NAME OF THE PROJECT

Car Price Prediction using Machine Learning

Submitted by:

Mrs. Swati Amit Motugade

FLIPROBO SME:

Ms. Gulshana Chaudhari

ACKNOWLEDGMENT

I would like to express my special gratitude to “Flip Robo” team, who has given me this opportunity to deal with a beautiful dataset and it has helped me to improve my analysis skills. Also, I want to express my huge gratitude to Ms. Gulshana Chaudhari Mam (SME, Flip Robo), she is the person who has helped me to get out of all the difficulties I faced while doing this project.

A huge thanks to “Data trained” who are the reason behind Internship at Fliprobo. Last but not least my parents who are there to support me at every step of my life.

References used in this project:

- SCIKIT Learn Library Documentation.
- Blogs from towardsdatascience, Analytics Vidya, Medium.
- Andrew Ng Notes on Machine Learning (GitHub).
- Data Science Projects with Python Second Edition by Packt
- Hands on Machine learning with scikit learn and tensor flow by Aurelien Geron.
- TEM Journal on Car Price Prediction using ML Techniques by Enis Gegic, Becir Isakovic, Dino Keco, Zerina Masetic, Jasmin Kevric.
- Also, there are so many people who helped me directly and indirectly to complete this project.

Mrs. Swati Amit Motugade

Chapter 1

Introduction

1.1 Business Problem Framing

Car price prediction is somehow interesting and popular problem. As per information that was gotten from the Agency for Statistics of BiH, 921.456 vehicles were registered in 2014 from which 84% of them are cars for personal usage [1]. This number is increased by 2.7% since 2013 and it is likely that this trend will continue, and the number of cars will increase in future. This adds additional significance to the problem of the car price prediction.

Accurate car price prediction involves expert knowledge, because price usually depends on many distinctive features and factors. Typically, most significant ones are brand and model, age, horsepower and mileage. The fuel type used in the car as well as fuel consumption per mile highly affect price of a car due to a frequent change in the price of a fuel. Different features like exterior colour, door number, type of transmission, dimensions, safety, air condition, interior, whether it has navigation or not will also influence the car

price. In this paper, we applied different methods and techniques in order to achieve higher precision of the used car price prediction.

1.2 Conceptual Background of the Domain Problem

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. 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 have to make car price valuation model. This project contains two phase-

Data Collection Phase

You have to scrape at least 5000 used cars data. You can scrape more data as well, it's up to you.

more the data better the model

In this section You need to scrape the data of used cars from websites (Olx, cardekho, Cars24 etc.) You need web scraping for this. You have to fetch data for different locations. The number of columns for data doesn't have limit, it's up to you and your creativity. Generally, these columns are Brand, model, variant, manufacturing year, driven kilometres, fuel, number of owners, location and at last target variable Price of the car. This data is to give you a hint about important variables in used car model. You can make changes to it, you can add or you can remove some columns, it completely depends on the website from which you are fetching the data.

Model Building Phase

After collecting the data, you need to build a machine learning model. Before model building do all data pre-processing steps. Try different models with different hyper parameters and select the best model.

Follow the complete life cycle of data science. Include all the steps like.

1. Data Cleaning
2. Exploratory Data Analysis
3. Data Pre-processing
4. Model Building
5. Model Evaluation
6. Selecting the best model

1.3 Review of Literature

The price of a new car in the industry is fixed by the manufacturer with some additional costs incurred by the Government in the form of taxes. So, customers buying a brand-new vehicle may be confident of the money they make investments to be worth. But, due to the increased prices of new cars and the financial incapability of the customers to buy them, used Car sales are on a global increase. Even as new car sales have slowed down in the recent past, the pre-owned car market has continued to grow over the past year and is larger than the new car market now. Consider this: In 2018-19, while new car sales were recorded at 3.6 million units, 4 million second-hand cars were bought and sold, according to a recent report on India's pre-owned car market by Indian Bluebook, a used car pricing guide by Mahindra First Choice Wheels.
The growth rate of new car sales has slowed owing to a variety of reasons, including cyclical slowdown in auto sales in election years

and an overall consumption slowdown in the economy. New car sales grew 2.70% in 2018-19, the slowest growth rate for the industry in four years. In April, passenger vehicle sales saw a sharp decline compared with the same month last year, and domestic sales saw a contraction of 17.07%.

The Indian Bluebook report suggests that getting good value for money is among the top reasons why people prefer a pre-owned car over a new car.

Shubh Bansal, co-founder at Truebil, a pre-owned car marketplace, said this is not surprising given our economic development and increasing disposable income. In developed economies, the ratio of people buying a new car and a pre-owned car is 1:3, meaning out of four people who buy a car, only one buys a new car and three settle for a used car, he said, adding that the trend of higher sales of pre-owned cars will continue in India.

“The value of a car depreciates significantly in the first year itself, up to 50% for some models. From a buyer’s perspective, the aspiration of owning a car is getting fulfilled at a lower price point," he said.

However, the pre-owned car market is largely driven by supply and not demand, which is not the case with the new car market. The average ownership period of a car is coming down to about three to five years compared to eight to 10 years about a decade ago, said Bansal. “In many cases, a three-year-old vehicle is as good as a new one, which encourages the buyer to go for a used car," he said. Since demand is high, this is also perhaps a good time to sell a car.

Why Used Cars?

There are numerous aspects to take into consideration while purchasing a car – the main being should you buy a new or a used car. If you are trying to manage your finances wisely, opting for a pre-owned car would be a wise decision. Though the idea of purchasing a new car may sound

tempting, the quick rate of depreciation, higher price, and greater insurance, among others, do not work in the favour of new cars.

Given that there are numerous dealerships selling reliable and good condition pre-owned cars, you may opt for them. Following are five smart reasons to buy a second-hand car.

