FLIP ROBO

Car Price Prediction Project

Submitted by:

Debanti Roy

ACKNOWLEDGMENT

I, Debanti Roy, would like to convey my sincere gratitude to DataTrained Academy and Flip Robo Technologies for giving me this opportunity to do this project. I would like to thank all mentors and SME for extending their support all through the process which helped me complete this project.

INTRODUCTION

Business Problem Framing

With the COVID-19 impact in the market, we have seen lot of changes in the car market. Now some cars are in demand hence making them costly and some are not in demand hence cheaper.

Conceptual Background of the Domain Problem

The Used Car Market in India is Segmented by different Vehicle Types, Owner Type (1^{st}, 2^{nd}, 3^{rd} and 4^{th}), Manual or Automatic, and Fuel Type (Petrol and Diesel), etc. As per reports by Modor Intelligence, the Indian used car market was valued at USD 32.14 billion in 2021, and it is expected to reach USD 74.70 billion in 2027, registering a CAGR of 15.1% during the forecast period (2022-2027).

The COVID-19 pandemic had a minimal impact on the industry. With the increased number of people preferring individual mobility and more finance options available in the used car market, the market is set to grow considerably.

Reduced cash inflow due to the pandemic has forced buyers to look for alternatives other than new cars, and the used car industry has high growth potential in these terms.

One of our clients works with small traders, who sell used cars. With the change in market due to COVID-19 impact, our client is facing problems with their previous car price valuation machine learning models. So, they are looking for new machine learning models from new data. We are to help the client get a better understanding of the market.

The sample data I used to build different predictive models was scraped from a popular used car selling and purchasing site, OLX.

Review of Literature

In order to invest in any field market analysis is essential to understand the dynamics of the market. Here our client who works with small traders selling used cars needs to understand which factors can affect the price of a used car. Predicting the price gives them a fair idea where to invest and which cars would be profitable. The literature attempts to derive useful knowledge from historical data of same market. Machine learning techniques are applied to analyze historical transactions to discover useful models for price prediction.

Motivation for the Problem Undertaken

Here we are to understand which variables are significant in predicting the price of a car. This will help our client understand the relevant factors and make a prediction of the car prices.

Analytical Problem Framing

As we have scraped the data from a site, it means our data is unsupervised and not ready for machine learning. Thus, I have performed both univariate and bivariate analysis to analyse these values using different plots like distribution and box plot.

In this project I have also done various mathematical and statistical analysis such as describing the statistical summary of the columns in which I found that the data has outliers. I used label encoding method to convert the object data into numerical data. Checked for correlation between the features and visualized it using heatmap.

Data Sources and their formats

The data was scraped from OLX. Results indicate the price of cars based on several features. The dataset is in CSV format. The data contains 5000 rows and 8 columns.

Data Preprocessing Done

- **Importing the necessary libraries and packages**

First we have imported the necessary libraries and packages

```
#Importing the necessaary libraries and packages
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings("ignore")

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import LabelEncoder
```

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LinearRegression
import statsmodels.api as sm
from sklearn.feature_selection import RFE
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
```

Then we have imported our dataset which was in CSV format and printed the shape of the dataset, i.e., the total rows and columns.

```
#loading the train and test datasets
car_df=pd.read_csv(r"Used Car Details")


print("Shape of the dataset:", car_df.shape)
```

```
Shape of the dataset: (5000, 8)
```

We can see the dataset has 5000 rows and 8 columns.

- **EDA**

Next we have printed the head, tail and sample dataset to get a general understanding of the data values.

```
#printing the head of the dataset
car_df.head()
```

	Unnamed: 0	Brand	Variant	Fuel	Km_driven	Owner	Location	Price
0	0	Honda Amaze (2019)	MANUAL	DIESEL	69511.0 KM	1st	Location\nRTC Colony, Hyderabad	₹ 7,55,000
1	1	Maruti Suzuki Alto K10 (2018)	MANUAL	PETROL	57071.0 KM	1st	Location\nSindhu Nagar, Bhilwara	₹ 3,30,000
2	2	Mahindra Bolero Power Plus (2020)	MANUAL	DIESEL	67000.0 KM	--	Location\nPolo Field, Tezpur	₹ 8,65,000
3	3	Toyota Innova (2008)	MANUAL	DIESEL	120000 KM	Second	Location\nHussaini Alam, Hyderabad	₹ 2,70,000
4	4	Maruti Suzuki 1000 (1999)	MANUAL	PETROL	40000 KM	Second	Location\nSector 2A, Chandigarh	₹ 50,000

Here we find that the column 'Unnamed: 0' can be dropped as the first one contains the numbers of the rows in the dataset. Thus, we will drop it as it is not necessary for prediction.

