000	2.070000	7.000000	48.000000
25%	52.000000	0.000000	94.500000	166.300000	64.100000	52.000000	2145.000000	97.000000	3.150000	3.110000	8.600000	70.000000
50%	103.000000	1.000000	97.000000	173.200000	65.500000	54.100000	2414.000000	120.000000	3.310000	3.290000	9.000000	95.000000
75%	154.000000	2.000000	102.400000	183.100000	66.900000	55.500000	2935.000000	141.000000	3.580000	3.410000	9.400000	116.000000
max	205.000000	3.000000	120.900000	208.100000	72.300000	59.800000	4066.000000	326.000000	3.940000	4.170000	23.000000	288.000000

Figure 4: Summary of Numeric data

3.2 DATA PREPROCESSING AND DATA CLEANING

3.2.1 CHECKING FOR NULL, DUPLICATE AND DATA UNIQUE VALUES IN ALL COLUMNS

```
cars.isna()
```

	car_ID	symboling	CompanyName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	fuelsystem	boreratio	stroke	cc
0	False	False	False	False	False	False	False	False	False	False	...	False	False	False	
1	False	False	False	False	False	False	False	False	False	False	...	False	False	False	
2	False	False	False	False	False	False	False	False	False	False	...	False	False	False	
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False	
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
200	False	False	False	False	False	False	False	False	False	False	...	False	False	False	
201	False	False	False	False	False	False	False	False	False	False	...	False	False	False	
202	False	False	False	False	False	False	False	False	False	False	...	False	False	False	
203	False	False	False	False	False	False	False	False	False	False	...	False	False	False	
204	False	False	False	False	False	False	False	False	False	False	...	False	False	False	

205 rows × 27 columns

Figure 5: Checking null values

We can infer from above figure that the dataset is free of null values.

```
CompanyName = cars['CarName'].apply(lambda x : x.split(' ')[0])
cars.insert(3,"CompanyName",CompanyName)
cars.drop(['CarName'],axis=1,inplace=True)
cars.head()
```

	car_ID	symboling	CompanyName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	enginesize	fuelsystem	boreratio
0	1	3	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47
1	2	3	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47
2	3	1	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68
3	4	2	audi	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19
4	5	2	audi	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19

5 rows × 26 columns

Figure 6: Split car name and company name

It is observed in earlier figures that the car name and company name are written together, which is creating difficulty to segregate according to company or car. Hence splitting the Company name and car name for easier analysis.

Once the company and car names are split we check for the uniqueness of the company names. And observe that there are spelling errors (can be seen in figure 7). Hence the spellings are corrected and checked for the update.

The spelling error in the CompanyName column and the fix for invalid names:
maxda = mazda
nissan = Nissan
porsche = porcshce
toyota = toyouta
vokswagen = volkswagen = vw

```
1 cars.CompanyName.unique()
```

```
array(['alfa-romero', 'audi', 'bmw', 'chevrolet', 'dodge', 'honda',
       'isuzu', 'jaguar', 'maxda', 'mazda', 'buick', 'mercury',
       'mitsubishi', 'Nissan', 'nissan', 'peugeot', 'plymouth', 'porsche',
       'porcshce', 'renault', 'saab', 'subaru', 'toyota', 'toyouta',
       'vokswagen', 'volkswagen', 'vw', 'volvo'], dtype=object)
```

Correcting the spelling errors

```
1 cars['CompanyName'] = cars['CompanyName'].replace('maxda',"mazda")
2 cars['CompanyName'] = cars['CompanyName'].replace('nissan',"Nissan")
3 cars['CompanyName'] = cars['CompanyName'].replace('porcshce',"porsche")
4 cars['CompanyName'] = cars['CompanyName'].replace('toyouta',"toyota")
5 cars['CompanyName'] = cars['CompanyName'].replace('vokswagen',"volkswagen")
6 cars['CompanyName'] = cars['CompanyName'].replace('vw',"volkswagen")
```

```
1 cars.CompanyName.unique()
```

```
array(['alfa-romero', 'audi', 'bmw', 'chevrolet', 'dodge', 'honda',
       'isuzu', 'jaguar', 'mazda', 'buick', 'mercury', 'mitsubishi',
       'Nissan', 'peugeot', 'plymouth', 'porsche', 'renault', 'saab',
       'subaru', 'toyota', 'volkswagen', 'volvo'], dtype=object)
```

```
1 cars['CompanyName'].value_counts().plot(kind="bar")
2 plt.xlabel("Car name")
3 plt.ylabel("No. of occurrences of the car")
4 plt.title("Count of Car")
5 plt.show()
```

Figure 7: Unique values in the dataset

Below are the graphs of the company/car names before fixing the invalid names and then updating to the unique values.

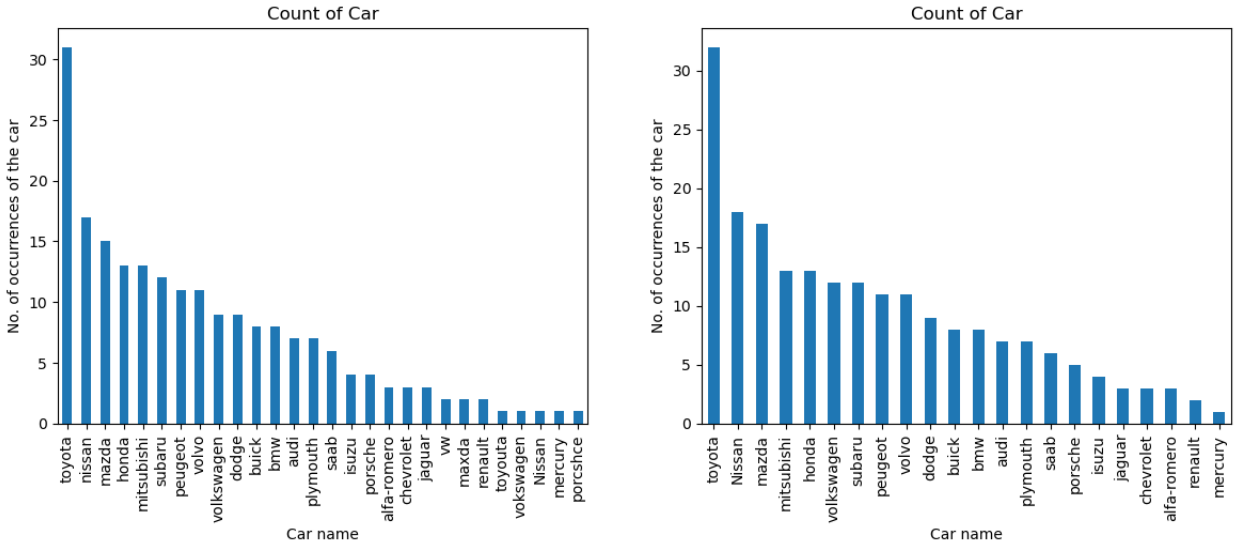


Figure 8: Invalid and Unique Company/Car names

Checking for duplicate values, we can observe that there are no duplicates in the dataset.

Checking for any duplicates

```
1 cars.loc[cars.duplicated()]
```

car_ID	symboling	CompanyName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	enginesize	fuelsystem	boreratio	s

0 rows × 26 columns

Figure 9: Checking for duplicates

It was observed that there are no interval data in the dataset. The curbweight weight attribute which is originally a ratio type is converted to interval datatype. I have taken the mean of the ratio data and each value in the interval is represented as a variance.

```
1 ratio_data = cars['curbweight'].values
2 log_data = np.log(ratio_data)
3 interval_data = log_data - np.mean(log_data)
4 cars['curbweight_interval'] = interval_data
5 print(cars['curbweight_interval'])
```

```
0      0.016838
1      0.016838
2      0.119330
3     -0.069602
4      0.119684
         ...
200    0.164013
201    0.196343
202    0.184134
203    0.249979
204    0.200598
Name: curbweight_interval, Length: 205, dtype: float64
```

Figure 10: Ratio to Interval conversion

3.2.2 DATA VISUALIZATIONS

Data visualization is the process of representing complex data in a graphical or pictorial format, which allows us to quickly and easily understand patterns, relationships, and trends that may not be apparent from raw data alone.

Below are the visualizations of the car features with itself and against price, which is the target variable.

```
plt.figure(figsize=(20,8))
plt.title('Distribution Plot for car price')
plt.xlabel("Car price")
plt.ylabel("Distribution in percentage")
sns.distplot(cars.price)
plt.show()
```

```
C:\Users\KetakiS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

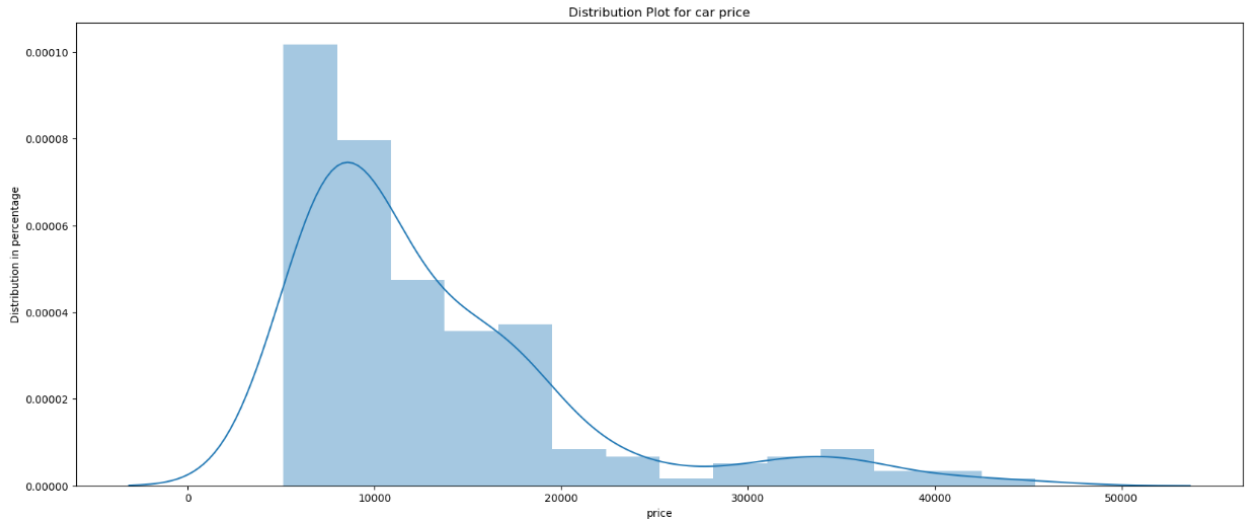


Figure 11: Distribution plot for price

```
plt.figure(figsize=(20,8))
plt.title('Range for car price')
sns.boxplot(x=cars.price)
plt.show()
```

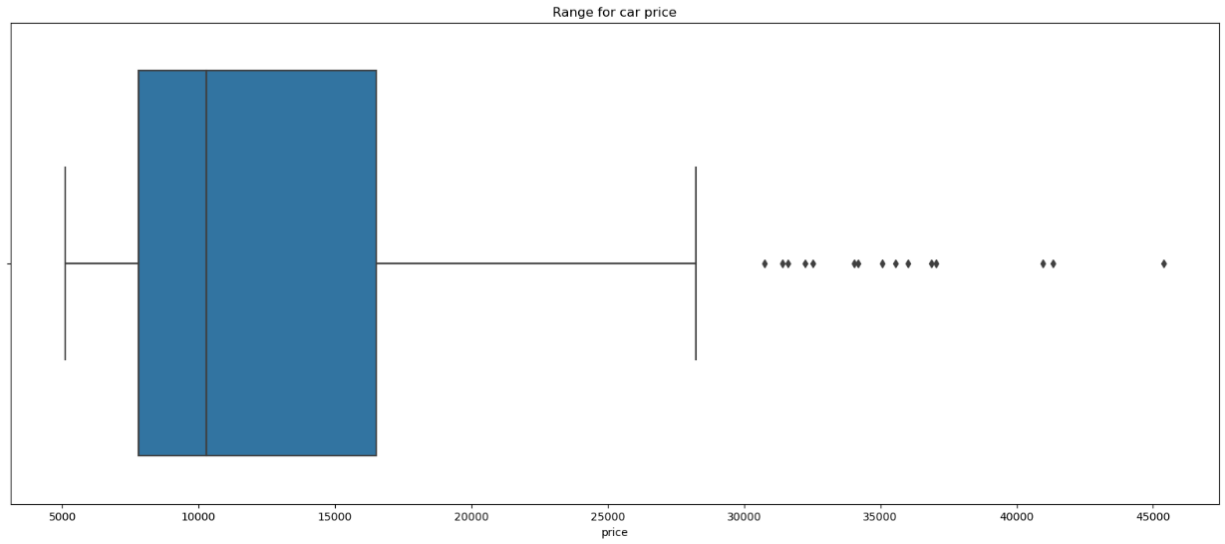


Figure 12: Range of price

```
1  cars.price.describe()
```

```
count      205.000000
mean     13276.710571
std       7988.852332
min       5118.000000
25%       7788.000000
50%      10295.000000
75%      16503.000000
max      45400.000000
Name: price, dtype: float64
```

Figure 13: Summary statistics of price

The plot seemed to be right-skewed, meaning that the most prices in the dataset are low(Below 15,000). There is a significant difference between the mean and the median of the price distribution. The data points are far spread out from the mean, which indicates a high variance in the car prices(75% of the prices are below 16,500, whereas the remaining 25% are between 16,500 and 45,400).