1. Value for money

Pre-owned cars come with a lower price tag and offer a much better value for the amount paid. You may compare numerous models from various used car dealerships, and select the model based on your needs. While doing so, compare the quoted rates and choose the dealer offering the best rate for the car. If you are making the purchase from a private seller, you may get a better price – given that there is no commission to be paid to middlemen. In order to finance your used car purchase, you may borrow a vehicle loan, and drive home the car of your dreams.

2. Slow rate of depreciation

A huge disadvantage of purchasing a new car is that its value depreciates the moment it is driven out of the showroom. The market value of the car decreases at a very rapid pace in the early years of the car. Hence, you may avoid this huge depreciation hit by opting for a pre-owned vehicle. Though a used car will depreciate, you will lose money less quickly.

3. Lower insurance and registration charges

The rate of insurance is generally based on the age of the car. Newer the vehicle, higher is the cost towards insurance and vice versa. The rate of insurance for pre-owned vehicles is therefore lower. Besides insurance, you also have to pay a lower amount towards registration fees. As the cost of registration is based on the transaction price of the car, buying a used car reduces the cost of registration.

4. Higher inflation

Given that the rate of inflation is increasing, consumers have to bear the brunt as the high cost is passed on to them. Automobile manufacturers quote a higher price for new models. In order to be protected against such rising prices, purchasing a used car is a safer option.

5. Lower loan amount to be borrowed

As used cars come with a lower price tag, the amount you may have to borrow will be lower. Many financial institutes offer used car loans with higher borrowing amount and attractive interest rates. Competitive interest rates indicate that a lower amount has to be repaid towards Equated Monthly Instalments (EMIs). You may choose to borrow a used car loan and buy a pre-owned car without any financial difficulties. Given that cars are now needed for everyday travel, investing in a pre-owned car is a wise decision.

Machine Learning for Car Price Prediction

When it comes to shopping for a used car, budget is one of the main constraints. Prices can differ for a variety of factors, and the customer experience is quite different than buying a new vehicle. The search process is the first step, and the buyer has access to a wealth of information and reviews on the internet. With all that said, there is virtually no way to know if an offer is **fair** or **overpriced**. In practice, *prior experience* and *extensive search* can help.

Therefore, to find the car price which would be best suited for the buyer, we are going to predict its cost with the help of Machine Learning algorithms [1] which are made available by the Python Environment. Our dataset comprises data related to different car brands with a set of parameters (Name, Model, Year, Mileage, Ratings, Reviews, Price). The primary purpose is to design a model for a collected dataset and predict the car price with better accuracy.

1.4 Motivation for the Problem Undertaken

The project was the first provided to me by Flip Robo Technologies as a part of the internship programme. The opportunity to deploy my skillset in collecting data from car selling websites and solving a real time problem has been the primary motivation.

This project includes the real time problem for price prediction of used cars and it is related to Preowned Cars Market, as I believe that with growing technologies and Idea can make a difference, there are so much in the Car market to explore and analyse and with Data Science the car world becomes more interesting. The objective of the project is to collect data of used cars which are posted for sales on car selling websites and prepare a model based on the collected dataset that predicts price for used cars based on different features.

Chapter 2

Analytical Problem Framing

2.1 Data Collection

We will collect the data from the website 'car.com' using web scrapper in python. Here we will use Beautiful Soup to collect the data and then the data cleaning will be performed. We will put the cleaned data in CSV format so that we can easily use this for further analysis and model building.

2.1 Mathematical / Analytical Modelling

Whenever we employ any ML algorithm, statistical models or feature pre-processing in background lot of mathematical framework work. In this project we have done lot of data pre-processing & ML model building.

Our objective is to predict the price for preowned cars which can be resolve by use of linear regression-based algorithm. In this project we are going to use different types of algorithms which uses their own mathematical equation on background. Initially data cleaning & pre-processing perform over data. Feature engineering is performed to remove unnecessary feature & for dimensionality reduction. In model building Final model is select based on evaluation benchmark among different models with different algorithms. Further Hyperparameter tuning performed to build more accurate model out of best model.

Followings are the algorithms used in this project

1. Linear Regression
2. Random Forest Regressor
3. Decision Tree Regressor
4. K-Neighbours Regressor
5. Extra Tree Regressor

2.2 Data Collection and Description

For collecting data, it is required to import all the necessary libraries.

```
from bs4 import BeautifulSoup
import requests
import pandas as pd
```

```
website = 'https://www.cars.com/shopping/results/?dealer_id=&keyword=&list_price_max=&list_price_min=&makes[]=&maximum_dista
```

```
response = requests.get(website)
```

```
response.status_code
```

```
200
```

```
soup = BeautifulSoup(response.content,'html.parser')
```

```
soup
```

The data is collected from car.com website which is one of the top cars selling site. The front display page of this site provides the information about preowned cars which are posted for sale. The information