```
#Dropping unimportant column
car_df.drop(['Unnamed: 0'], axis=1, inplace=True)
```

We have also checked null values in the dataset through the '.isnull().sum()' and '.isnull().sum().sum()' methods. Both the methods indicate that the dataset has null values.

```
#checking null values
car_df.isnull().sum()
```

```
Unnamed: 0      0
Brand         139
Variant       139
Fuel          139
Km_driven     139
Owner         139
Location      139
Price         139
dtype: int64
```

```
#checking the total no. of null values
car_df.isnull().sum().sum()
```

```
973
```

We have dropped the null values using '.dropna()' method.

```
# drop all rows with any null values
df = car_df.dropna()
```

We have also extracted car brand using split function from the 'Brand' column and save the details in a new colum. Hence, we will drop the 'Brand' column.

```
#Extracting Brand name of the car in a new column Car_brand.
df['Car_brand'] = df['Brand'].apply(lambda x:x.split('(')[0])
```

```
#Dropping the original column brand
df = df.drop('Brand',axis=1)
```

We have used replace function thereafter to replace certain "—" present in certain cells with "NA".

```
#replace all the  '--' with 'NA'
df= df.replace('--', 'NA')
```

Henceforth, I used replace function to remove rupee sign from 'Price' column and 'KM' from 'Km_driven' column.

```
#replacing ',' and '₹' sign with blank in price column and converting the same into integer
df['Price'] = df['Price'].str.replace(',', '')
df['Price'] = df['Price'].str.replace('₹', '')
df['Price'] = df['Price'].astype(int)
```

```
#replacing ',' and 'KM' sign with blank in price column
df['Km_driven'] = df['Km_driven'].str.replace(',', '')
df['Km_driven'] = df['Km_driven'].str.replace('KM', '')
```

As I proceeded forward I observed the 'Owner' column has certain cells that are mentioned as First, Second, Third, Fourth contrary to the usual 1st, 2nd, 3rd, etc. I have, thus, replaced them with the latter values.

```
df['Owner'] = df['Owner'].replace('First', '1st')
```

```
df['Owner'] = df['Owner'].replace('Second', '2nd')
```

```
df['Owner'] = df['Owner'].replace('Third', '3rd')
```

```
df['Owner'] = df['Owner'].replace('Fourth', '4th')
```

```
df['Owner'] = df['Owner'].replace('NA', '4th')
```

We can see that there are certain cells that have '--' inplace of values, which we will try to replace with 0 using different methods.

```
df1=df.fillna(0)
```

```
df1['Km_driven']=df1['Km_driven'].apply(lambda x: x.replace(',','') if x!='-' else '-')
```

```
df1['Km_driven'] = df1['Km_driven'].replace('--', '0')
```

```
df1['Km_driven'] = df1['Km_driven'].fillna('0')
```

We have then separated the categorical and numerical values. And finally used describe method to understand the statistical details of our target column 'Price'.

```
numerical_cols['Price'].describe()
```

```
count    4.861000e+03
mean     8.696927e+05
std      1.366803e+06
min      5.000000e+04
25%      2.100000e+05
50%      4.750000e+05
75%      8.100000e+05
max      7.500000e+06
Name: Price, dtype: float64
```

Consequently, I have also used replace function on the ‘Fuel’, ‘Variant’ and ‘Owner’ column wherever there were NA or similar values.

After completing all the above preprocessing, I finally used label encoder to encode the categorical columns.