```
plt.figure(figsize=(25, 6))

plt.subplot(1,3,1)
plt1 = cars.CompanyName.value_counts().plot(kind="bar")
plt.title('Companies Histogram')
plt1.set(xlabel = 'Car company', ylabel='Frequency of company')

plt.subplot(1,3,2)
plt1 = cars.fueltype.value_counts().plot(kind="bar")
plt.title('Fuel Type Histogram')
plt1.set(xlabel = 'Fuel Type', ylabel='Frequency of fuel type')

plt.subplot(1,3,3)
plt1 = cars.carbody.value_counts().plot(kind="bar")
plt.title('Car Type Histogram')
plt1.set(xlabel = 'Car Type', ylabel='Frequency of Car type')

plt.show()
```

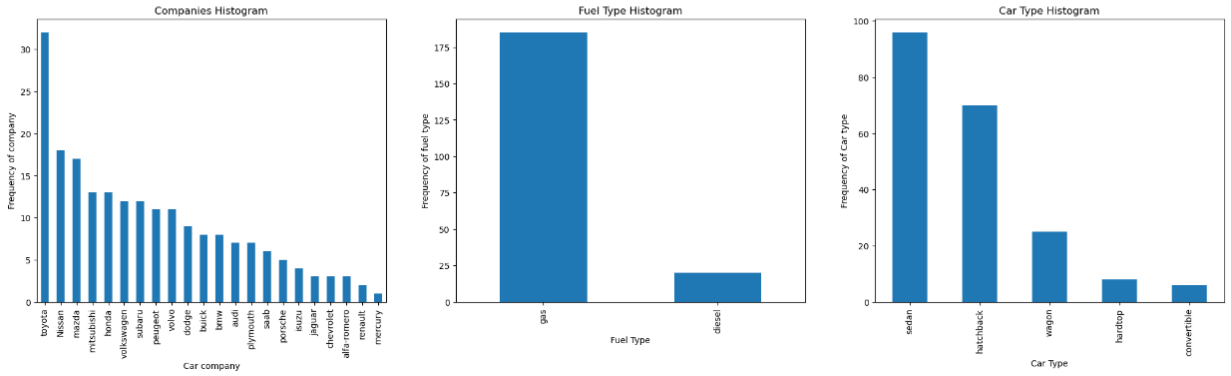


Figure 14: Histogram for Company, Fuel type and Car type

Here, I am visualizing the categorical data. From the graphs we can infer that Toyota seemed to be favored car company. The number of gas fueled cars are more than diesel. Sedan is the top car type preferred.

Checking for different features of the cars and its relation to the price. This will answer the research questions,

1. Can we predict the price of a car based on its features, such as engine size, mileage, and type of transmission?[1]
2. The relation and variance between city mpg and highway mpg. How it affects the performance of the car?[2]
3. How does the car's body type (e.g., sedan, SUV, hatchback) affect its price and other features?[3]

```
1 plt.figure(figsize=(20,8))
2
3 plt.title('Engine Type Histogram')
4 sns.countplot(cars.enginetype, palette=("Blues_d"))
```

```
C:\Users\KetakiS\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:title={'center':'Engine Type Histogram'}, xlabel='enginetype', ylabel='count'>
```

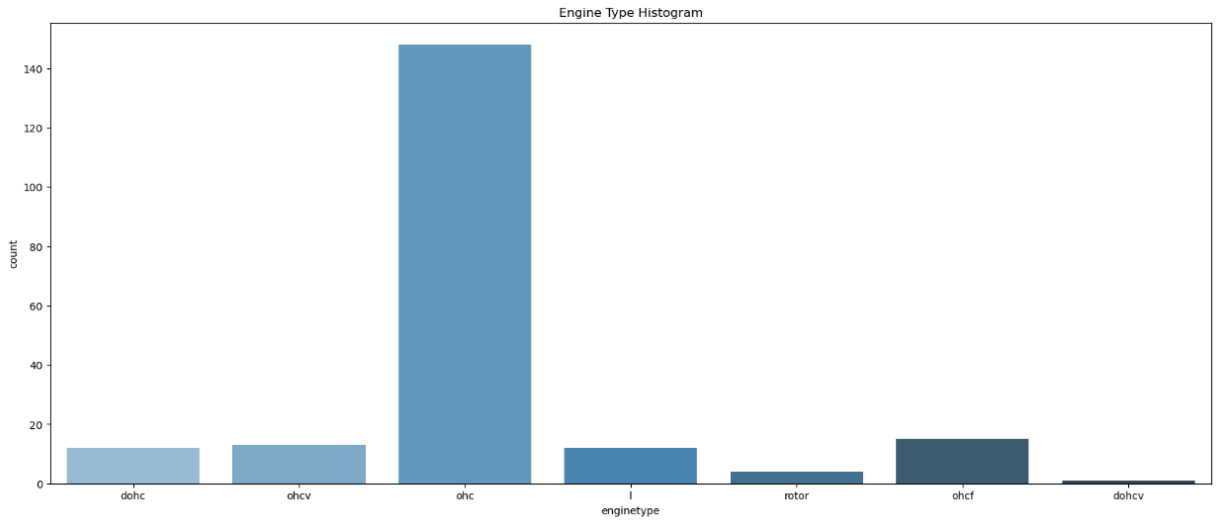


Figure 15: Histogram for Enginetype

```
df = cars.groupby(['enginetype'])['price'].mean().sort_values(ascending = False)
ax = df.plot(kind="bar", figsize=(8,6))
ax.set_title('Engine Type vs Average Price')
ax.set_xlabel('Engine Type')
ax.set_ylabel('Average Price')
plt.show()
```

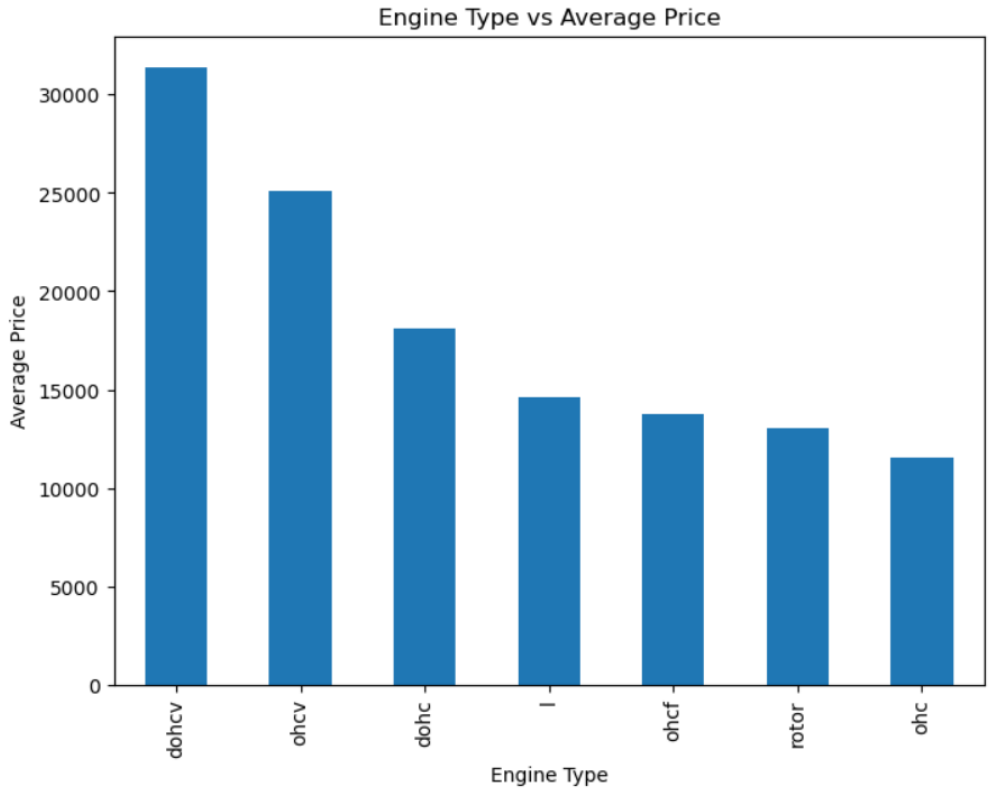


Figure 16: Relation between Engine type and price

From the graphs we can infer that ohc Engine type seems to be most favored type. dohcv has the highest price range, while ohc has lowest, dohc, I and ohcf have the low-price range.

```
df = cars.groupby(['CompanyName'])['price'].mean().sort_values(ascending = False)
ax = df.plot(kind="bar", figsize=(8,6))
ax.set_title('Company Name vs Average Price')
ax.set_xlabel('Company Name')
ax.set_ylabel('Average Price')
plt.show()
```

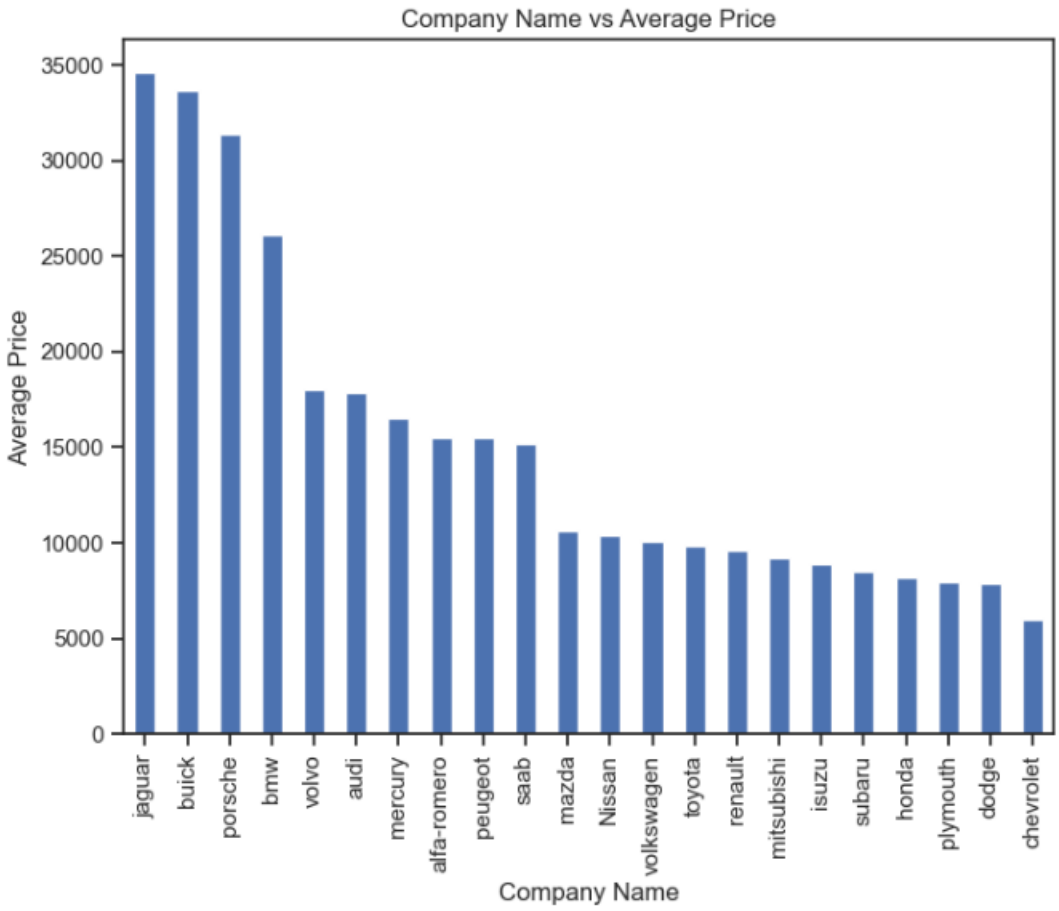


Figure 17: Relation between Company name and price

From the graph we can infer that, Jaguar and Buick seem to have highest average price and Chevrolet has lowest price range.

```
fig = plt.figure(figsize=(15,5))


plt.subplot(1,2,1)
plt.title('Fuel Type Histogram')
sns.countplot(cars.fueltype)


plt.subplot(1,2,2)
df = cars.groupby(['fueltype'])['price'].mean().sort_values(ascending = False)
ax = df.plot(kind="bar", figsize=(8,6))
ax.set_title('Fuel Type vs Average Price')
ax.set_xlabel('Fuel Type')
ax.set_ylabel('Average Price')

fig.tight_layout(pad = 5.0)
plt.show()
```

```
C:\Users\KetakiS\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follo
arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other argum
eyword will result in an error or misinterpretation.
  warnings.warn(
```

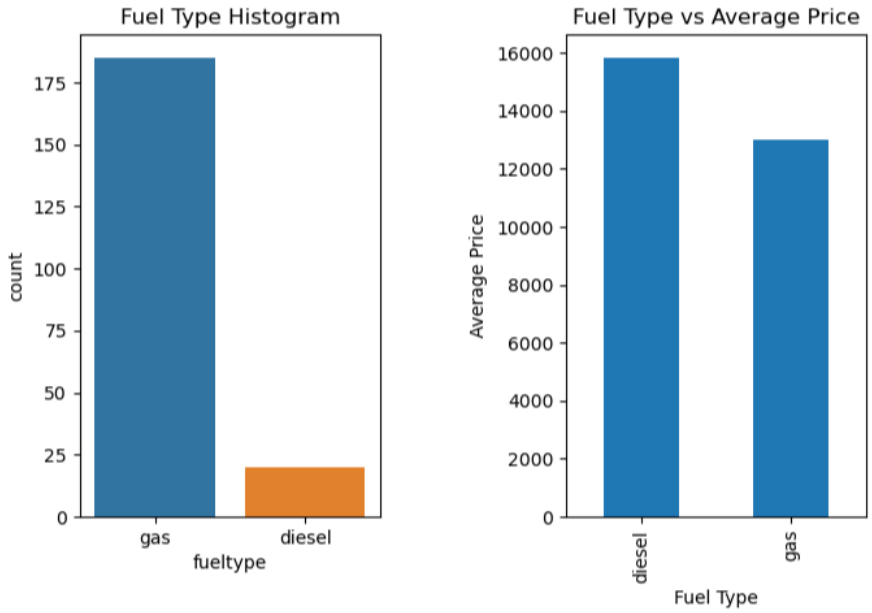


Figure 18: Relation between Fuel type and price

From the graph above we can infer that, Diesel has higher average price than gas and cars using gas fuel are more preferred.

```
df = cars.groupby(['carbody'])['price'].mean().sort_values(ascending = False)
ax = df.plot(kind="bar", figsize=(8,6))
ax.set_title('Car Type vs Average Price')
ax.set_xlabel('Car Type')
ax.set_ylabel('Average Price')
plt.show()
```

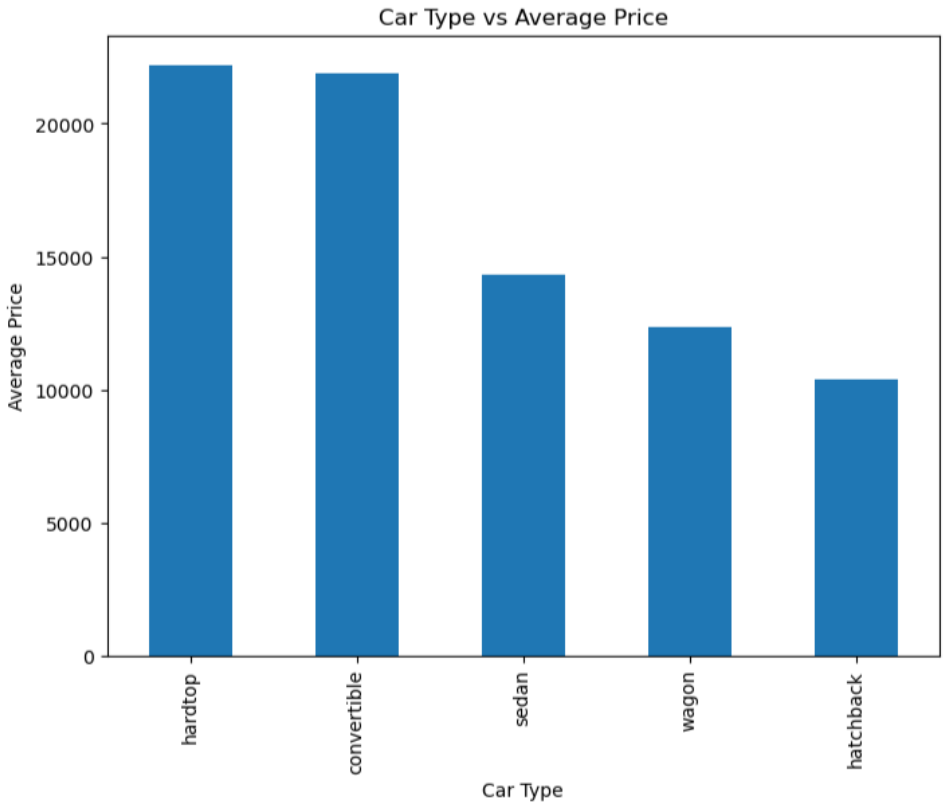


Figure 19: Relation between Car type and price

From the above graph we can observe that, Hardtop and Convertible have higher price range and Hatchback is the lowest price.