contains the factors Car Model, make year, Dealer Name, Mileage or Kilometres driven, Condition of car, Customers Ratings, Total count of Reviews and Price. The company asked to collect the data for more than 5000 cars. Here we have collected the data for 6000 cars. To get this number we scrapped 300 pages of the website.

```
results = soup.find_all('div',{'class':'vehicle-card'})
```

```
len(results)
```

```
21
```

```
results[0].find('h2').get_text()
```

```
'2020 Mercedes-Benz GLS 450 Base 4MATIC'
```

```
results[0].find('div',{'':'mileage'}).get_text()
```

```
'27,617 mi.'
```

```
results[0].find('span',{'class':'sds-rating__count'}).get_text()
```

```
'4.4'
```

```
results[0].find('span',{'class':'sds-rating__link sds-button-link'}).get_text()
```

```
'(1,983 reviews)'
```

```
results[0].find('span',{'class':'primary-price'}).get_text()
```

```
'$70,925'
```

```
results[0].find('div',{'class':'dealer-name'}).get_text().strip()
```

```
"O'Meara Ford"
```

```
results[0].find('p',{'class':'stock-type'}).get_text()
```

```
'Used'
```

```
Condition =[]
Model = []
Dealer_name = []
Mileage = []
Ratings = []
Reviews = []
Price = []
```

```
for i in range(1,301):
    for result in results:
        # Condition
        try:
            Condition.append(result.find('p',{'class':'stock-type'}).get_text())
        except :
            Condition.append('N/A')

        # Model
        try :
            Model.append(result.find('h2').get_text())
        except :
            Model.append('N/A')

        # Dealer_name
        try:
            Dealer_name.append(result.find('div',{'class':'dealer-name'}).get_text().strip())
        except :
            Dealer_name.append('N/A')

        # Mileage
        try:
            Mileage.append(result.find('div',{'class':'mileage'}).get_text())
        except :
            Mileage.append('N/A')

        # Ratings
        try :
            Ratings.append(result.find('span',{'class':'sds-rating__count'}).get_text())
        except:
            Ratings.append('N/A')

        # Rating_counts
        try:
            Reviews.append(result.find('span',{'class':'sds-rating__link sds-button-link'}).get_text())
        except:
```

```
        # Price
        try:
            Price.append(result.find('span',{'class':'primary-price'}).get_text())
        except:
            Price.append('N/A')
```

```
cars_data = pd.DataFrame({'Condition':Condition,'Model Name':Model,'Dealer':Dealer_name,'Mileage':Mileage,'Ratings':Ratings,
```

```
cars_data
```

	Condition	Model Name	Dealer	Mileage	Ratings	Reviews	Price
0	Used	2018 Jeep Wrangler Unlimited Rubicon	Superior Dodge Chrysler Jeep Ram Of Siloam Spr...	54,107 mi.	2.4	(53 reviews)	$42,995
1	Volkswagen Certified	2022 Volkswagen Atlas 3.6L SE w/Technology	Don Thornton Volkswagen of Tulsa	10,301 mi.	3.8	(281 reviews)	$46,355
2	Used	2022 Buick Encore Preferred	Impex Chevrolet GMC Buick	3 mi.	N/A	(6 reviews)	$23,950
3	Used	2022 Volvo XC90 T6 Inscription 6 Passenger	Volvo Cars Mall of Georgia	9,109 mi.	4.9	(2,209 reviews)	$71,482
4	Used	2020 Chevrolet Bolt EV Premier	RideShift - Richmond	15,642 mi.	N/A	N/A	$32,699
...	...	...	...	...	...	...	...
5995	Used	2018 Dodge Charger R/T	Golden Motors	81,886 mi.	N/A	(4 reviews)	$25,950
5996	Used	2020 INFINITI QX60 Pure	Pischke Motors of La Crosse	27,804 mi.	3.4	(60 reviews)	$32,499
5997	Used	2018 Chevrolet Traverse LS	Modern Classic Motors	67,277 mi.	4.0	(351 reviews)	$23,373
5998	Used	2021 GMC Yukon Denali	Tadd Jenkins Ford	19,080 mi.	N/A	(0 reviews)	$79,995
5999	Used	2022 Kia Carnival SX	Magic City Ford Roanoke	23,381 mi.	2.5	(90 reviews)	$49,600

6000 rows × 7 columns

Chapter 3

Data Pre-processing

3.1 Data Cleaning

```
cars_data['Mileage']=cars_data['Mileage'].apply(lambda x:x.strip(' mi.'))
cars_data['Reviews'] = cars_data['Reviews'].apply(lambda x:x.strip('reviews)').strip('('))
cars_data['Price']=cars_data['Price'].apply(lambda x:x.strip('$'))
```

```
cars_data['make_year'] = [x[:4] for x in cars_data['Model Name']]
cars_data['car_model'] = [x[4:] for x in cars_data['Model Name']]
```

```
cars_data.drop(columns=['Model Name'], axis=1, inplace=True)
```

And finally got the dataset as below

	Condition	Dealer	Mileage	Ratings	Reviews	Price	make_year	car_model
0	Used	Superior Dodge Chrysler Jeep Ram Of Siloam Spr...	54,107	2.4	53	42,995	2018	Jeep Wrangler Unlimited Rubicon
1	Volkswagen Certified	Don Thornton Volkswagen of Tulsa	10,301	3.8	281	46,355	2022	Volkswagen Atlas 3.6L SE w/Technology
2	Used	Impex Chevrolet GMC Buick	3	N/A	6	23,950	2022	Buick Encore Preferred
3	Used	Volvo Cars Mall of Georgia	9,109	4.9	2,209	71,482	2022	Volvo XC90 T6 Inscription 6 Passenger
4	Used	RideShift - Richmond	15,642	N/A	N/A	32,699	2020	Chevrolet Bolt EV Premier
...	...	...	...	...	...	...	...	...
5995	Used	Golden Motors	81,886	N/A	4	25,950	2018	Dodge Charger R/T
5996	Used	Pischke Motors of La Crosse	27,804	3.4	60	32,499	2020	INFINITI QX60 Pure
5997	Used	Modern Classic Motors	67,277	4.0	351	23,373	2018	Chevrolet Traverse LS
5998	Used	Tadd Jenkins Ford	19,080	N/A	0	79,995	2021	GMC Yukon Denali
5999	Used	Magic City Ford Roanoke	23,381	2.5	90	49,600	2022	Kia Carnival SX