```
#Taking care of categorical columns using label encoder
from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()
for i in df_final.columns:
    if df_final[i].dtypes=="object":
        df_final[i]=le.fit_transform(df_final[i])
```

Data Inputs- Logic- Output Relationships

- **Feature and Target Value**

We are now ready to prepare our data for model building. Let’s start splitting the data into train and test. Here we have used 70% data for training and 30% for testing.

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LinearRegression
import statsmodels.api as sm
from sklearn.feature_selection import RFE
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
```

```
df_train, df_test = train_test_split(df_final, train_size = 0.7, test_size = 0.3, random_state = 100)
```

For building the model our feature variables are saved in x and target variable is saved in y.

```
y_train = df_train.pop('Price')
X_train = df_train
```

Thereafter, we applied Standardization of the features variables. Standardization entails scaling data to fit a standard normal distribution.

```
scaler = StandardScaler()
```

```
df_train[col_list] = scaler.fit_transform(df_train[col_list])
```

The first step is to import the necessary packages that enable training and testing which we have done right at the beginning.

- **Hardware and Software Requirements and Tools Used**

  Hardware required:

a. Processor: core i5 or above
b. RAM: 8 GB or above
c. ROM/SSD: 250 GB or above

  Software required:

d. Anaconda 3- language used Python 3
e. Microsoft Excel Libraries: The important libraries that I have used for this project are below:

import numpy as np

It is defined as a Python package used for performing various numerical computations and processing of the multidimensional and single dimensional array elements. The calculations using Numpy arrays are faster than the normal Python array.

import pandas as pd

Pandas is a Python library that is used for faster data analysis, data cleaning and data pre-processing. The data-frame term is coming from Pandas only.

import matplotlib.pyplot as plt and import seaborn as sns

Matplotlib and Seaborn acts as the backbone of data visualization through Python.

Matplotlib: It is a Python library used for plotting graphs with the help of other libraries like Numpy and Pandas. It is a powerful tool for visualizing data in Python. It is used for creating statical interferences and plotting 2D graphs of arrays.

Seaborn: It is also a Python library used for plotting graphs with the help of Matplotlib, Pandas, and Numpy. It is built on the roof of Matplotlib and is considered as a superset of the Matplotlib library. It helps in visualizing univariate and bivariate data.

from sklearn.preprocessing import LabelEncoder

There are several encoding techniques like Label Encoder, OneHotEncoder, Ordinal Encoder.

In this project I have used LabelEncoder technique to convert categorical data or object type data into numerical data.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
- I have used ".drop()" and ".dropna()" functions to drop unwanted entries in the columns.
- Used "LabelEncoder" method to encode the columns.
- I have used replace function to replace a column with appropriate values.
- Described the statistical details of the features using ".describe()" method.
- I also used ".count()" to get a detailed understanding of the no. of each type of data present in our dataset.
- To check null values I have used ".isnull().sum()" and ".isnull().sum().sum()".
- Used "Pearson's method" to check the correlation between the features.
- Performed both univariate and bivariate analysis using seaborn and matplotlib.

Testing of Identified Approaches (Algorithms)

I have tested the data using RFE.

Model 1 Results:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.165
Model:                            OLS   Adj. R-squared:                  0.163
Method:                 Least Squares   F-statistic:                     111.4
Date:                Sat, 17 Sep 2022   Prob (F-statistic):          1.16e-128
Time:                        02:58:18   Log-Likelihood:                -4521.4
No. Observations:                3402   AIC:                             9057.
Df Residuals:                    3395   BIC:                             9100.
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.1737      0.075     15.554      0.000       1.026       1.322
Variant       -0.1475      0.042     -3.515      0.000      -0.230      -0.065
Fuel           0.0220      0.026      0.832      0.405      -0.030       0.074
Km_driven     -0.0100      0.002     -6.303      0.000      -0.013      -0.007
Owner         -0.1885      0.014    -13.085      0.000      -0.217      -0.160
Location      -0.0157      0.002     -9.397      0.000      -0.019      -0.012
Car_brand     -0.0332      0.002    -18.894      0.000      -0.037      -0.030
==============================================================================
Omnibus:                     2046.412   Durbin-Watson:                   1.981
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            16896.462
Skew:                           2.849   Prob(JB):                         0.00
Kurtosis:                      12.312   Cond. No.                         155.
==============================================================================
```

Model 2 Results:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.165
Model:                            OLS   Adj. R-squared:                  0.163
Method:                 Least Squares   F-statistic:                     111.4
Date:                Sat, 17 Sep 2022   Prob (F-statistic):          1.16e-128
Time:                        02:59:02   Log-Likelihood:                -4521.4
No. Observations:                3402   AIC:                             9057.
Df Residuals:                    3395   BIC:                             9100.
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.1737      0.075     15.554      0.000       1.026       1.322
Variant       -0.1475      0.042     -3.515      0.000      -0.230      -0.065
Fuel           0.0220      0.026      0.832      0.405      -0.030       0.074
Km_driven     -0.0100      0.002     -6.303      0.000      -0.013      -0.007
Owner         -0.1885      0.014    -13.085      0.000      -0.217      -0.160
Location      -0.0157      0.002     -9.397      0.000      -0.019      -0.012
Car_brand     -0.0332      0.002    -18.894      0.000      -0.037      -0.030
==============================================================================
Omnibus:                     2046.412   Durbin-Watson:                   1.981
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            16896.462
Skew:                           2.849   Prob(JB):                         0.00
Kurtosis:                      12.312   Cond. No.                         155.
==============================================================================
```