```
fig = plt.figure(figsize=(15,5))

plt.subplot(1,2,1)
plt.title('Aspiration Histogram')
sns.countplot(cars.aspiration)
plt.xticks(rotation=45)

plt.subplot(1,2,2)
df = cars.groupby(['aspiration'])['price'].mean().sort_values(ascending = False)
ax = df.plot(kind="bar", figsize=(8,6))
ax.set_title('Aspiration vs Average Price')
ax.set_xlabel('Aspiration')
ax.set_ylabel('Average Price')
plt.xticks(rotation=45)

fig.tight_layout(pad = 5.0)
plt.show()

plt.title('Aspiration vs Price')
sns.boxplot(x=cars.aspiration, y=cars.price)
plt.show()
```

```
C:\Users\KetakiS\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass
arg: x. From version 0.12, the only valid positional argument will be `data`, and passing ot
eyword will result in an error or misinterpretation.
  warnings.warn(
```

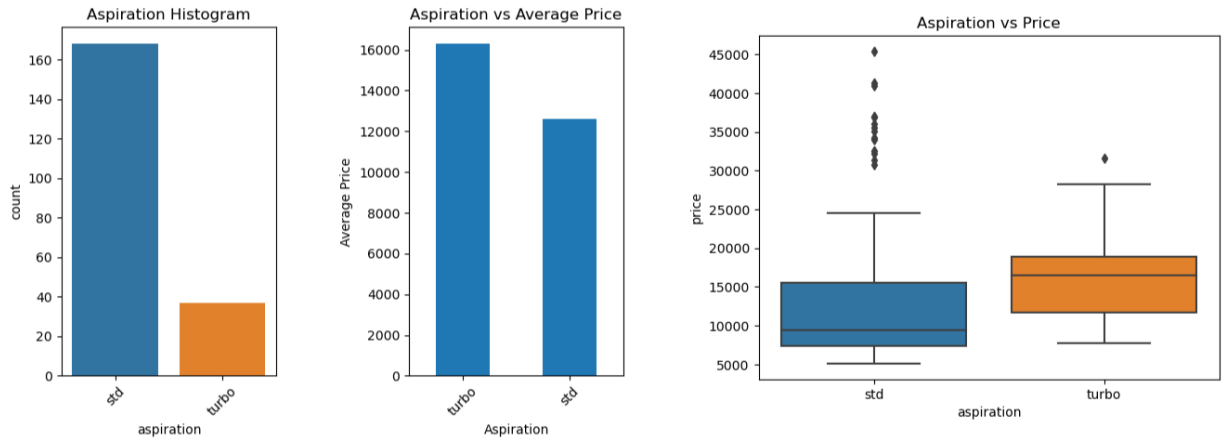


Figure 20: Relation between Aspiration and price

We can see that the aspiration with turbo have higher price range than the std (though it has some high values outside the whiskers).

```
fig = plt.figure(figsize=(15,5))

plt.subplot(1,2,1)
plt.title('Door Number Histogram')
sns.countplot(cars.doornumber)
plt.xticks(rotation=45)

plt.subplot(1,2,2)
df = cars.groupby(['doornumber'])['price'].mean().sort_values(ascending = False)
ax = df.plot(kind="bar", figsize=(8,6))
ax.set_title('Door Number vs Average Price')
ax.set_xlabel('Door Number')
ax.set_ylabel('Average Price')
plt.xticks(rotation=45)

fig.tight_layout(pad = 5.0)
plt.show()
```

```
C:\Users\KetakiS\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass t
arg: x. From version 0.12, the only valid positional argument will be `data`, and passing oth
eyword will result in an error or misinterpretation.
  warnings.warn(
```

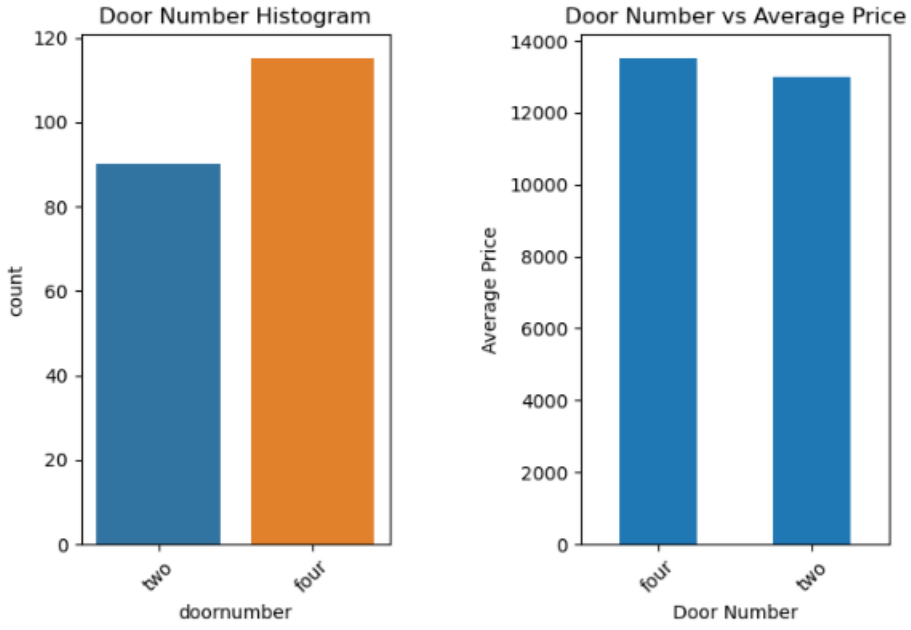


Figure 21: Relation between door number and price

From the above graph we can see that doornumber variable is not affecting the price much. There is no significant difference between the categories in it. However, we can see that the cars with two doors have lesser price. Cars with four doors is preferred than with two doors though the price for two doors is less.

```
fig = plt.figure(figsize=(15,5))

plt.subplot(1,2,1)
plt.title('Engine Location Histogram')
sns.countplot(cars.enginelocation)
plt.xticks(rotation=45)

plt.subplot(1,2,2)
df = cars.groupby(['enginelocation'])['price'].mean().sort_values(ascending = False)
ax = df.plot(kind="bar", figsize=(8,6))
ax.set_title('Engine Location vs Average Price')
ax.set_xlabel('Engine Location')
ax.set_ylabel('Average Price')
plt.xticks(rotation=45)

fig.tight_layout(pad = 5.0)
plt.show()
```

```
C:\Users\KetakiS\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follo
arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other argu
eyword will result in an error or misinterpretation.
  warnings.warn(
```

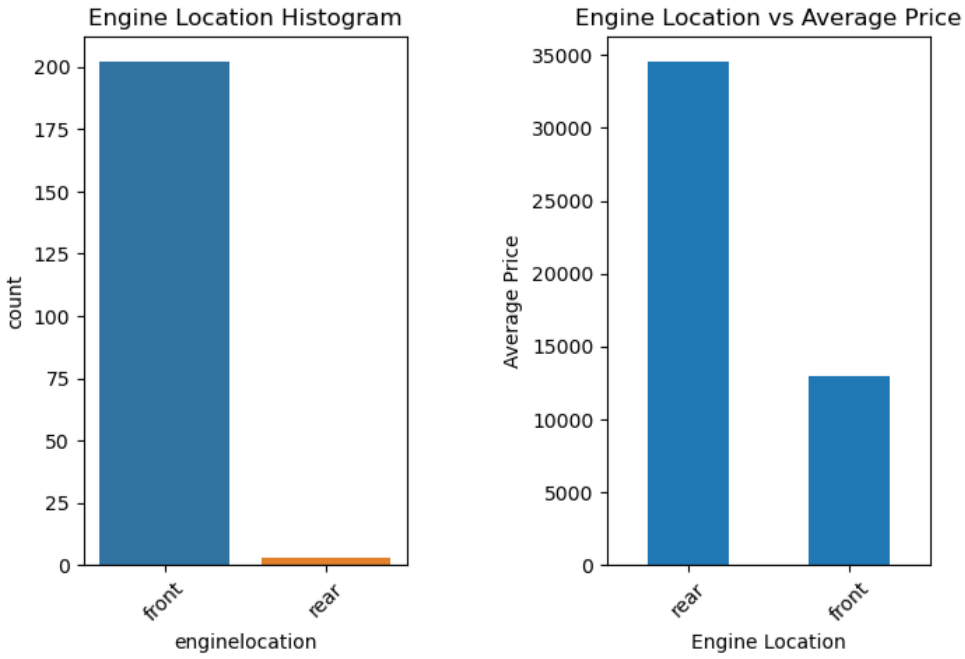


Figure 22: Relation between Engine location and price

```
fig = plt.figure(figsize=(15,5))

plt.subplot(1,2,1)
plt.title('Wheel Drive Histogram')
sns.countplot(cars.drivewheel)
plt.xticks(rotation=45)

plt.subplot(1,2,2)
df = cars.groupby(['drivewheel'])['price'].mean().sort_values(ascending = False)
ax = df.plot(kind="bar", figsize=(8,6))
ax.set_title('Wheel Drive vs Average Price')
ax.set_xlabel('Wheel Drive')
ax.set_ylabel('Average Price')
plt.xticks(rotation=45)


fig.tight_layout(pad = 5.0)
plt.show()
```

```
C:\Users\KetakiS\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follo
arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other argu
eyword will result in an error or misinterpretation.
  warnings.warn(
```

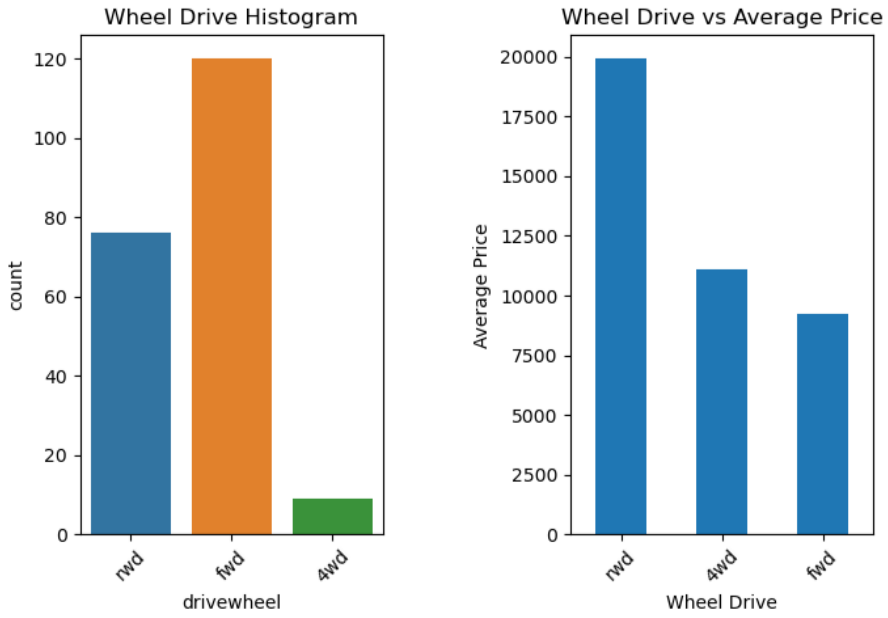


Figure 23: Relation between wheel drive and price

We can observe a very significant difference in drivewheel category. Most high ranged cars prefer rear wheel and hence economic and preferred drive wheel is the front wheel drive.

```
fig = plt.figure(figsize=(15,5))

plt.subplot(1,2,1)
plt.title('Cylinder Number Histogram')
sns.countplot(cars.cylindernumber)
plt.xticks(rotation=45)

plt.subplot(1,2,2)
df = cars.groupby(['cylindernumber'])['price'].mean().sort_values(ascending = False)
ax = df.plot(kind="bar", figsize=(8,6))
ax.set_title('Cylinder Number vs Average Price')
ax.set_xlabel('Cylinder Number')
ax.set_ylabel('Average Price')
plt.xticks(rotation=45)

fig.tight_layout(pad = 5.0)
plt.show()
```

```
C:\Users\KetakiS\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass th
arg: x. From version 0.12, the only valid positional argument will be `data`, and passing othe
eyword will result in an error or misinterpretation.
  warnings.warn(
```

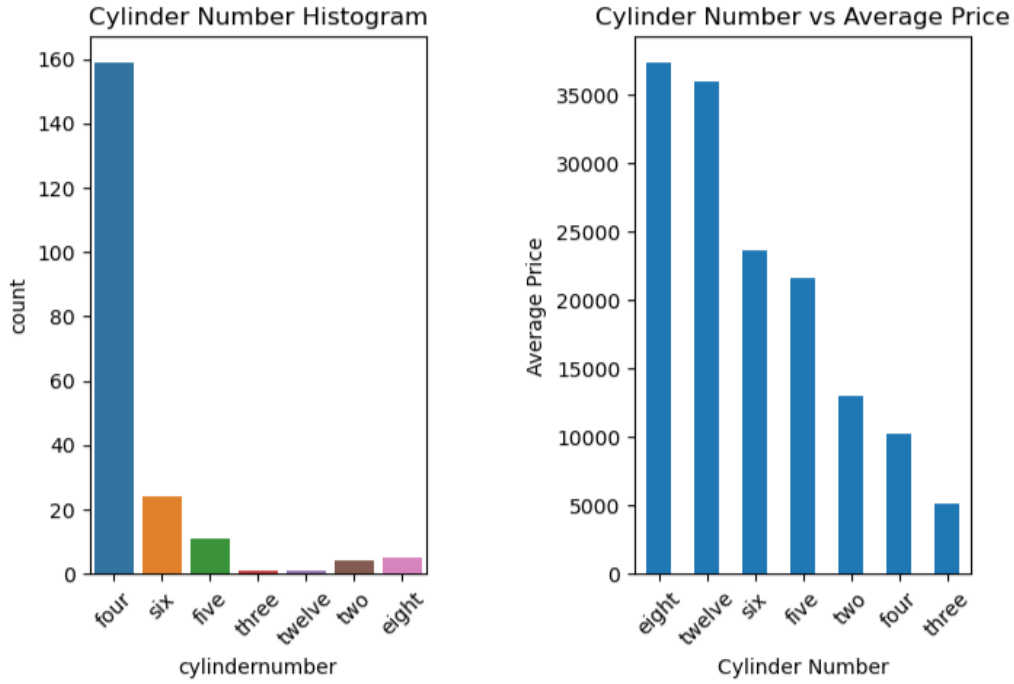


Figure 24: Relation between cylinder number and price

From the graph we understand that most common number of cylinders are four, five and six. And four-cylinder car is the most preferred one. Eight cylinders cars have the highest price range and three- and four-cylinders cars have low prices.