6000 rows × 8 columns

Importing Libraries

```
import pandas as pd # for data wrangling purpose
import numpy as np # Basic computation library
import seaborn as sns # For Visualization
import matplotlib.pyplot as plt # ploting package
%matplotlib inline
import warnings # Filtering warnings
warnings.filterwarnings('ignore')
```

Checking for Shape of Dataset

```
print('No. of Rows :',cars_data.shape[0])
print('No. of Columns :',cars_data.shape[1])
pd.set_option('display.max_columns',None)
cars_data.head()
```

```
No. of Rows : 6000
No. of Columns : 8
```

Checking for datatypes

```
cars_data.columns.to_series().groupby(cars_data.dtypes).groups
```

```
{object: ['Condition', 'Dealer', 'Mileage', 'Ratings', 'Reviews', 'Price', 'make_year', 'car_model']}
```

Since the datatypes of features Reviews, Mileage and Price are object type here we have to convert them into float type.

```
cars_data['Reviews']=cars_data['Reviews'].apply(lambda x:x.strip(','))
cars_data['Mileage']=cars_data['Mileage'].apply(lambda x:x.strip(','))
cars_data['Price']=cars_data['Price'].apply(lambda x:x.strip(','))
```

```
# removing the ',' from 'Reviews','Price' and 'Mileage'
cars_data['Reviews']=cars_data['Reviews'].str.replace(',','')
cars_data['Mileage']=cars_data['Mileage'].str.replace(',','')
cars_data['Price']=cars_data['Price'].str.replace(',','')
# Converting datatype into float
cars_data['Reviews'] = cars_data['Reviews'].astype(float)
cars_data['Mileage'] = cars_data['Mileage'].astype(float)
cars_data['Price'] = cars_data['Price'].astype(float)
```

```
cars_data.head()
```

	Condition	Dealer	Mileage	Ratings	Reviews	Price	make_year	car_model
0	Used	O'Meara Ford	27617.0	4.4	1983.0	70925.0	2020	Mercedes-Benz GLS 450 Base 4MATIC
1	Used	Suntrup Hyundai South	3244.0	4.8	3368.0	34941.0	2022	Hyundai Santa Fe XRT
2	Used	Mercedes-Benz of Chantilly	0.0	4.6	921.0	53660.0	2022	Mercedes-Benz GLC 300 Base 4MATIC
3	Used	Smith Motor Sales of Haverhill, Inc.	7658.0	4.9	1516.0	47969.0	2021	Mercedes-Benz GLC 300 Base 4MATIC
4	Used	Giles Nissan Lafayette	42469.0	3.2	24.0	68980.0	2019	Lexus LX 570 570

Checked the datatype again after changing it to float.

```
cars_data.columns.to_series().groupby(cars_data.dtypes).groups
```

```
{float64: ['Mileage', 'Reviews', 'Price'], object: ['Condition', 'Dealer', 'Ratings', 'make_year', 'car_model']}
```

3.2 Data Pre-processing

The dataset is large and it may contain some data error. In order to reach clean, error free data some pre-processing is done on data. At first integrity check is perform on data for presence of missing values, whitespaces. After that statistical matrix is plotted using df.describe () command to gain more insight about data.

- Data integrity check –

```
cars_data.isin(['NA','N/A','-',' ','?',' ?']).sum().any()
```

```
True
```

```
cars_data.isin(['N/A']).sum().any()
```

```
True
```

It shows that some ‘N/A’ are present in our dataset which we will replace by np. NaN.

```
cars_data=cars_data.replace('N/A',np.NaN)
```

- Missing value check

```
missing_values = cars_data.isnull().sum().sort_values(ascending = False)
percentage_missing_values =(missing_values/len(cars_data))*100
print(pd.concat([missing_values, percentage_missing_values], axis =1, keys =['Missing Values', '% Missing data']))
```

The result shoes that the features Ratings and Reviews have missing values.

- Missing value imputation

  Since the feature ‘Ratings’ is a categorical type feature we used mode imputation method for it and the feature Reviews is a numerical feature we used the mean imputation method.

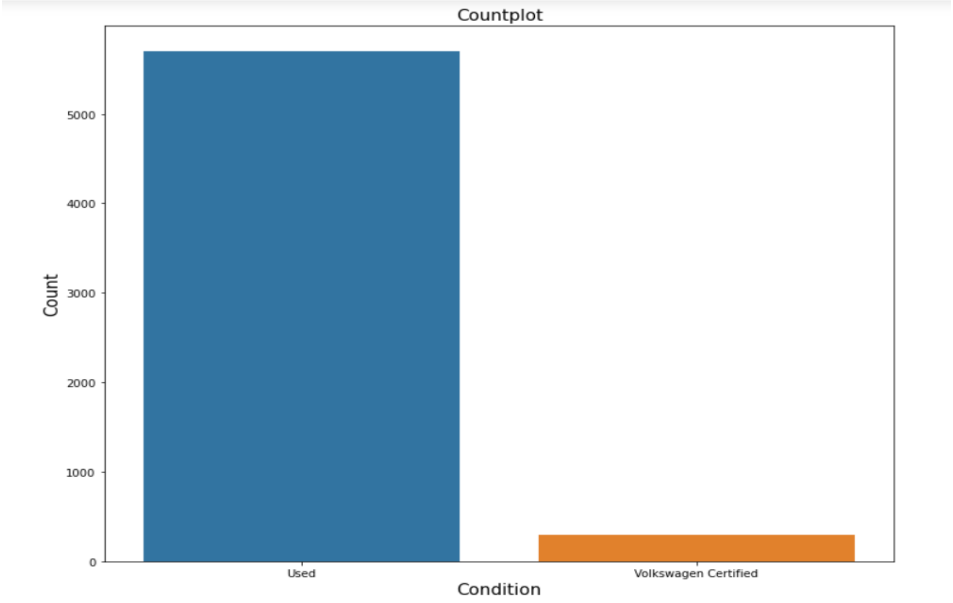
```
# Since 'Ratings' is a object type feature we will make imputation using mode
cars_data['Ratings'].fillna(cars_data['Ratings'].mode()[0],inplace=True)
cars_data['Reviews'].fillna(cars_data['Reviews'].mean(),inplace=True)