Model 3 Results:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.164
Model:                            OLS   Adj. R-squared:                  0.163
Method:                 Least Squares   F-statistic:                     133.6
Date:                Sat, 17 Sep 2022   Prob (F-statistic):          1.34e-129
Time:                        02:59:06   Log-Likelihood:                -4521.8
No. Observations:                3402   AIC:                             9056.
Df Residuals:                    3396   BIC:                             9092.
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.2068      0.064     18.807      0.000       1.081       1.333
Variant       -0.1495      0.042     -3.569      0.000      -0.232      -0.067
Km_driven     -0.0100      0.002     -6.269      0.000      -0.013      -0.007
Owner         -0.1888      0.014    -13.110      0.000      -0.217      -0.161
Location      -0.0158      0.002     -9.513      0.000      -0.019      -0.013
Car_brand     -0.0330      0.002    -18.889      0.000      -0.036      -0.030
==============================================================================
Omnibus:                     2054.013   Durbin-Watson:                   1.983
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            17157.279
Skew:                           2.858   Prob(JB):                         0.00
Kurtosis:                      12.400   Cond. No.                         136.
==============================================================================
```

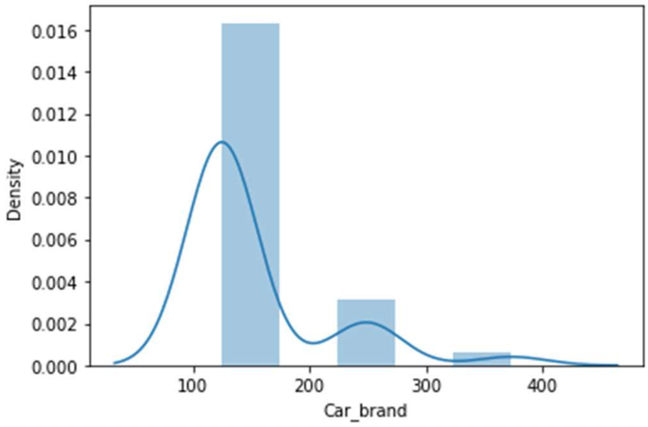
Model 4 Results:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.161
Model:                            OLS   Adj. R-squared:                  0.160
Method:                 Least Squares   F-statistic:                     163.3
Date:                Sat, 17 Sep 2022   Prob (F-statistic):          5.52e-128
Time:                        02:59:09   Log-Likelihood:                -4528.2
No. Observations:                3402   AIC:                             9066.
Df Residuals:                    3397   BIC:                             9097.
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.0570      0.049     21.740      0.000       0.962       1.152
Km_driven     -0.0106      0.002     -6.698      0.000      -0.014      -0.008
Owner         -0.1825      0.014    -12.746      0.000      -0.211      -0.154
Location      -0.0139      0.002     -8.818      0.000      -0.017      -0.011
Car_brand     -0.0327      0.002    -18.694      0.000      -0.036      -0.029
==============================================================================
Omnibus:                     2023.601   Durbin-Watson:                   1.975
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            16423.583
Skew:                           2.814   Prob(JB):                         0.00
Kurtosis:                      12.175   Cond. No.                         92.8
==============================================================================
```

- **Visualizations**

  To understand any kind of data it is important to perform Exploratory data analysis (EDA). This is a combination of visualizations and statistical analysis (uni, bi, and multivariate) that helps us to better understand the data we are working with and to gain insight into their relationships. So, let's explore our target variable and how the other features influence it.