```python
def scatter(x, fig):
    ax = fig.add_subplot(6, 2, fig)
    ax.scatter(cars[x], cars['price'])
    ax.set_title(x + ' vs Price')
    ax.set_xlabel(x)
    ax.set_ylabel('Price')

fig, ax = plt.subplots(5, figsize=(10, 15))

ax[0].set_title("Car Length vs price")
ax[0].scatter(x = cars['carlength'], y = cars['price'])
ax[0].set_xlabel("Car Length")
ax[0].set_ylabel("Price")

ax[1].scatter(x = cars['carwidth'], y = cars['price'])
ax[1].set_title('Car Width vs price')
ax[1].set_xlabel("Car Width")
ax[1].set_ylabel("House Price")

ax[2].scatter(x = cars['carheight'], y = cars['price'])
ax[2].set_title("Car Height vs Price")
ax[2].set_xlabel("Car Height")
ax[2].set_ylabel("Price")

ax[3].scatter(x = cars['curbweight'], y = cars['price'])
ax[3].set_title("Curb Weight vs Price")
ax[3].set_xlabel("Curb Weight")
ax[3].set_ylabel("Price")

ax[4].scatter(x = cars['curbweight_interval'], y = cars['price'])
ax[4].set_title("Curb Weight (Interval data type) vs Price")
ax[4].set_xlabel("Curb Weight")
ax[4].set_ylabel("Price")

fig.tight_layout(pad = 6.0)
plt.show()
```

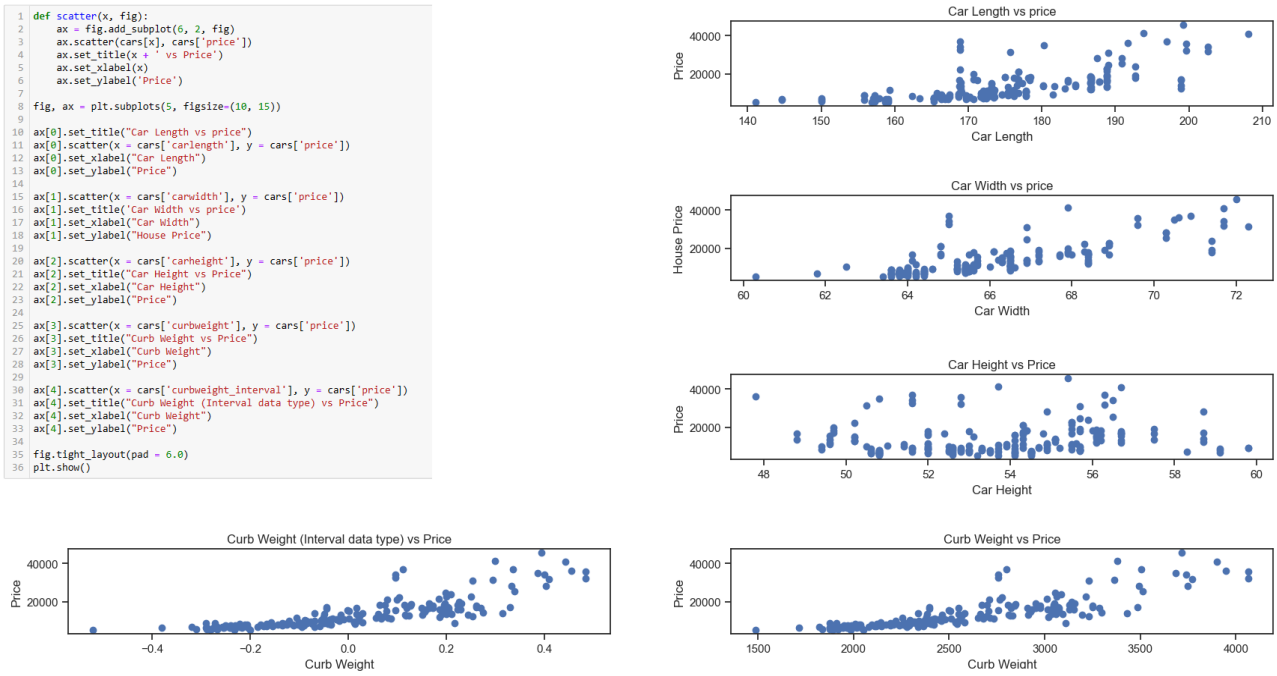


Figure 25: Relation between numeric data and price

From the graph we can observe that carwidth, carlength and curbweight show a positive correlation with price. However, carheight doesn't show any significant relation with price.
Curb weight interval is interval data type derived from curb weight which is ratio data type originally. We can observe that there is no difference in the graphs of ratio data type and interval data type.

```python
sns.scatterplot(x=cars.citympg, y=cars.price, hue=cars.highwaympg).set(title='Relation between mileage and price')
```

```
[Text(0.5, 1.0, 'Relation between mileage and price')]
```

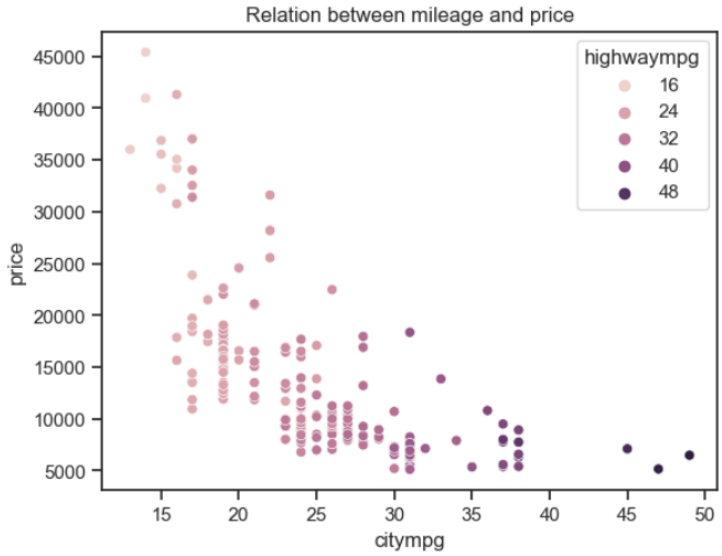


Figure 26: Relation between mileage and price

Observations from the above graph are, as the price goes up the mileage is lesser in the city as well as on the highways when compared to lesser priced cars. High end cars give lesser mileage.
However, average mileage between 20 to 30 is given by cars on highways and in the city, with the price also being on the average side ranging between $5000 to $20000.

```
plt.figure(figsize=(25, 6))
def pp(x, y, z):
    sns.set(style='ticks')
    fig.tight_layout(pad = 5.0)
    sns.pairplot(data=cars, vars=[x, y, z], y_vars=['price'], height=4, aspect=1)
    plt.suptitle('Pairplot of ' + x + ', ' + y + ', ' + z + ' vs. Price')
    plt.show()

pp('enginesize', 'boreratio', 'stroke')
pp('compressionratio', 'horsepower', 'peakrpm')
pp('wheelbase', 'citympg', 'highwaympg')
```

```
<Figure size 2500x600 with 0 Axes>
```

Figure 27: Code for graph to show relation between ratio data and price

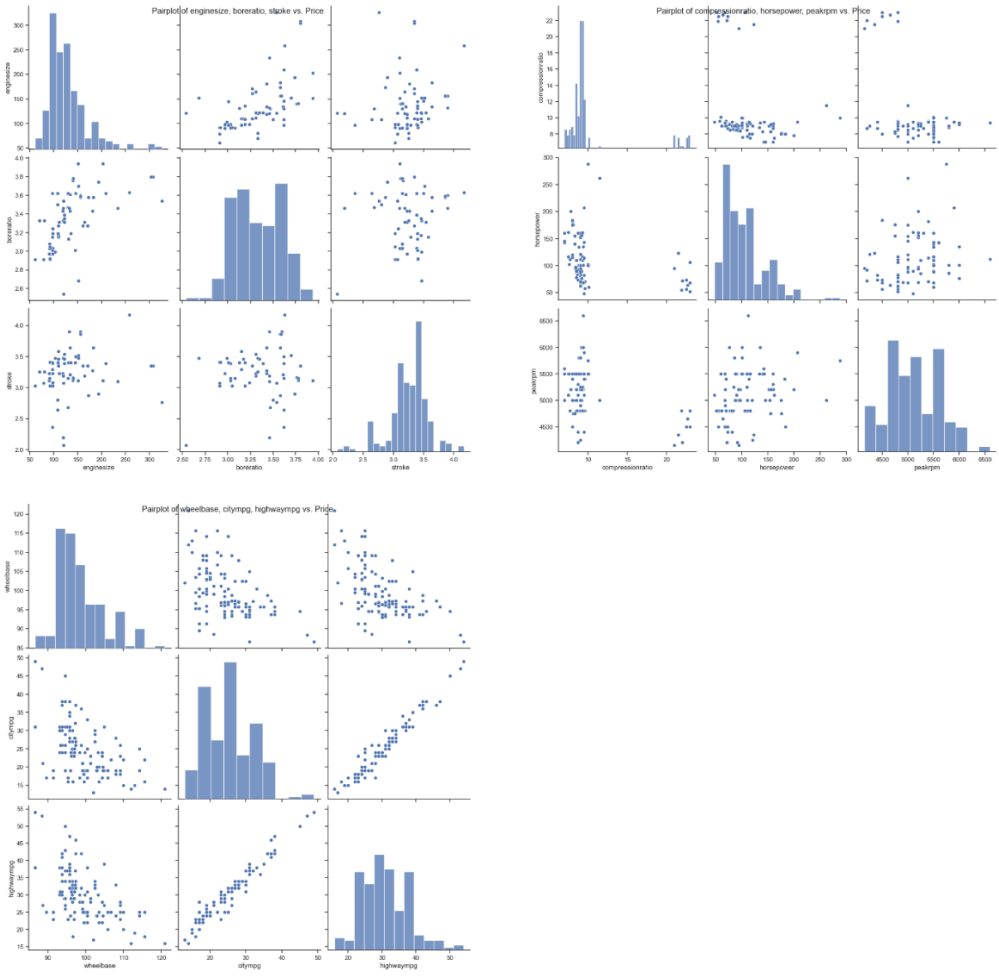


Figure 28: Relation between ratio data and price

From the visualizations we can infer that enginesize, boreratio, horsepower, wheelbase - have a significant positive correlation with price and citympg, highwaympg - have a significant negative correlation with price.

```
plt.figure(figsize=(25, 6))

df = pd.DataFrame(cars.groupby(['fuelsystem','drivewheel'])['price'].mean().unstack(fill_value=0))
df.plot.bar()
plt.title('Relation between the Fuel system, Average Price and drive wheel')
plt.show()
```

```
<Figure size 2500x600 with 0 Axes>
```

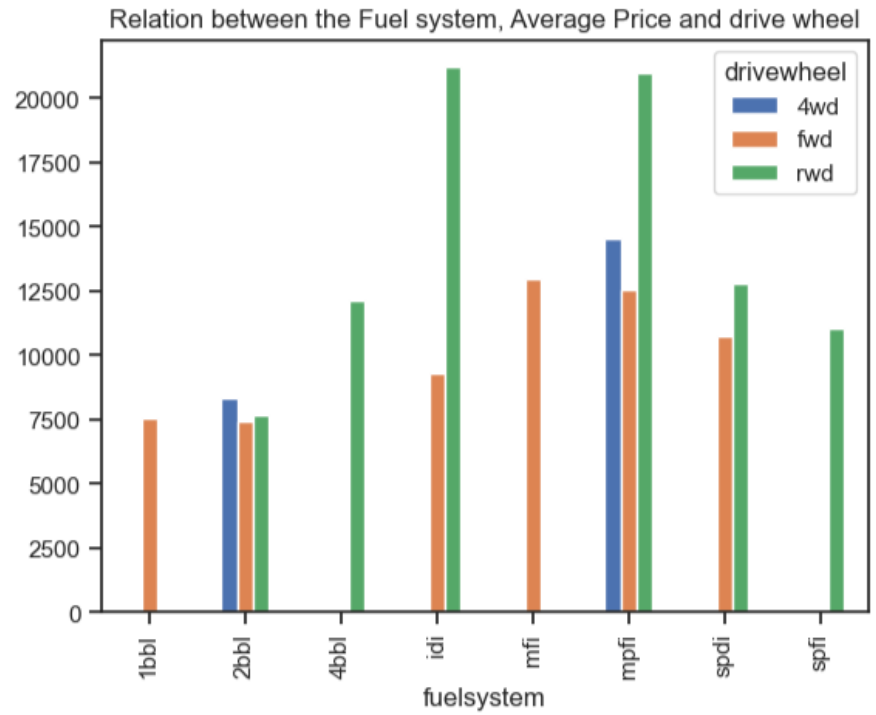


Figure 29: Relation between fuel system, drive wheel and price

High ranged cars prefer rwd drivewheel with idi or mpfi fuelsystem.

```
1 sns.pairplot(car_final)
2 plt.show()
```

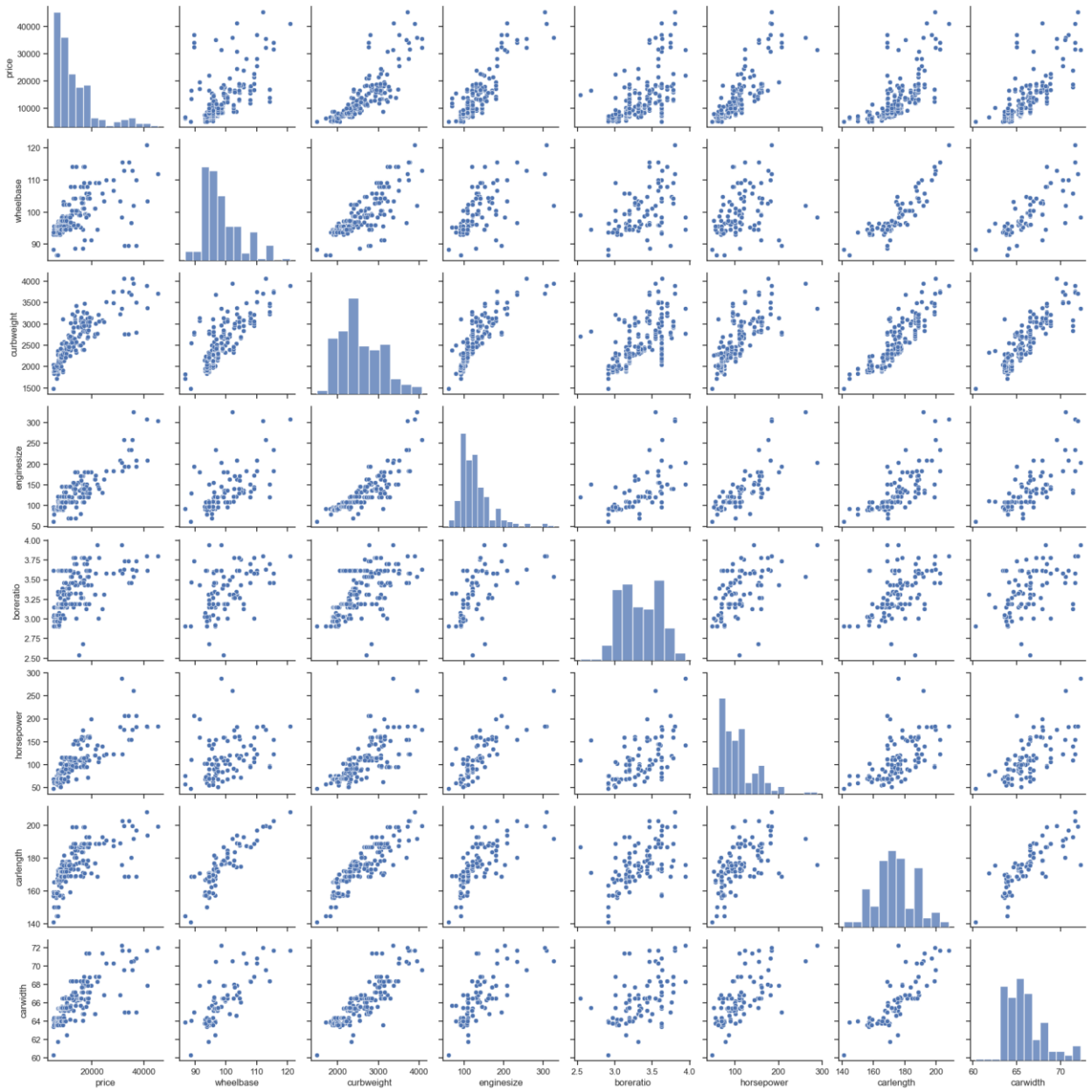


Figure 30: Pair plot for significant attributes

The data is split into numerical and categorical data to convert the categorical data to numeric data. This is required for data modeling , to fit the data into the model. After the conversion, the converted numeric data is concatenated with the numeric data.