```

3.3 Data Visualization

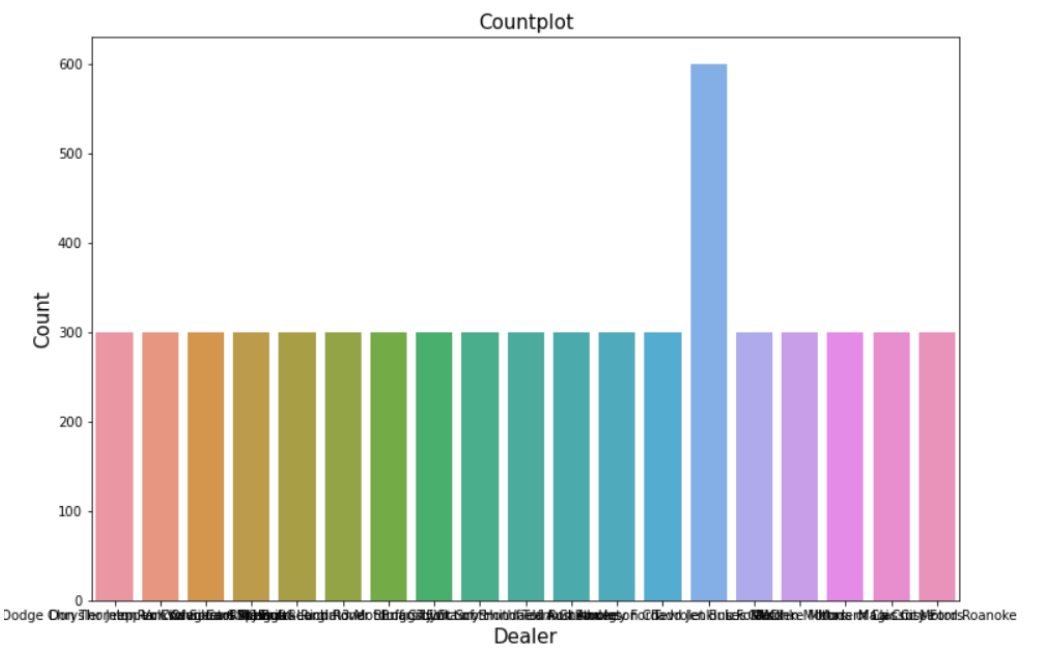
Data Visualization is the process of analysing data in the form of graphs or maps, making it a lot easier to understand the trends or patterns in the data.

1. Condition



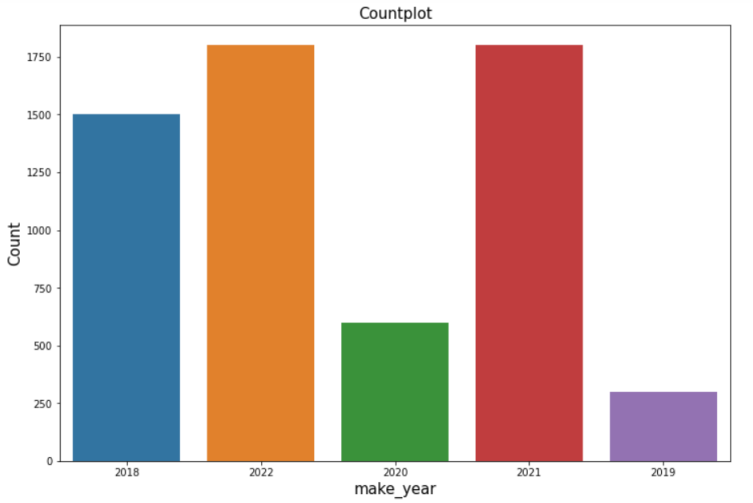
There are 5700 used cars and only 300 cars are RAM Certified.

2. Dealer



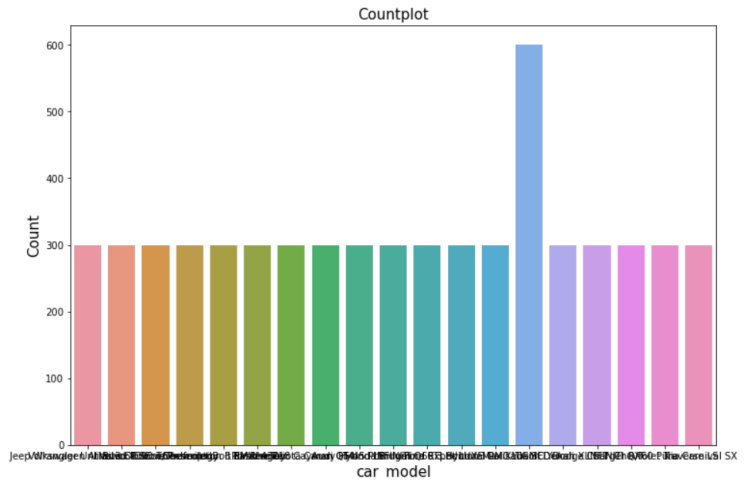
There is total 19 dealers out of which 18 are having equal numbers of vehicles i.e., 300 and only one dealer has 600 cars.