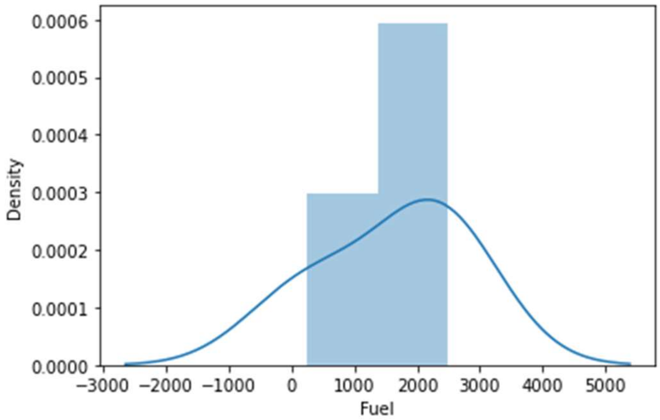
We have used bar plots to visualise the data. ‘




```
# Brand type
df_final['Car_brand'].value_counts()
```

```
Hyundai Santro Xing           373
Volkswagen Polo               250
Hyundai Verna                 249
Maruti Suzuki 800             249
Hyundai Creta                 249
Honda Amaze                   247
Ford Figo                     125
Maruti Suzuki Swift           125
Toyota Etios Liva             125
Honda Accord                  125
Mahindra Rexton               125
Honda City                    125
Nissan Micra                  125
Hyundai I10                   125
Maruti Suzuki Alto K10        125
Volkswagen Vento              125
Maruti Suzuki Ciaz            125
Toyota Etios                  125
Maruti Suzuki Alto 800        125
Mercedes-Benz Gle Class       125
Bmw 5 Series                  125
Bmw M2                        125
Skoda Superb                  125
Maruti Suzuki 1000            125
Toyota Innova                 125
Mahindra Bolero Power Plus    125
Tata Sumo                     124
Maruti Suzuki Baleno          124
Kia Seltos                    124
Hyundai Venue                 124
Hyundai Fluidic Verna         124
Audi A6                       124
Name: Car_brand, dtype: int64
```

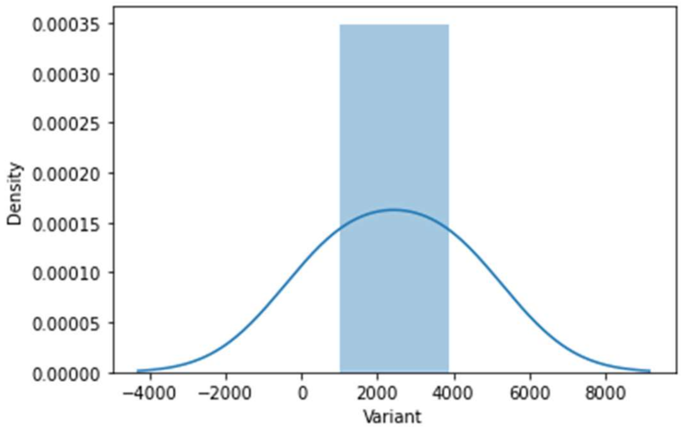
Here we can see as per the Car Brand count Hyundai Santro xing is the most available followed by Volkswagen Polo, Hyundai Verna, etc. Tata sumo, Maruti Suzuki Baleno, Kia Seltos, Hyundai Venue, Hyundai Fluidic Verna, Audi A6 has the least count.




```
df_final['Fuel'].value_counts()
```

```
PETROL     2494
DIESEL     2117
CNG         250
Name: Fuel, dtype: int64
```