```
car_final_quant = car_final[car_final.select_dtypes(include=[np.number]).columns.tolist()]
car_final_quant.head()
```

	price	wheelbase	curbweight	enginesize	boreratio	horsepower	carlength	carwidth
0	13495.0	88.6	2548	130	3.47	111	168.8	64.1
1	16500.0	88.6	2548	130	3.47	111	168.8	64.1
2	16500.0	94.5	2823	152	2.68	154	171.2	65.5
3	13950.0	99.8	2337	109	3.19	102	176.6	66.2
4	17450.0	99.4	2824	136	3.19	115	176.6	66.4

```
car_final_qual = car_final[car_final.select_dtypes(include=['object']).columns.tolist()]
car_final_qual.head()
```

	fueltype	aspiration	carbody	drivewheel	enginetype	cylindernumber
0	gas	std	convertible	rwd	dohc	four
1	gas	std	convertible	rwd	dohc	four
2	gas	std	hatchback	rwd	ohcv	six
3	gas	std	sedan	fwd	ohc	four
4	gas	std	sedan	4wd	ohc	five

Figure 31: Splitting numeric and categorical data

```
car_final_qual = pd.get_dummies(car_final_qual)
car_final_qual.head(5)
```

	fueltype_diesel	fueltype_gas	aspiration_std	aspiration_turbo	carbody_convertible	carbody_hardtop	carbody_hatchback	carbody_sedan	carbody_wagon	dr
0	0	1	1	0	1	0	0	0	0	
1	0	1	1	0	1	0	0	0	0	
2	0	1	1	0	0	0	1	0	0	
3	0	1	1	0	0	0	0	1	0	
4	0	1	1	0	0	0	0	1	0	

5 rows × 26 columns

Figure 32: Converting categorical variables to numeric variables

```
carsProssed = pd.concat([car_final_quant,car_final_qual], axis=1)
carsProssed.head(5)
```

	price	wheelbase	curbweight	enginesize	boreratio	horsepower	carlength	carwidth	fueltype_diesel	fueltype_gas	...	enginetype_ohcf	enginetype_ohcv
0	13495.0	88.6	2548	130	3.47	111	168.8	64.1	0	1	...	0	0
1	16500.0	88.6	2548	130	3.47	111	168.8	64.1	0	1	...	0	0
2	16500.0	94.5	2823	152	2.68	154	171.2	65.5	0	1	...	0	1
3	13950.0	99.8	2337	109	3.19	102	176.6	66.2	0	1	...	0	0
4	17450.0	99.4	2824	136	3.19	115	176.6	66.4	0	1	...	0	0

5 rows × 34 columns

Figure 33:Concatenate the actual numeric data and converted numeric data

```
plt.figure(figsize = (30, 25))
sns.heatmap(carsProssed.corr(), annot = True).set(title = "HeatMap")
plt.show()
```

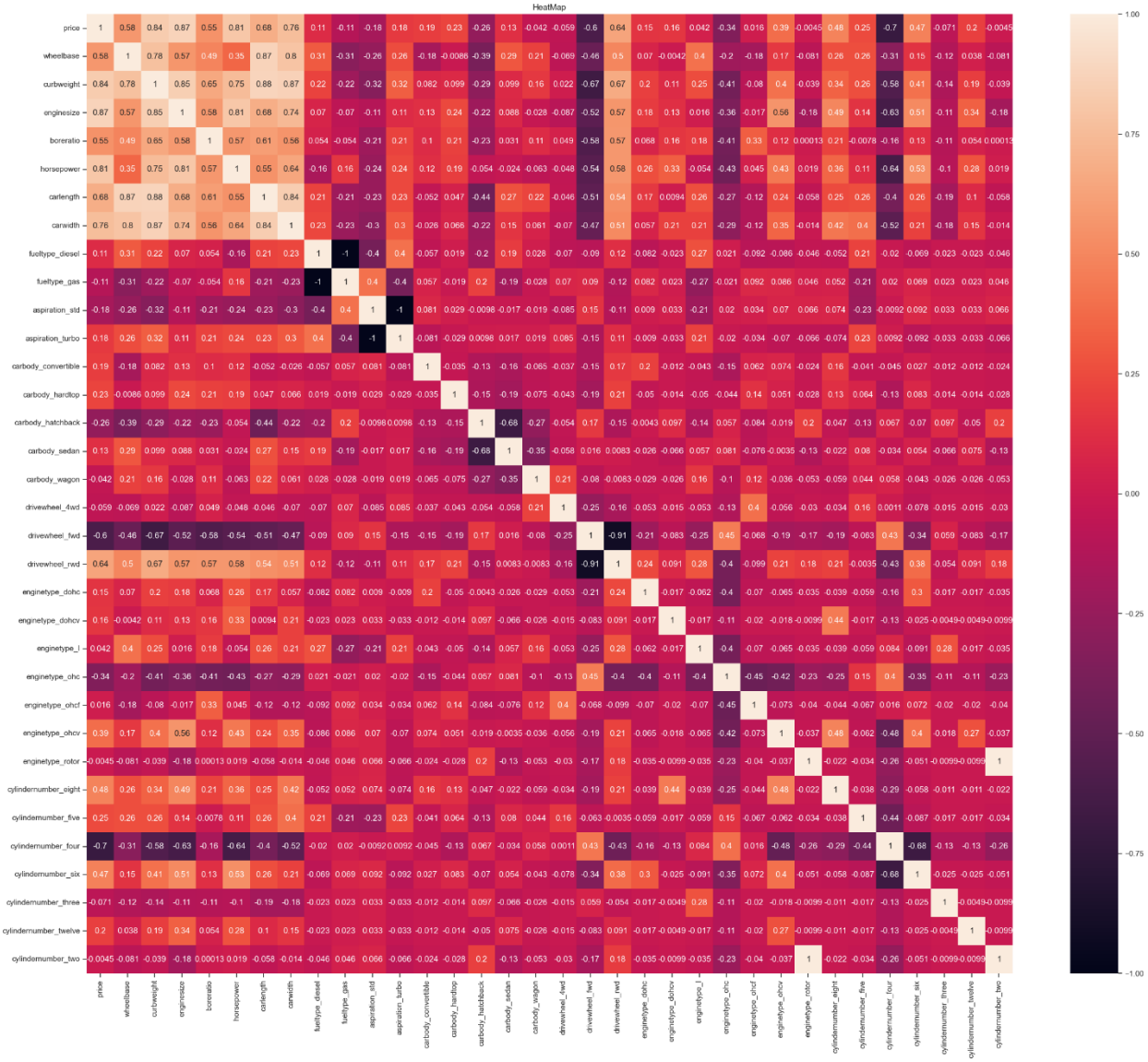


Figure 34: HeatMap

To see multiple correlations between numerical variables, a Heatmap is helpful. I have used it to comprehend the correlation of numerical variables with the target variable "Price".

Correlation of target variable “Price” with independent variables: Wheelbase, carlength, carwidth, curbweight, enginesize, and horsepower are all strongly (positively) correlated with price; take note of how all of these variables correspond to the size, weight, and engine power of the car. The relationship between price and "citympg" and "highwaympg" is negative (-0.70 approximately). This implies that vehicles with high mileage may be classified as "economy" vehicles and are therefore less expensive.

I have used linear regression model for data modeling. Here, I have imported the LinearRegression model from sklearn.linear_model library. The model is then trained with X_train and y_train. The same learned model is then used to test on unseen data. Then the model is evaluated based on r2 score which is imported from sklearn.metrics library. The model accuracy is found to be 87%.

```
from sklearn.feature_selection import RFE
from sklearn.linear_model import LinearRegression
import statsmodels.api as sm
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
lr = LinearRegression()
lr.fit(X_train,y_train)
```

```
LinearRegression()
```

```
y_pred = lr.predict(X_test)
```

```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred)
```

```
0.8733182974901593
```

Figure 35: Model Training

The Root Mean Square Error(RMSE) is calculated and is 2978.49. The testing accuracy of data is 87% and training accuracy is 92%.

Computing MSE, RMSE and R squared values

```
from sklearn.metrics import mean_squared_error

# Computing MSE and RMSE
lin_mse1 = mean_squared_error(y_test, y_pred)
lin_rmse1 = np.sqrt(lin_mse1)
print("Root mean squared value:",lin_rmse1)

# R squared value
r2_lin_test1=lr.score(X_test,y_test)
r2_lin_train1=lr.score(X_train,y_train)
print("Training Accuracy:",r2_lin_train1)
print("Testing Accuracy:", r2_lin_test1)
```