3. make_year

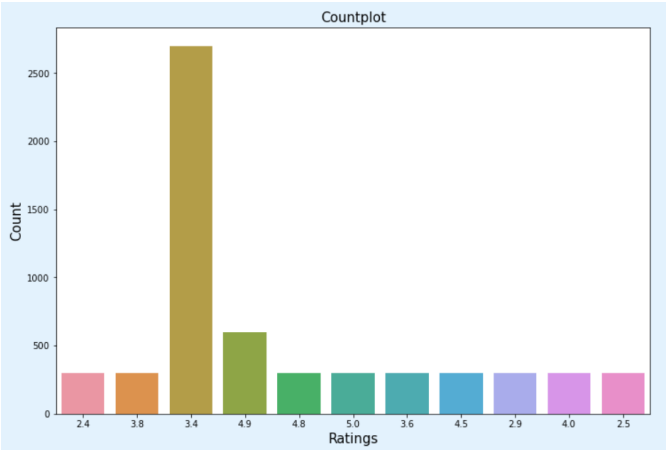


Most of the cars are of make year 2021 followed by 2022.

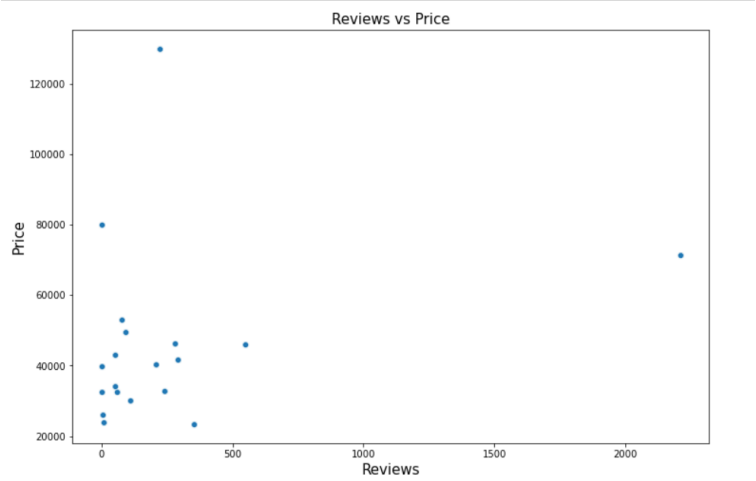
4. car_model



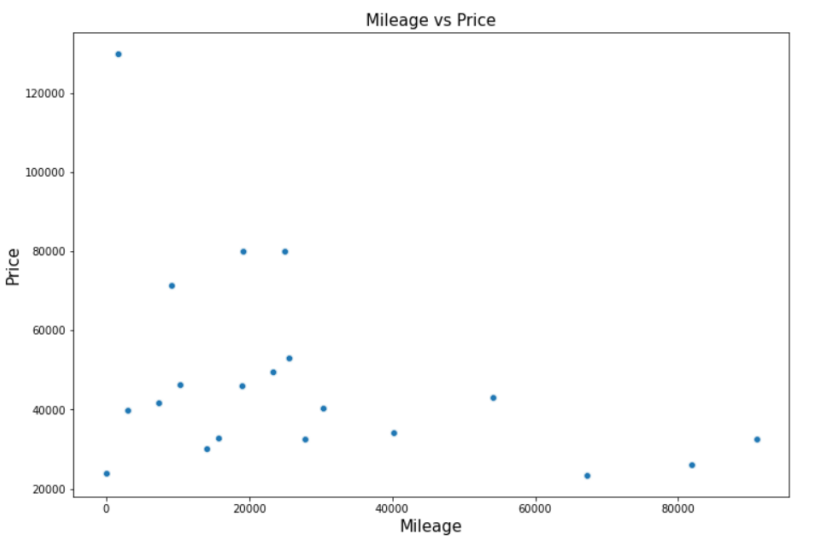
5. Ratings



6. Reviews



7. Mileage



The prices are more for the cars which have less km-driven.

3.4 Outliers

Label Encoding

```
# Using Label Encoder on categorical variable
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for i in cat:
    cars_data[i] = le.fit_transform(cars_data[i])
cars_data.head()
```

Outliers Detection

```
plt.figure(figsize=(20,10),facecolor='white')
plotnumber=1

for column in num:
    if plotnumber<=3:
        ax=plt.subplot(1,3,plotnumber)
        sns.boxplot(cars_data[column],color='r')
        plt.xlabel(column,fontsize=20)
    plotnumber+=1
plt.show()
```

Outliers Removal

We got observation from boxplot that outliers do not exist in lower bound but outliers exist in upper bound of features. Based on this observation we decided to employ quantile-based flooring- capping method. Flooring is performed at 0th percentile for lower bound and capping perform at 95th percentile for upper bound.

```
Q1 = cars_data.quantile(0)
Q3= cars_data.quantile(0.95)
IQR = Q3 - Q1
print(IQR)
```

```
df_new = cars_data[~((cars_data < (Q1 - 1.5 * IQR)) |(cars_data > (Q3 + 1.5 * IQR))).any(axis=1)]
print(df_new.shape)
print ("Percentage of data loss post outlier removal: ", (cars_data.shape[0]-df_new.shape[0])/cars_data.shape[0]*100)
```

```
(5400, 8)
Percentage of data loss post outlier removal:  10.0
```

Here we lost 10% data on removing outliers which we can afford.

Skewness

Skewness is checked for all the features by plotting distribution plots for the features

```
plt.figure(figsize=(18,12),facecolor='white')
sns.set_palette('plasma')
plotnum=1
for col in num:
    if plotnum<=3:
        plt.subplot(1,3,plotnum)
        sns.distplot(df_new[col])
        plt.xlabel(col,fontsize=20)
    plotnum+=1
plt.show()
```

Considerable amount of skewness found in most features by skew () function. Power transformer from sklearn. preprocessing library used to transform skewness in features.