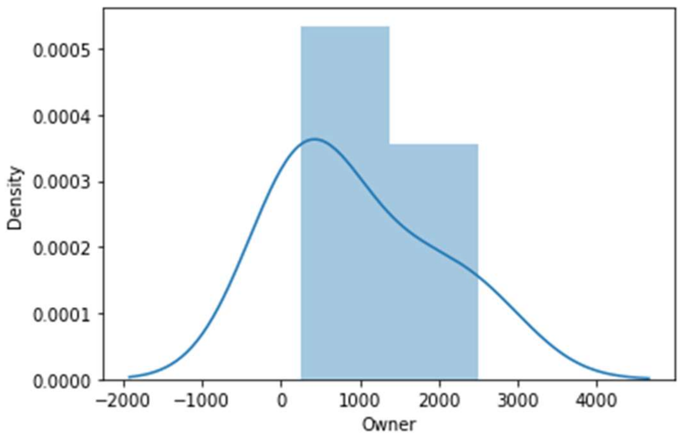
We can see Petrol cars are mostly available followed by diesel and CNG.




```
df_final['Variant'].value_counts()
```

```
MANUAL       3863
AUTOMATIC     998
Name: Variant, dtype: int64
```

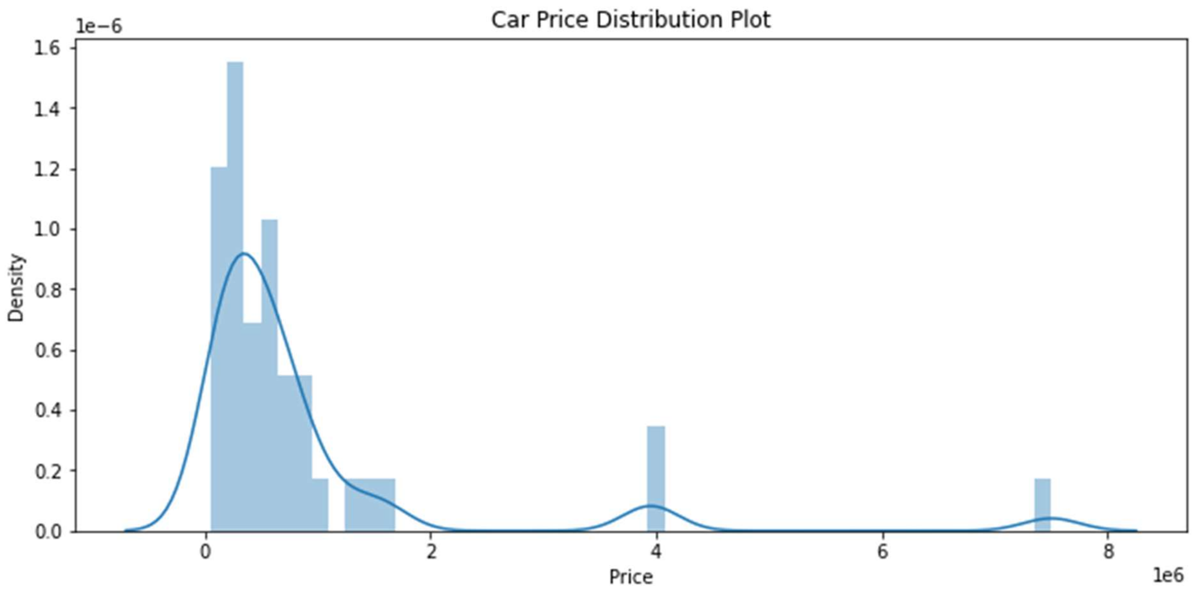
We can see more manual cars available followed by automatic.