```
Root mean squared value: 2978.4904169032243
Training Accuracy: 0.9174040074084739
Testing Accuracy: 0.8733182974901593
```

Figure 36: Code to calculate Accuracy and RMSE

4. ANALYSIS USING R

R is a comprehensive statistical software package that enables for data processing and visualization. It is a statistical computing and graphics language and environment. Using R to analyze a dataset entails a combination of data exploration, cleaning, modification, analysis, and visualization.

I used R studio to explore, analyze and visualize the data. I imported the required libraries and read the dataset, performed data exploration and visualizations. Below are the code snippets and the outputs.

NOTE: The observations from the graphs in R are same as that of Python.

4.1 READING DATASET

There are various packages in R that can be used to read datasets, including readr, data.table, readxl, and read.csv(. The package chosen is determined by the specific demands and characteristics of the dataset being read. To read the dataset, I used the read.csv() function.

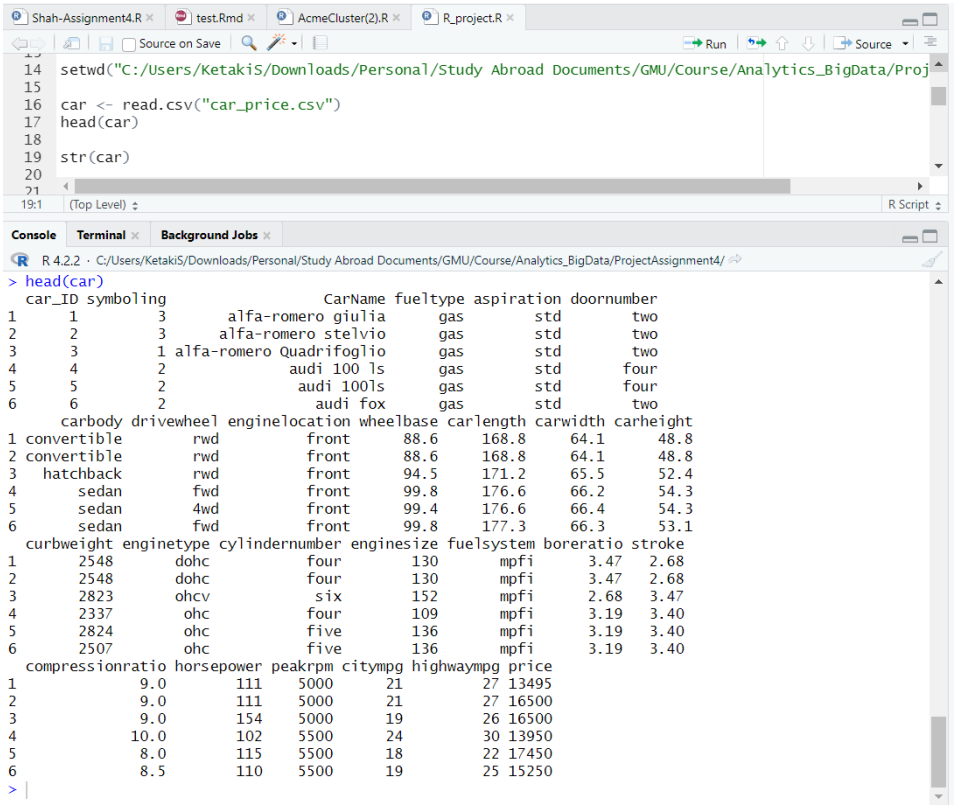


Figure 37: Read dataset in R

4.2 DATA UNDERSTANDING

Data understanding is a crucial step in any data analysis project. It allows you to figure out if the data is complete and gives basic information of the data. This helps to perform further analysis.

- str() gives the structure of the dataset, including the variable names, data types, and dimensions
- head() gives a quick view of the dataset
- summary() gives a summary of each variable, including the minimum, maximum, mean, and median

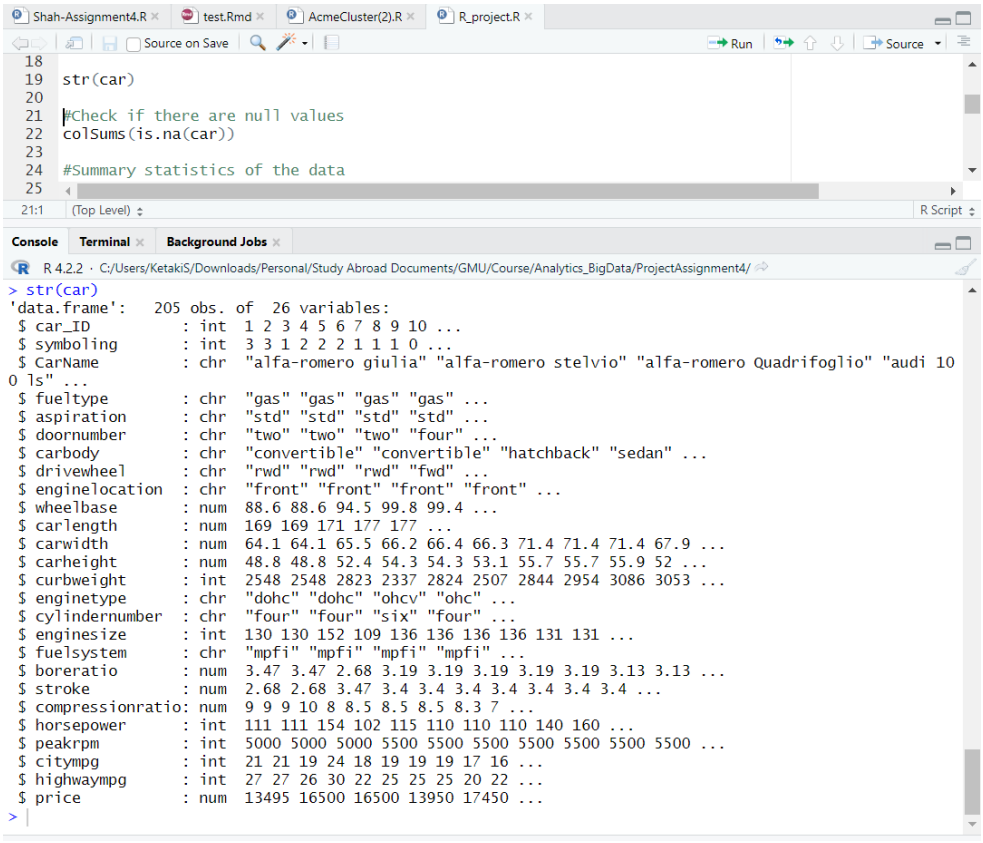


Figure 38: Check for str values

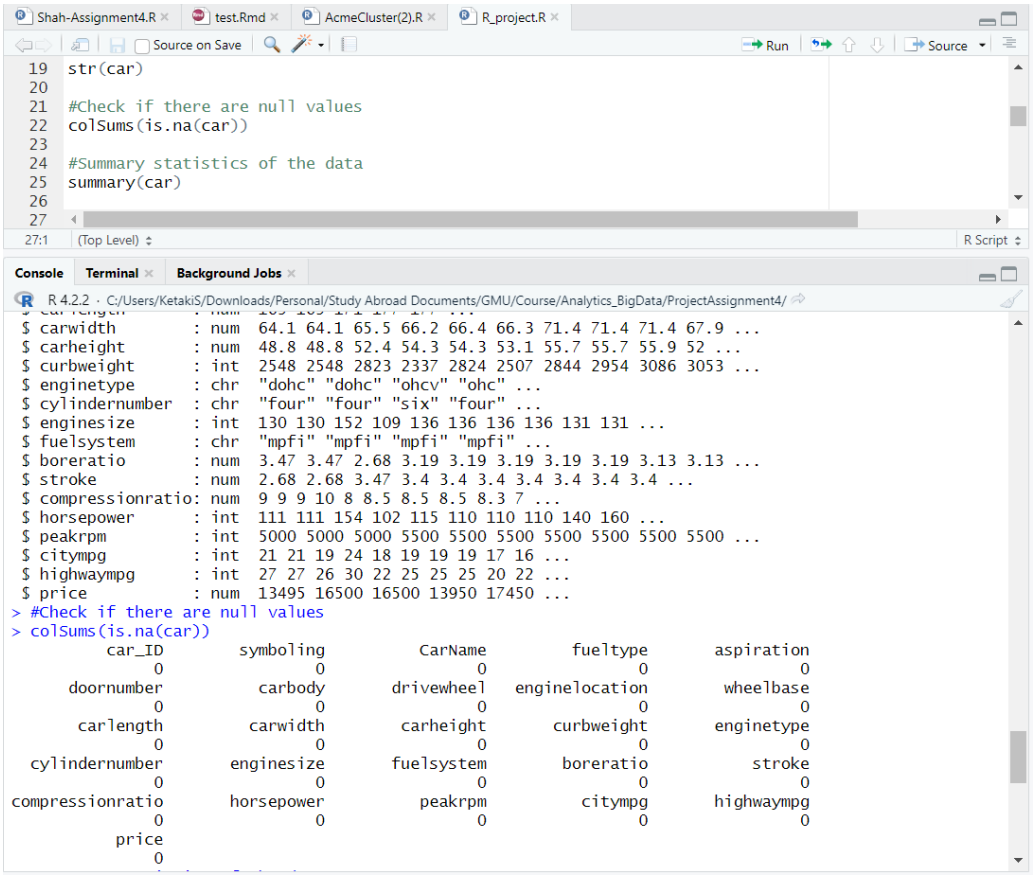


Figure 39: Check for null values

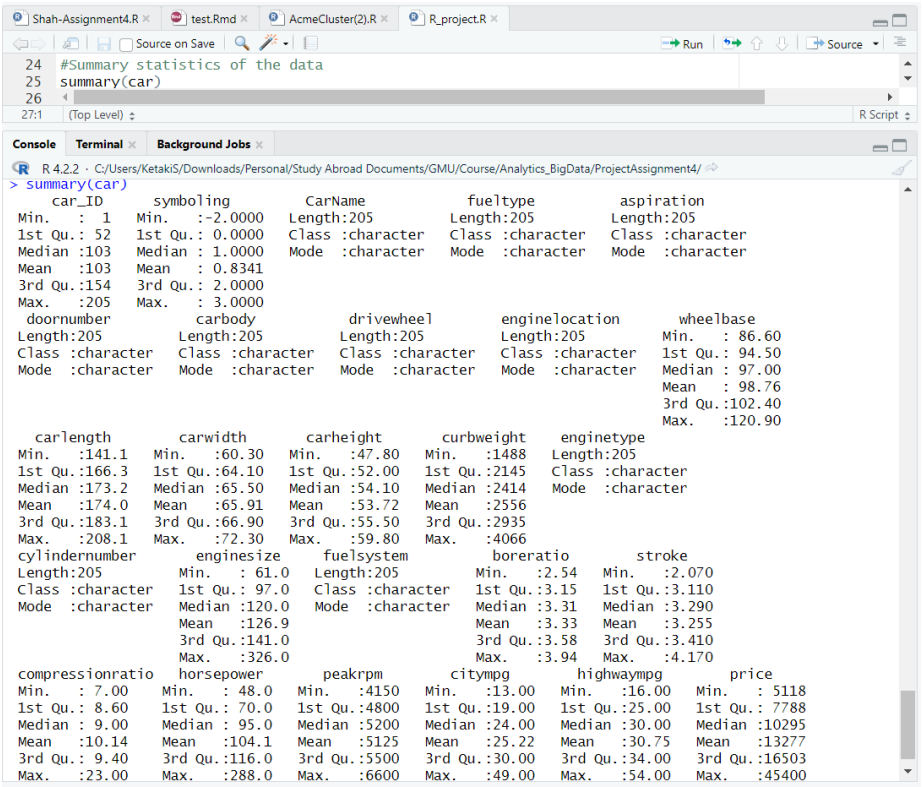


Figure 40: Summary Statistics

4.3 DATA ANALYSIS

Data analysis in R involves a variety of techniques and packages for data visualization. Below are a few examples with code snippet and the graphs.

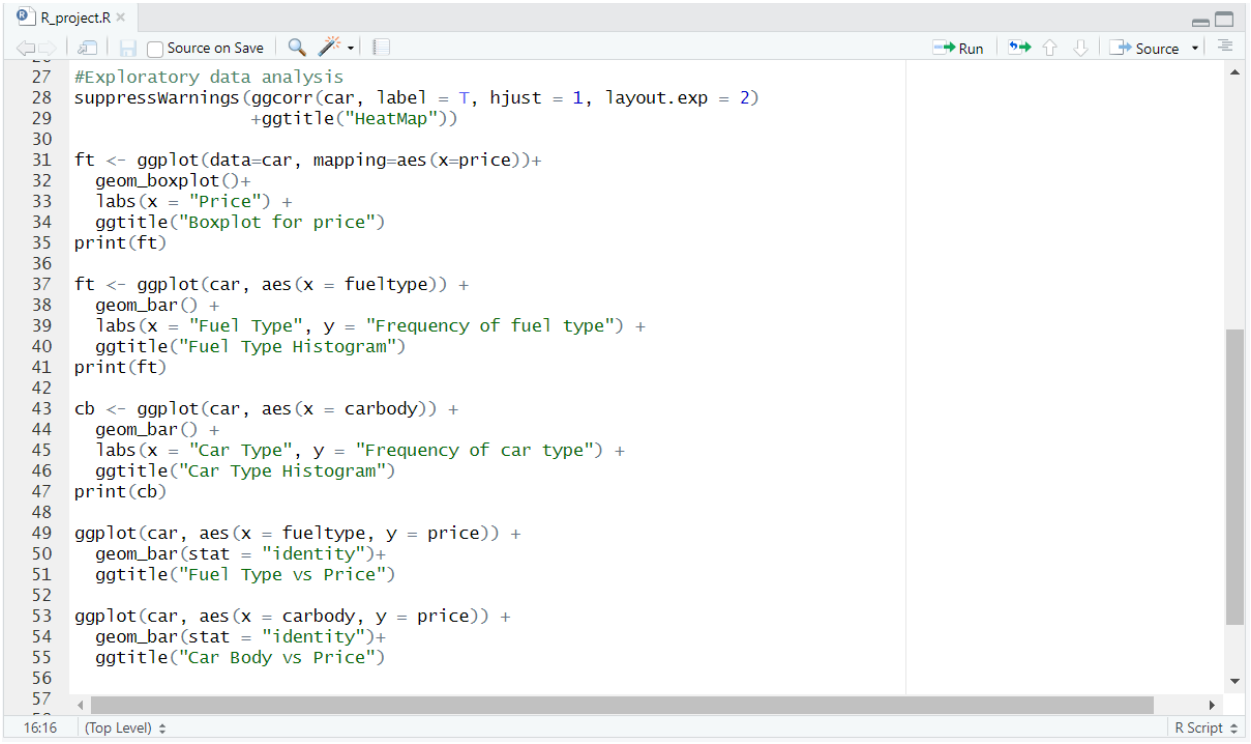


Figure 41: Data Exploration

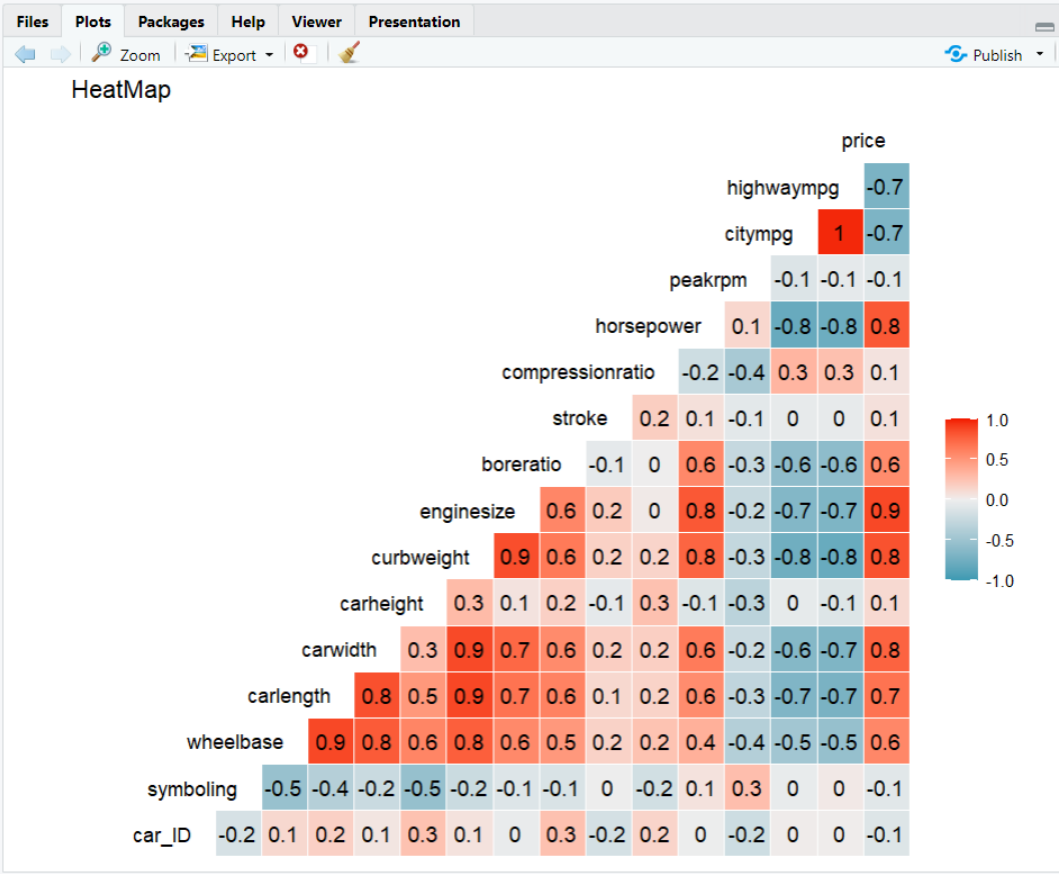


Figure 42: HeatMap in R

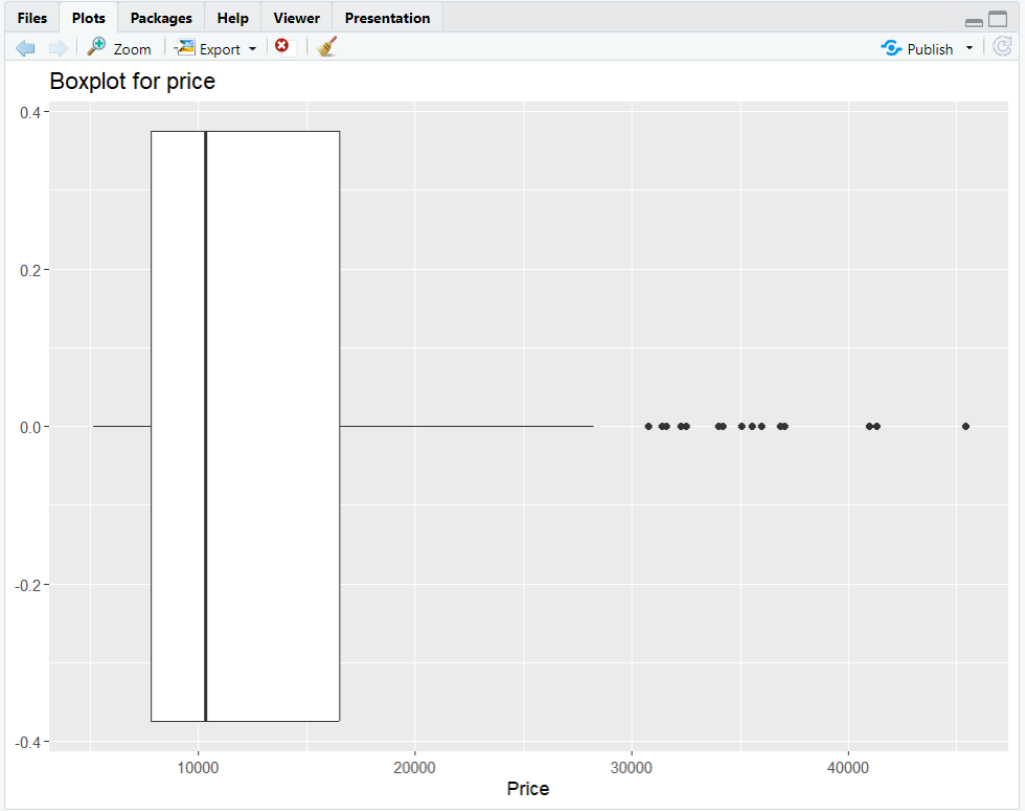


Figure 43: Boxplot for price in R

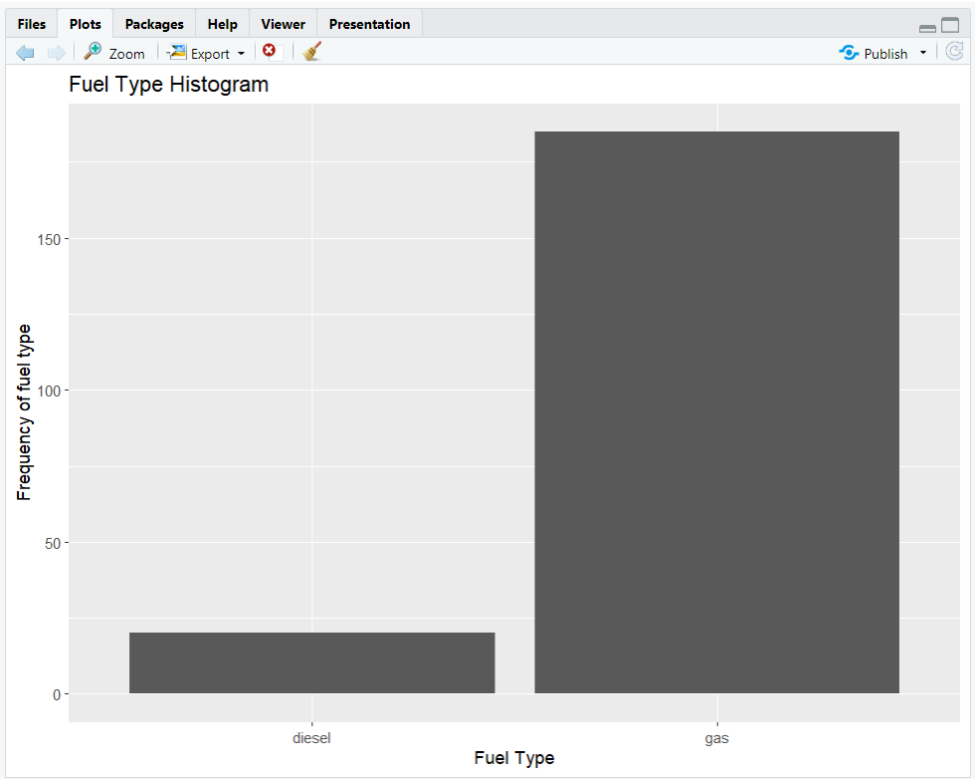


Figure 44: Histogram for fuel type in R

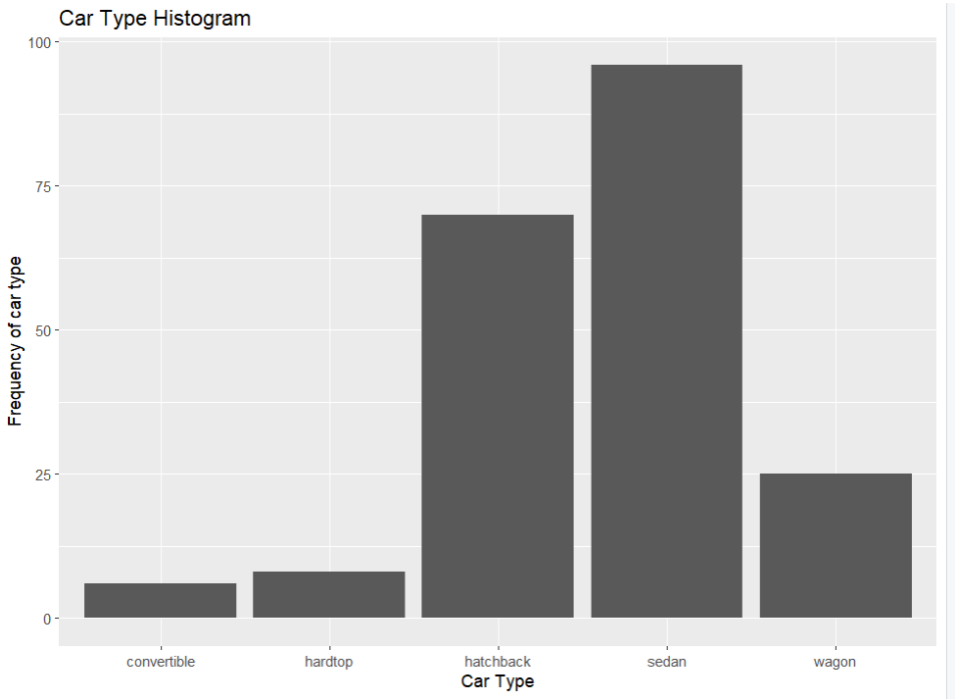


Figure 45: Histogram for Car type in R

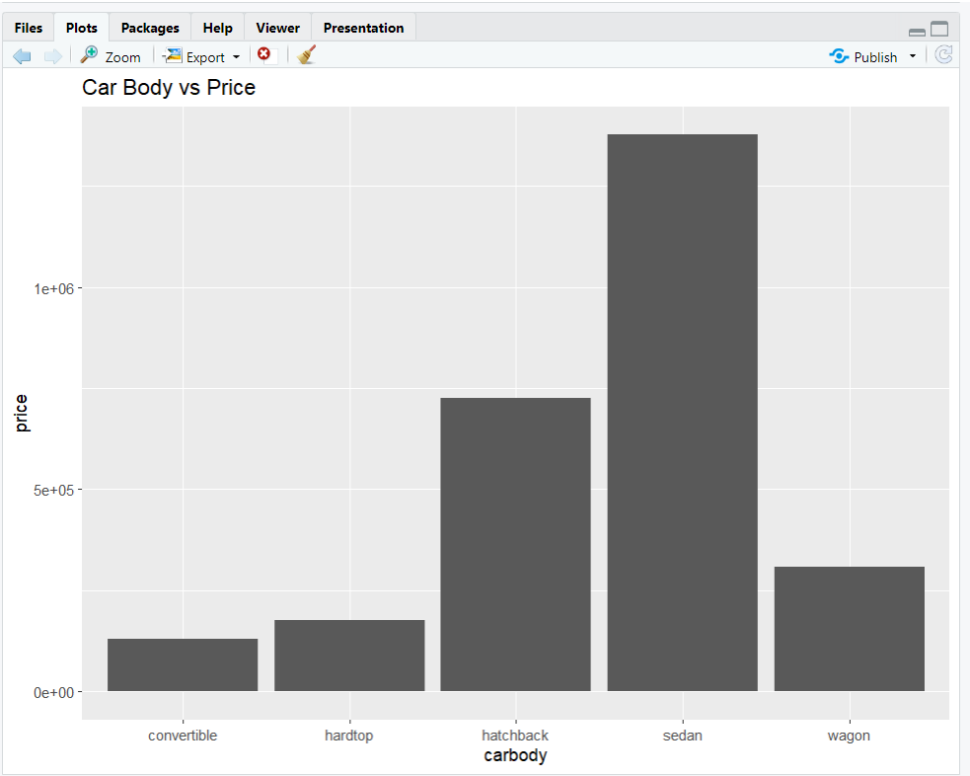


Figure 46: Relation between car body and price

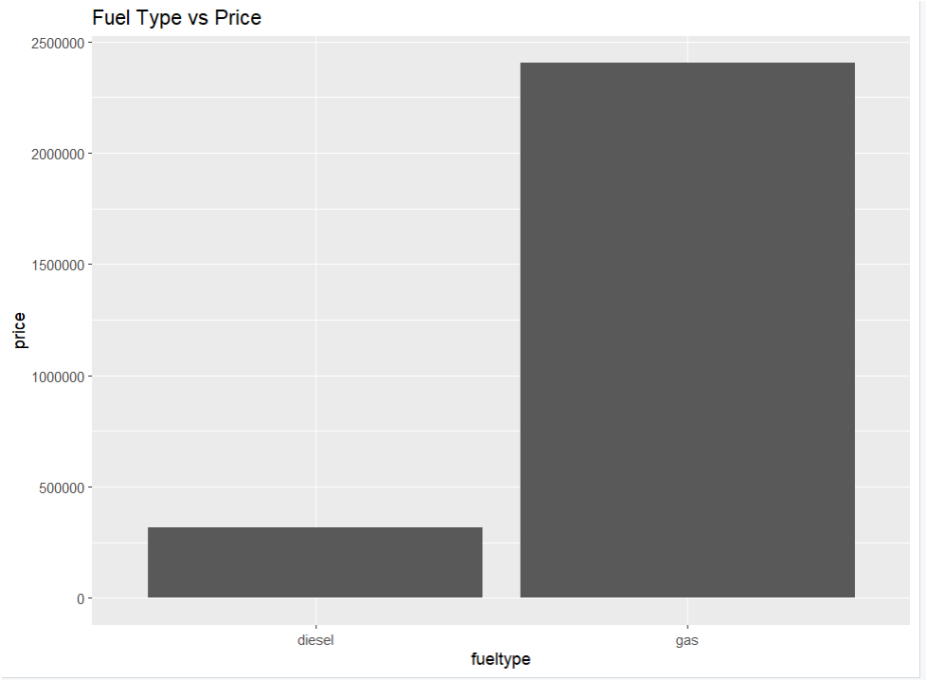


Figure 47: Relation between fuel type and price

5. ANALYSIS USING SQL

SQL (Structured Query Language) is a strong data analysis and manipulation tool for relational databases. We may create the database structure, which includes specifying the tables, columns, and relationships between them. SQL queries can be used to access, filter, and aggregate data from one or more tables once the data has been represented. The most often used SQL command for querying data is the SELECT statement.

I used Oracle SQL to create the schema of the table and import the data. Once the data was imported I ran a few queries for analysis. Below are the code snippets and their outputs.

5.1 SCHEMA FOR THE TABLE CARS

A table schema defines a table's structure, including the names and data types for each column.