The features with skewness are Reviews and Mileage
To remove the skewness from the above skewed features we used the Power transform method.

```
skew=['Reviews','Mileage']
from sklearn.preprocessing import PowerTransformer
scaler = PowerTransformer(method='yeo-johnson')

# Transfroming skew data
df_new[skew] = scaler.fit_transform(df_new[skew].values)
```

```
df_new[skew].skew()
```

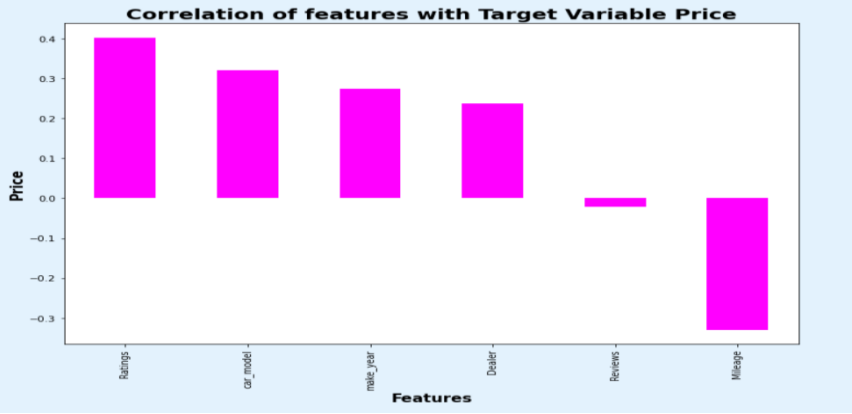
```
Reviews    -0.220411
Mileage    -0.273702
dtype: float64
```

The skewness is reduced within permissible limit .

Chapter 4

Feature Scaling

4.1 Correlation



From the above correlation graph it is clear that the only feature Mileage has negative correlation with Price. The rest are positively correlated with price.

The feature Ratings has highest positive correlation.

The lowest correlated feature is Reviews.

4.2 Splitting Data into target and features

```
Y = df_new['Price']
X = df_new.drop(['Price'], axis =1)
```

4.3 Feature Scaling

```
from sklearn.preprocessing import StandardScaler
```

```
sc= StandardScaler()
X_scale = sc.fit_transform(X)
```

4.4 multi-Collinearity

To check the multicollinearity, I used the VIF method.

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
vif = pd.DataFrame()
vif["VIF values"] = [variance_inflation_factor(X_scale,i) for i in range(len(X.columns))]
vif["Features"] = X.columns
vif
```

	VIF values	Features
0	1.159718	Dealer
1	3.771350	Mileage
2	1.545193	Ratings
3	1.225512	Reviews
4	3.786467	make_year
5	1.050478	car_model

All the Independent features VIF are within the permissible limit of 10 which means there is no multicollinearity present.

Chapter 5

Model Development and Evaluation

Our objective is to predict the price of cars. This becomes linear regression problem which can be solved using various regressor algorithms. In order to gain high accuracy of model we will train model with different regressor models and select final model among them. To enhance performance of best model will employ hyper parameter tuning over it. At end we will save our final model using joblib.

5.1 Testing of Identified Approaches (Algorithms)

The different classification algorithm used in this project to build ML model are as below:

- Linear Regression
- Random Forest Regressor
- Decision Tree Regressor
- K-Neighbours Regressor
- AdaBoost Regressor

5.2 Splitting Data into Training & test data sets

```
1 X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state=42, test_size=.33)
2 print('Training feature matrix size:',X_train.shape)
3 print('Training target vector size:',Y_train.shape)
4 print('Test feature matrix size:',X_test.shape)
5 print('Test target vector size:',Y_test.shape)
6
```

```
Training feature matrix size: (3618, 6)
Training target vector size: (3618,)
Test feature matrix size: (1782, 6)
Test target vector size: (1782,)
```

5.3 Finding Best Random State

```
maxR2_score=0
maxRS=0
for i in range(1,500):
    X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state=i, test_size=.33)
    lin_reg=LinearRegression()
    lin_reg.fit(X_train,Y_train)
    y_pred=lin_reg.predict(X_test)
    R2=r2_score(Y_test,y_pred)
    if R2>maxR2_score:
        maxR2_score=R2
        maxRS=i
print('Best R2 Score is', maxR2_score ,'on Random_state', maxRS)
```

```
Best R2 Score is 0.5574936301341684 on Random_state 36
```

Here, we got 36 as best random state and the best accuracy at random state 244 is 0.557439 with the linear regression algorithm.

5.4 Run & Evaluate the selected algorithms

1. Linear Regression

```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state=36, test_size=.33)
lr=LinearRegression()
lr.fit(X_train,Y_train)
lr.score(X_train,Y_train)
y_pred_lr=lr.predict(X_test)
print('Mean absolute error :', mean_absolute_error(Y_test,y_pred_lr))
print('Mean squared error :', mean_squared_error(Y_test,y_pred_lr))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(Y_test,y_pred_lr)))
print('R2_Score :' ,r2_score(Y_test,y_pred_lr))
print('CV_Score_lr : ',cross_val_score(lr, X_scale, Y, cv =5).mean())
```

```
Mean absolute error : 13581.128659656477
Mean squared error : 318408546.5524712
Root Mean Squared Error: 17844.00589981048
R2_Score : 0.5574936301341684
CV_Score_lr :  0.5193445661086012
```

2. Random Forest Regressor

```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state=36, test_size=.33)
rf=RandomForestRegressor()
rf.fit(X_train,Y_train)
rf.score(X_train,Y_train)
y_pred_rf=rf.predict(X_test)
print('Mean absolute error :', mean_absolute_error(Y_test,y_pred_rf))
print('Mean squared error :', mean_squared_error(Y_test,y_pred_rf))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(Y_test,y_pred_rf)))
print('R2_Score :' ,r2_score(Y_test,y_pred_rf))
print('CV_Score_rf : ',cross_val_score(rf, X_scale, Y, cv =5).mean())
```