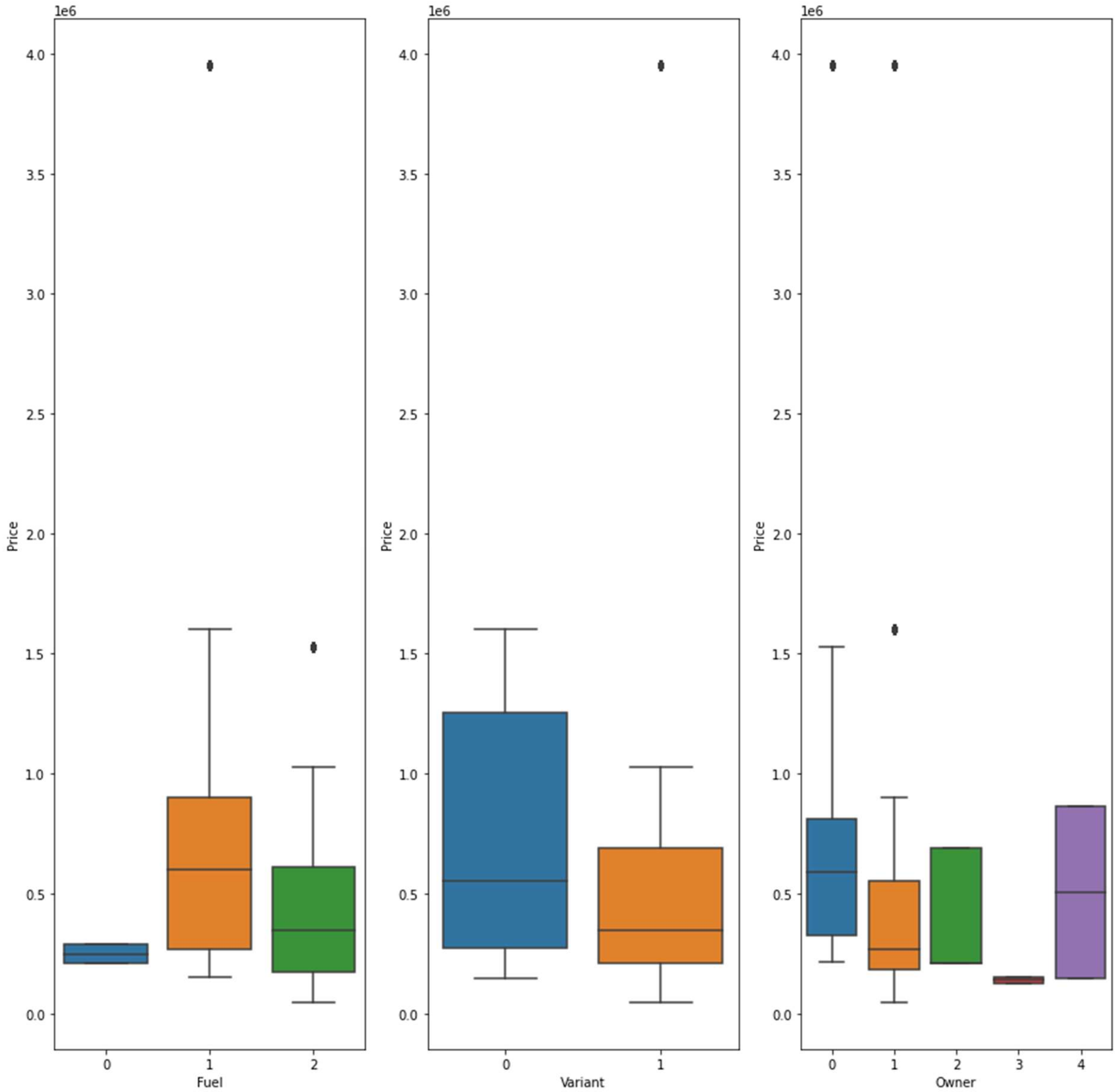

```
# owner type
df_final['Owner'].value_counts()
```

```
1st    2493
2nd    1495
3rd     375
4th     250
4+      248
Name: Owner, dtype: int64
```

The cars are mostly sold by first and second owners.



This plot shows the distribution of the target feature 'Price'. We can definitely say that the cost of used cars is definitely high as we see some cars priced at Rs. 3300000 or above.



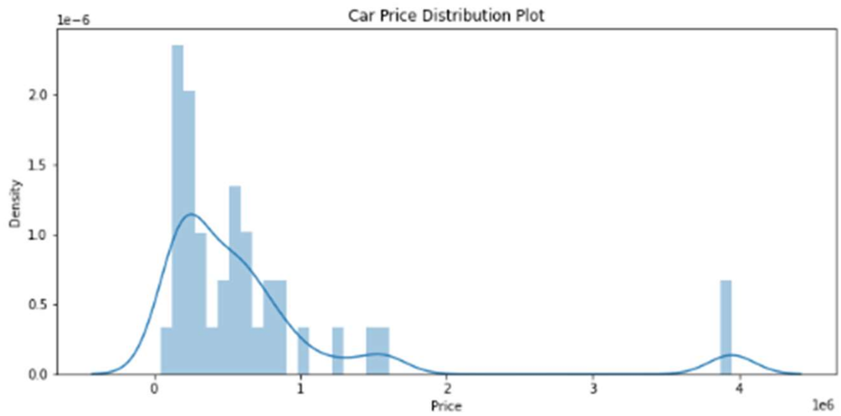
The above plots show a relation between car fuel, variant, owner vs price.