```
CREATE TABLE CARS
(
CAR_ID NUMBER(38,0),
SYMBOLING NUMBER(38,0),
CARNAME VARCHAR2(128 BYTE),
FUELTYPE VARCHAR2(26 BYTE),
ASPIRATION VARCHAR2(26 BYTE),
DOORNUMBER VARCHAR2(26 BYTE),
CARBODY VARCHAR2(26 BYTE),
DRIVEWHEEL VARCHAR2(26 BYTE),
ENGINELOCATION VARCHAR2(26 BYTE),
WHEELBASE NUMBER(38,1),
CARLENGTH NUMBER(38,1),
CARWIDTH NUMBER(38,1),
CARHEIGHT NUMBER(38,1),
CURBWEIGHT NUMBER(38,0),
ENGINETYPE VARCHAR2(26 BYTE),
CYLINDERNUMBER VARCHAR2(26 BYTE),
ENGINESIZE NUMBER(38,0),
FUELSYSTEM VARCHAR2(26 BYTE),
BORERATIO NUMBER(38,2),
STROKE NUMBER(38,3),
COMPRESSIONRATIO NUMBER(38,2),
HORSEPOWER NUMBER(38,0),
PEAKRPM NUMBER(38,0),
CITYMPG NUMBER(38,0),
HIGHWAYMPG NUMBER(38,0),
PRICE NUMBER(38,3)
);
```

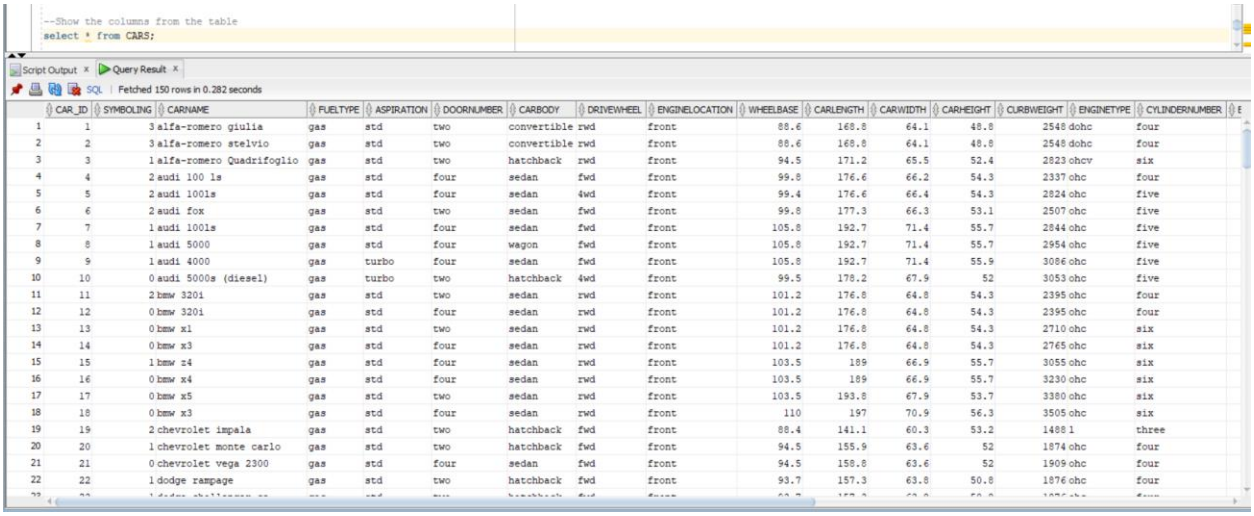
Figure 48: Schema for table CARS

5.2 SELECT STATEMENT

The SELECT statement is used to retrieve data from one or more tables. It lets you select which columns to retrieve, which tables to retrieve data from, and the filters or sorting criteria to use.

```
--Show the columns from the table
select * from CARS;
```

Figure 49: Select statement



```
--Show the columns from the table
select * from CARS;
```

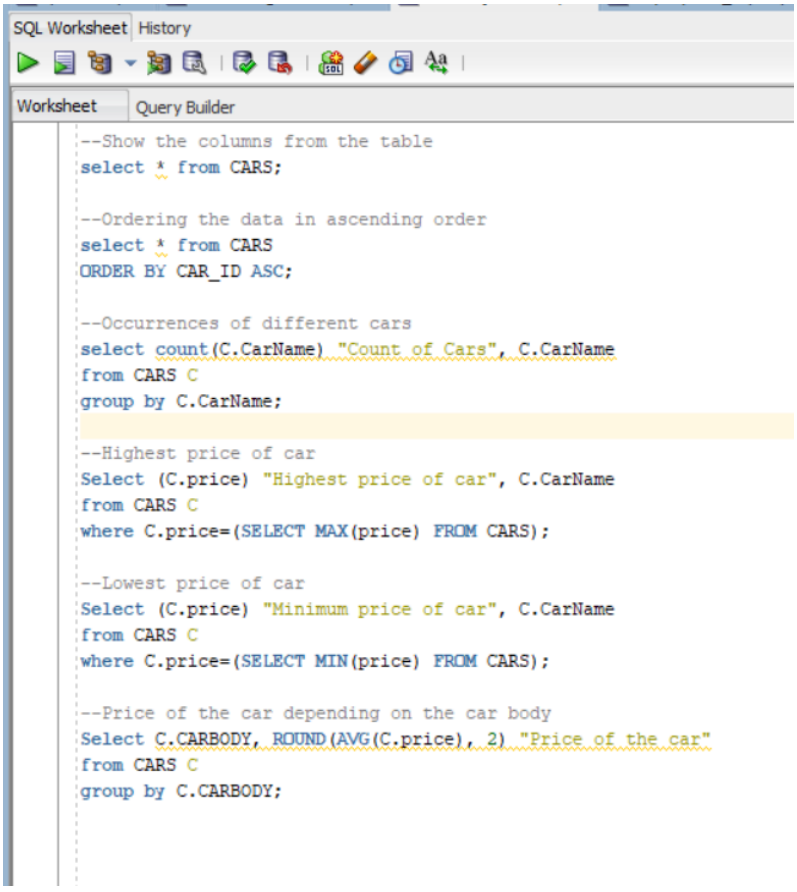
Script Output | Query Result | SQL | Fetched 150 rows in 0.282 seconds

	CAR_ID	SYMBOLING	CARNAME	FUELTYPE	ASPIRATION	DOORNUMBER	CARBODY	DRIVEWHEEL	ENGINELOCATION	WHEELBASE	CARLENGTH	CARWIDTH	CARHEIGHT	CURBWEIGHT	ENGINETYPE	CYLINDERNUMBER
1	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four
2	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four
3	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	171.2	65.5	52.4	2823	ohcv	six
4	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	176.6	66.2	54.3	2337	ohc	four
5	5	2	audi 1001s	gas	std	four	sedan	4wd	front	99.4	176.6	66.4	54.3	2824	ohc	five
6	6	2	audi fox	gas	std	two	sedan	fwd	front	99.8	177.3	66.3	53.1	2507	ohc	five
7	7	1	audi 1001s	gas	std	four	sedan	fwd	front	105.8	192.7	71.4	55.7	2844	ohc	five
8	8	1	audi 5000	gas	std	four	wagon	fwd	front	105.8	192.7	71.4	55.7	2954	ohc	five
9	9	1	audi 4000	gas	turbo	four	sedan	fwd	front	105.8	192.7	71.4	55.9	3086	ohc	five
10	10	0	audi 5000s (diesel)	gas	turbo	two	hatchback	4wd	front	99.5	178.2	67.9	52	3053	ohc	five
11	11	2	bmw 320i	gas	std	two	sedan	rwd	front	101.2	176.8	64.8	54.3	2395	ohc	four
12	12	0	bmw 320i	gas	std	four	sedan	rwd	front	101.2	176.8	64.8	54.3	2395	ohc	four
13	13	0	bmw x1	gas	std	two	sedan	rwd	front	101.2	176.8	64.8	54.3	2710	ohc	six
14	14	0	bmw x3	gas	std	four	sedan	rwd	front	101.2	176.8	64.8	54.3	2765	ohc	six
15	15	1	bmw z4	gas	std	four	sedan	rwd	front	103.5	189	66.9	55.7	3055	ohc	six
16	16	0	bmw x4	gas	std	four	sedan	rwd	front	103.5	189	66.9	55.7	3230	ohc	six
17	17	0	bmw x5	gas	std	two	sedan	rwd	front	103.5	193.8	67.9	53.7	3380	ohc	six
18	18	0	bmw x3	gas	std	four	sedan	rwd	front	110	197	70.9	56.3	3505	ohc	six
19	19	2	chevrolet impala	gas	std	two	hatchback	fwd	front	88.4	141.1	60.3	53.2	1488	1	three
20	20	1	chevrolet monte carlo	gas	std	two	hatchback	fwd	front	94.5	155.9	63.6	52	1874	ohc	four
21	21	0	chevrolet vega 2300	gas	std	four	sedan	fwd	front	94.5	158.8	63.6	52	1909	ohc	four
22	22	1	dodge rampage	gas	std	two	hatchback	fwd	front	93.7	157.3	63.8	50.8	1876	ohc	four

Figure 50: Data display on select table

5.3 ANALYSIS ON THE DATA

We can use various tools to analyze data and gain insights from the datasets. Below are few queries their outputs.



SQL Worksheet | History

Worksheet | Query Builder