```
Mean absolute error : 0.0
Mean squared error : 0.0
Root Mean Squared Error: 0.0
R2_Score : 1.0
CV_Score_rf :  1.0
```

3. Decision Tree Regressor

```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state=36, test_size=.33)
dt=DecisionTreeRegressor()
dt.fit(X_train,Y_train)
dt.score(X_train,Y_train)
y_pred_dt=dt.predict(X_test)
print('Mean absolute error :', mean_absolute_error(Y_test,y_pred_dt))
print('Mean squared error :', mean_squared_error(Y_test,y_pred_dt))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(Y_test,y_pred_dt)))
print('R2_Score :' ,r2_score(Y_test,y_pred_dt))
print('CV_Score_dt : ',cross_val_score(dt, X_scale, Y, cv =5).mean())
```

```
Mean absolute error : 0.0
Mean squared error : 0.0
Root Mean Squared Error: 0.0
R2_Score : 1.0
CV_Score_dt :  1.0
```

4. K-Neighbours Regressor

```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state=36, test_size=.33)
knn=KNeighborsRegressor()
knn.fit(X_train,Y_train)
knn.score(X_train,Y_train)
y_pred_knn=knn.predict(X_test)
print('Mean absolute error :', mean_absolute_error(Y_test,y_pred_knn))
print('Mean squared error :', mean_squared_error(Y_test,y_pred_knn))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(Y_test,y_pred_knn)))
print('R2_Score :' ,r2_score(Y_test,y_pred_knn))
print('CV_Score_dt : ',cross_val_score(knn, X_scale, Y, cv =5).mean())
```

```
Mean absolute error : 0.0
Mean squared error : 0.0
Root Mean Squared Error: 0.0
R2_Score : 1.0
CV_Score_dt :  1.0
```

5. AdaBoost Regressor

```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state=36, test_size=.33)
ab=AdaBoostRegressor()
ab.fit(X_train,Y_train)
ab.score(X_train,Y_train)
y_pred_ab=ab.predict(X_test)
print('Mean absolute error :', mean_absolute_error(Y_test,y_pred_ab))
print('Mean squared error :', mean_squared_error(Y_test,y_pred_ab))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(Y_test,y_pred_ab)))
print('R2_Score :' ,r2_score(Y_test,y_pred_ab))
print('CV_Score_ab : ',cross_val_score(ab, X_scale, Y, cv =5).mean())
```

```
Mean absolute error : 3620.4601175906096
Mean squared error : 25944161.54890307
Root Mean Squared Error: 5093.541160028361
R2_Score : 0.9639442569286503
CV_Score_ab :  0.9882319421224525
```

Accuracy Scores of all used models

Algorithm	Accuracy Score	CV Score
Linear Regression	0.557493	0.519344
Random Forest Regressor	1.0	1.0
Decision Tree Regressor	1.0	1.0
K-Neighbours Regressor	1.0	1.0
AdaBoost Regressor	0.963944	0.988231

Among all the algorithms used Random Forest Regressor, Decision Tree Regressor and K-Neighbours Regressor gives us better R2 Score with zero Root Mean Squared Error.

Out of these 3 we will select Random Forest Regressor as final model and perform Hyperparameter optimization for this model

5.5Hyperparameter Optimization

```
from sklearn.model_selection import GridSearchCV
```

```
n_estimators = [2,5,7,9,15]
criterion_list = ['gini','entropy']
max_features = ["auto",'log']
max_depth = [1,4]
min_samples_split = [1,5]
min_samples_leaf = [2,6]
bootstrap = ['true','false']
```

```
param_grid ={'n_estimators':n_estimators,
    'max_features': max_features,
    'max_depth': max_depth,
    'min_samples_split': min_samples_split,
     'min_samples_leaf': min_samples_leaf,
     'bootstrap': bootstrap}
```

```
clf = GridSearchCV(rf,param_grid = param_grid,cv = 10,verbose = True,n_jobs = -1)
```

```
best_clf = clf.fit(X,Y)
```

```
Fitting 10 folds for each of 160 candidates, totalling 1600 fits
```

```
best_clf.best_estimator_
```

```
RandomForestRegressor(bootstrap='true', max_depth=4, max_features='auto',
                      min_samples_leaf=2, min_samples_split=5, n_estimators=5)
```

```
param_score_rf = round(best_clf.score(X,Y),3)
param_score_rf
```

```
0.982
```

5.6 Final Model Saving

```
import pickle
```

```
filename = "CarPricePrediction"
pickle.dump(rf,open(filename,'wb'))
```

5.7 Conclusion

```
loaded_model = pickle.load(open(filename,'rb'))
result = loaded_model.score(X_test,Y_test)
print(result*100)
```

```
100.0
```

5.8 Predictions on Test data

```
conclusion = pd.DataFrame([loaded_model.predict(X_test)[:],y_pred_rf[:]],index=["Predicted","Original"])
conclusion
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Predicted	49600.0	79995.0	23950.0	40500.0	45988.0	23950.0	29995.0	23950.0	32499.0	79995.0	41696.0	40500.0	52973.0	40500.0	40500.0	79995.0	32499.0
Original	49600.0	79995.0	23950.0	40500.0	45988.0	23950.0	29995.0	23950.0	32499.0	79995.0	41696.0	40500.0	52973.0	40500.0	40500.0	79995.0	32499.0

Chapter 6

Conclusion

- Random Forest Classifier Hyper parameter tuned gives maximum accuracy score of 0.982.
- The price is high for the cars which has minimum mileage means which has minimum running.