Checking & Treating Outliers

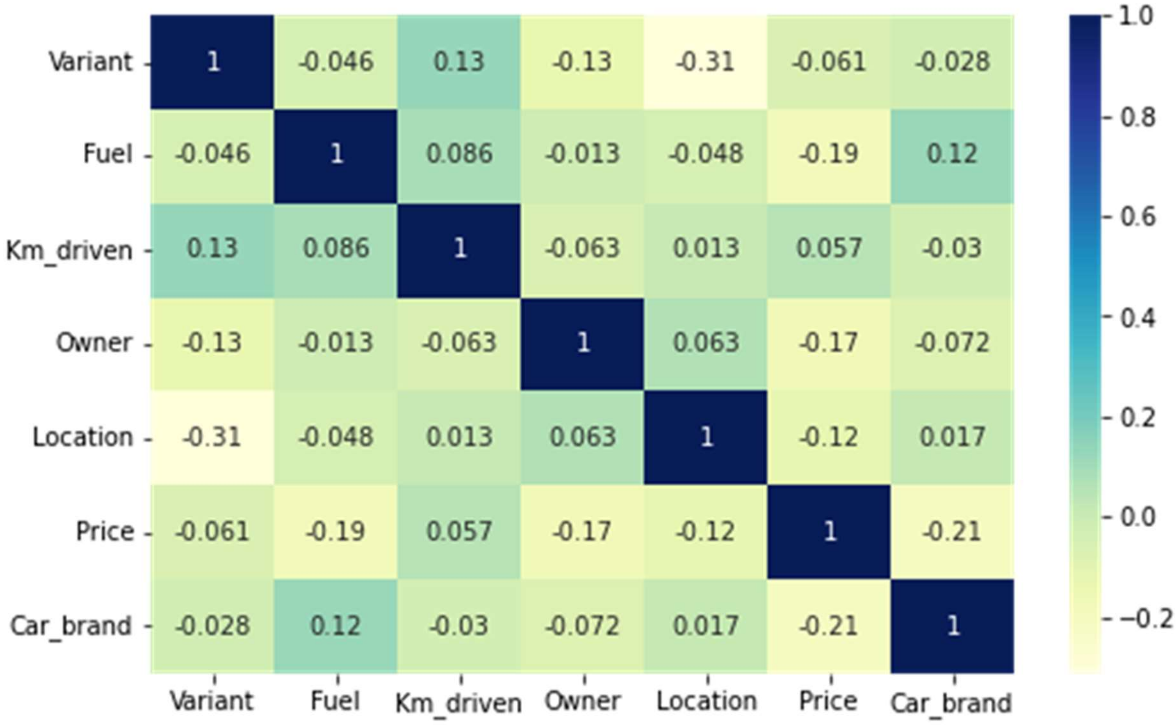
```
# Here, the outliers are situated around the higher prices (right side of the graph)
# we can deal with the problem easily by removing 0.5%, or 1% of the problematic samples
# This is a dataset about used cars, therefore one can imagine how ₹300,000 is an excessive price
# Outliers are a great issue for OLS, thus we must deal with them in some way

# Let's declare a variable that will be equal to the 99th percentile of the 'Price' variable
q = df_final['Price'].quantile(0.99)

# Then we can create a new df, with the condition that all prices must be below the 99 percentile of 'Price'
data_1 = df_final[df_final['Price']<q]
```

```
plt.figure(figsize=[11,5])
sns.distplot(data_1['Price'])
plt.title('Car Price Distribution Plot')
```

```
Text(0.5, 1.0, 'Car Price Distribution Plot')
```



Next, we plotted a heatmap of the data to get an idea of the correlation.



We have used a heatmap to understand the correlation of the data.

Price is highly (positively) correlated with km driven.

Price is negatively correlated to car brand, variant, owner, fuel, location, etc.

This suggest that cars may fall in the 'economy' cars category, and are priced lower.

Interpretation of the Results

We can see that our models indicated fuel, car brand, km driven are important parameters that can affect the price of a car.

Conclusion

- Through this project I was able to understand the factors which variables are significant in predicting the price of a car?
- The evaluation of the data in training and testing mode reveals 0.21487240894719895.