```
--Show the columns from the table
select * from CARS;

--Ordering the data in ascending order
select * from CARS
ORDER BY CAR_ID ASC;

--Occurrences of different cars
select count(C.CarName) "Count of Cars", C.CarName
from CARS C
group by C.CarName;

--Highest price of car
Select (C.price) "Highest price of car", C.CarName
from CARS C
where C.price=(SELECT MAX(price) FROM CARS);

--Lowest price of car
Select (C.price) "Minimum price of car", C.CarName
from CARS C
where C.price=(SELECT MIN(price) FROM CARS);

--Price of the car depending on the car body
Select C.CARBODY, ROUND(AVG(C.price), 2) "Price of the car"
from CARS C
group by C.CARBODY;
```

Figure 51: SQL Queries

Worksheet Query Builder

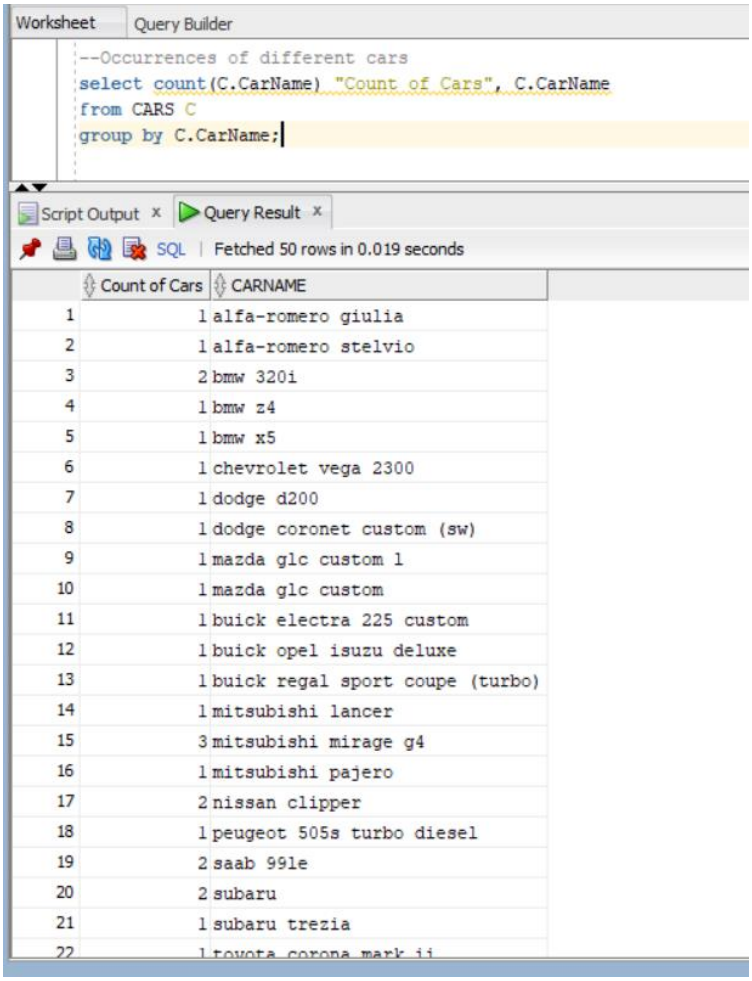
```
--Ordering the data in ascending order
select * from CARS
ORDER BY CAR_ID ASC;
```

Script Output × Query Result ×

SQL | Fetched 50 rows in 0.039 seconds

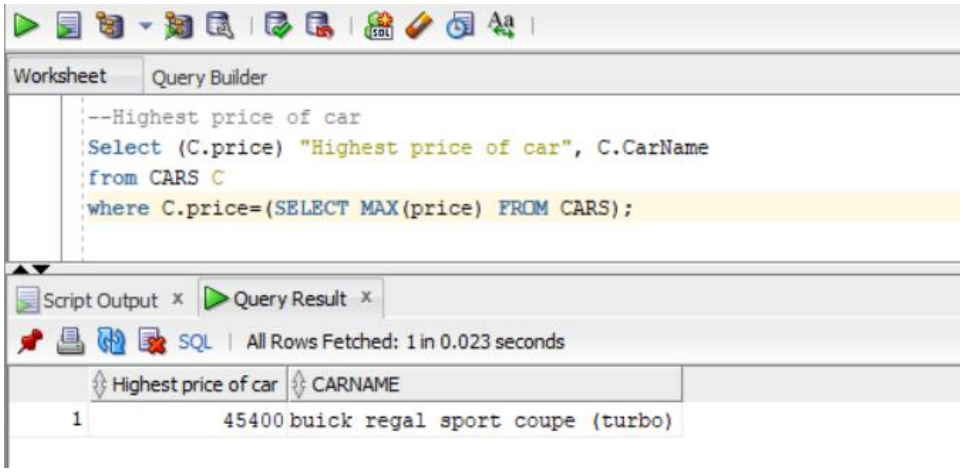
	CAR_ID	SYMBOLING	CARNAME	FUELTYPE	ASPIRATION	DOORNUMBER	CARBODY	DRIVEWHEEL	ENGINELOCATION	WHEELBASE	CARLENGTH	CARWIDTH	CARHEIGHT	CURBWEIGHT	ENGINET
1	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	dohc
2	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	dohc
3	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	171.2	65.5	52.4	2823	ohcv
4	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	176.6	66.2	54.3	2337	ohc
5	5	2	audi 1001s	gas	std	four	sedan	4wd	front	99.4	176.6	66.4	54.3	2824	ohc
6	6	2	audi fox	gas	std	two	sedan	fwd	front	99.8	177.3	66.3	53.1	2507	ohc
7	7	1	audi 1001s	gas	std	four	sedan	fwd	front	105.8	192.7	71.4	55.7	2844	ohc
8	8	1	audi 5000	gas	std	four	wagon	fwd	front	105.8	192.7	71.4	55.7	2954	ohc
9	9	1	audi 4000	gas	turbo	four	sedan	fwd	front	105.8	192.7	71.4	55.9	3086	ohc
10	10	0	audi 5000s (diesel)	gas	turbo	two	hatchback	4wd	front	99.5	178.2	67.9	52	3053	ohc
11	11	2	bmw 320i	gas	std	two	sedan	rwd	front	101.2	176.8	64.8	54.3	2395	ohc
12	12	0	bmw 320i	gas	std	four	sedan	rwd	front	101.2	176.8	64.8	54.3	2395	ohc
13	13	0	bmw x1	gas	std	two	sedan	rwd	front	101.2	176.8	64.8	54.3	2710	ohc
14	14	0	bmw x3	gas	std	four	sedan	rwd	front	101.2	176.8	64.8	54.3	2765	ohc
15	15	1	bmw z4	gas	std	four	sedan	rwd	front	103.5	189	66.9	55.7	3055	ohc
16	16	0	bmw x4	gas	std	four	sedan	rwd	front	103.5	189	66.9	55.7	3230	ohc
17	17	0	bmw x5	gas	std	two	sedan	rwd	front	103.5	193.8	67.9	53.7	3380	ohc
18	18	0	bmw x3	gas	std	four	sedan	rwd	front	110	197	70.9	56.3	3505	ohc
19	19	2	chevrolet impala	gas	std	two	hatchback	fwd	front	88.4	141.1	60.3	53.2	1488	l
20	20	1	chevrolet monte carlo	gas	std	two	hatchback	fwd	front	94.5	155.9	63.6	52	1874	ohc
21	21	0	chevrolet vega 2300	gas	std	four	sedan	fwd	front	94.5	158.8	63.6	52	1909	ohc
22	22	1	dodge rampage	gas	std	two	hatchback	fwd	front	93.7	157.3	63.8	50.8	1876	ohc

Figure 52: Ordering the data in ascending order



	Count of Cars	CARNAME
1	1	alfa-romero giulia
2	1	alfa-romero stelvio
3	2	bmw 320i
4	1	bmw z4
5	1	bmw x5
6	1	chevrolet vega 2300
7	1	dodge d200
8	1	dodge coronet custom (sw)
9	1	mazda glc custom l
10	1	mazda glc custom
11	1	buick electra 225 custom
12	1	buick opel isuzu deluxe
13	1	buick regal sport coupe (turbo)
14	1	mitsubishi lancer
15	3	mitsubishi mirage g4
16	1	mitsubishi pajero
17	2	nissan clipper
18	1	peugeot 505s turbo diesel
19	2	saab 99le
20	2	subaru
21	1	subaru trezia
22	1	toyota corona mark ii

Figure 53: Occurrences of different cars



	Highest price of car	CARNAME
1	45400	buick regal sport coupe (turbo)

Figure 54: Highest price of car

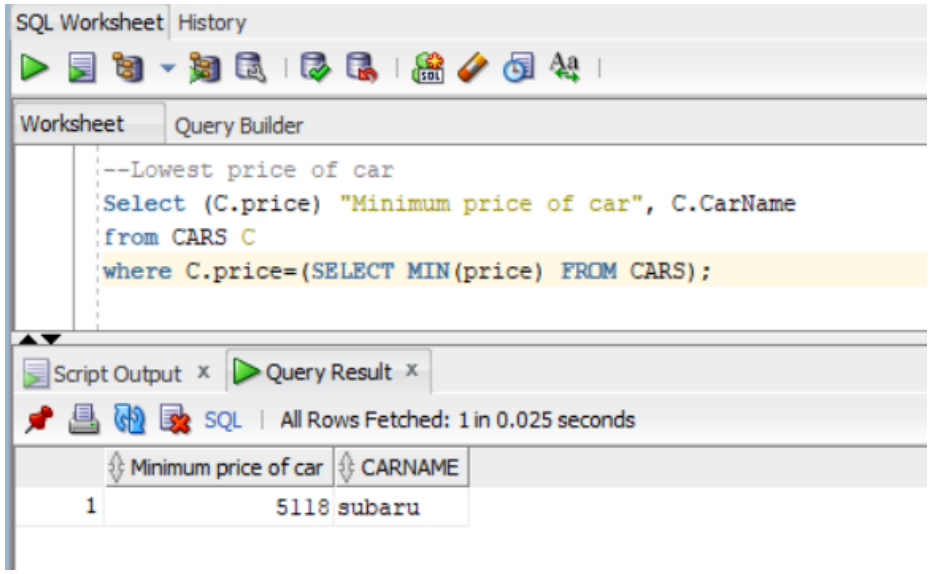


Figure 55: Lowest price of car

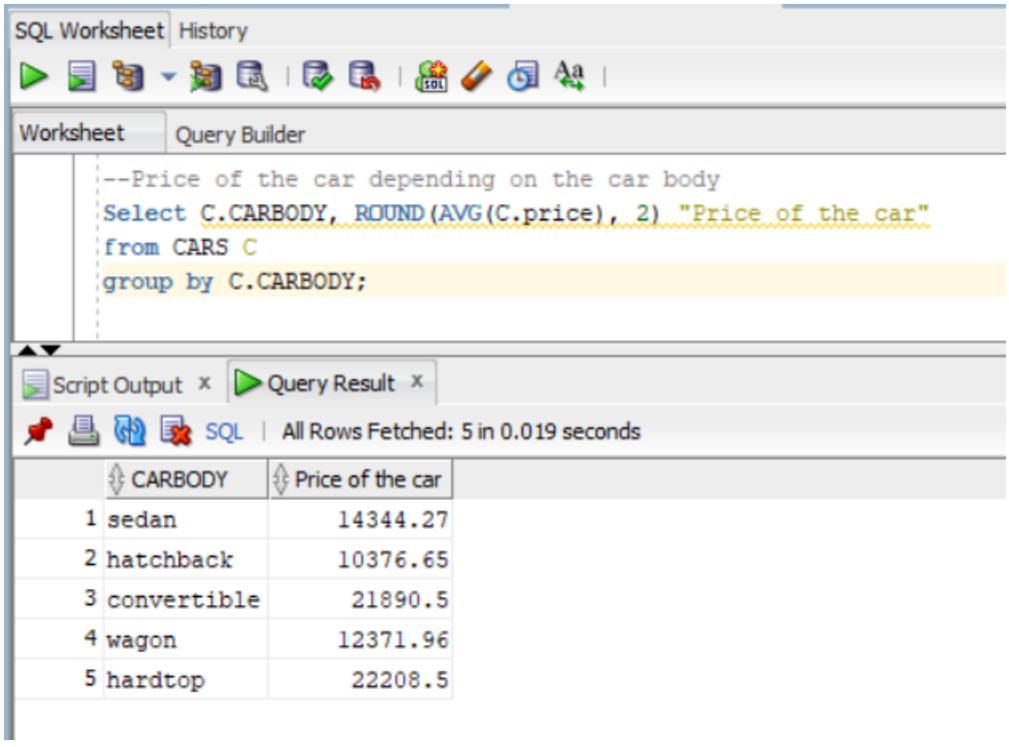


Figure 56: Relation between Price and car body

6. CONCLUSION

Aspiration, enginelocation, carwidth, curbweight, enginetype, cylindernumber, stroke, peakrpm, and price are all useful variables for describing pricing variations in cars. The linear regression model met the traditional assumptions. The model's R-squared is strong, with 90.72% of the variables explaining the price variations. The model's accuracy in predicting automobile prices is tested using RMSE; testing data had an RMSE of 2978.49. The training set has an R2 value of 0.92, whereas the test set has an R2 score of 0.87, which is quite close. As a result, we can assert that our model is good enough to estimate automobile prices using the factors listed above.

7. FUTURE SCOPE

In the future, machine learning models may be linked to a variety of websites that provide real-time data for price prediction. We may also contribute vast volumes of historical data on automobile prices to help improve the accuracy of the machine learning models. To improve performance, we intend to create deep learning network architectures, employ variable learning rates, and train on data clusters rather than the entire dataset. Therefore, a larger data set is always preferable. More training data can be fed into the model using a larger dataset. After data pre-processing, the dataset used in the study contained 205 samples. For a more accurate model, a larger dataset might be used. Internal elements such as the infotainment cluster, ergonomics, and GPS availability etc. can also be considered for price predictions.

8. REFERENCES

[1]Noor, Kanwal, and Sadaqat Jan. "Vehicle Price Prediction System Using Machine Learning Techniques." International Journal of Computer Applications, vol. 167, no. 9, 15 June 2017, pp. 27–31, Vehicle Price Prediction System using Machine Learning Techniques (ijcaonline.org)

[2] Li, Qilin. "US Auto Production and Price Prediction in the Context of Multiple Regression Analysis." BCP Business & Management, vol. 38, no. 10.54691/bcpbm.v38i.3790, 2 Mar. 2023, pp. 875–880,

https://doi.org/10.54691/bcpbm.v38i.3790. Accessed 31 Mar. 2023.

[3] Al-Turjman, Fadi, et al. "Vehicle Price Classification and Prediction Using Machine Learning in the IoT Smart Manufacturing Era." Sustainability, vol. 14, no. 15, 26 July 2022, p. 9147, (PDF) Vehicle Price Classification and Prediction Using Machine Learning in the IoT Smart Manufacturing Era (researchgate.net)

[4]Gegic, Enis, et al. "Car Price Prediction Using Machine Learning Techniques." *TEM Journal*, vol. 8, no. 1, 2019, pp. 113–118, www.temjournal.com/content/81/TEMJournalFebruary2019_113_118.pdf,

https://doi.org/10.18421/TEM81-16.