USED CAR PRICE PREDICTION PROJECT REPORT

SUBMITTED BY:

NASHEED ASAD

ACKNOWLEDGMENT

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to Flip Robo Technologies for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I want to thank my SME Mrs. Khusboo Garg helping us to solve the problem and addressing out our Query in right time.

I would like to express my gratitude towards my parents, my sister, Ummehani, who has been an integral part of many of my projects, & members of Flip Robo for their kind co-operation and encouragement which help me in completion of this project.

I would like to express my special gratitude and thanks to industry persons for giving me such attention and time.

The data was collected from www.cars24.com

INTRODUCTION

Business Problem Framing

The project's goal was to create a model that could forecast the price of used automobiles based on the available independent factors. This model may then be utilized by management to understand how the prices fluctuate in relation to the factors. As a result, they may alter the firm's strategy and focus on regions that will provide significant profits. Furthermore, the model will help management understand the price characteristics of a new market post covid.

Conceptual Background of the Domain Problem

We have witnessed several changes in the automotive market as a result of the Covid-19's influence on the market. Some automobiles are in high demand, thus they are more expensive, while others are not, so they are less expensive. One of our clients does business with small traders that sell secondhand cars. With the market changing as a result of the Covid-19 effect, our customer is having issues with their prior automobile price valuation machine learning models. As a result, they are seeking for new machine learning models based on new data. We must create an automobile price valuation model.

Review of literature

The cost of a used automobile is determined by a variety of factors. From the number of kilometers travelled to the condition of the vehicle. Old automobiles are a significant business since many people prefer to buy used cars to save money.

Motivation for the problem undertaken

To comprehend real-world challenges where Machine Learning and Data Analysis may be used to assist companies in many fields in making better judgments that will allow them to benefit or avoid losses that would otherwise be achievable without the study of data.

ANALYTICAL PROBLEM FRAMING

Mathematical/Analytical Modeling of the problem

This is a Regression problem, and the final aim is to estimate used vehicle prices based on data. I gathered the data from cars24.com and will build the model using the information I gathered.

Data sources and their formats

The dataset has 8116 rows and 11 columns. Using this dataset, we will train the Machine Learning models on 75% of the data and test the models on 25% of the data.

The data was obtained in csv format from cars24.com. The data is described further down.

Brand	Brand of the car
Model	Model of the car
Variant	Variant of the car
Make Year	Manufacturing Year of the car
Fuel Type	Type of fuel used in the car
Transmission	Type of transmission of the car
Kilometers Driven	Number of KMs driven by the car
Owner	Number of previous owners
Location	Cars location

Price	Price of the car
Car Age	Age of the car

Data Preprocessing Done

Importing the required libraries and viewing a preview of the data.

```
#Importing Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

#Preprocessing,Standardizing
from sklearn.preprocessing import StandardScaler

#For Multicollinearity
from statsmodels.stats.outliers_influence import variance_inflation_factor

#Models
from sklearn.model_selection import train_test_split,GridSearchCV,RandomizedSearchCV
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.linear_model import LinearRegression

#Metrics
from sklearn.metrics import r2_score

import warnings
warnings.filterwarnings('ignore')
```

```
df=pd.read_excel('cars.xlsx')
df.head()
```

	Unnamed: 0	Brand	Model	Variant	Make Year	Fuel Type	Transmission	Kilometers Driven	Owner	Location	Price
0	0	maruti	Maruti Wagon R 1.0	VXI	2011	Petrol	Manual	32,725	1	bengaluru	3,39,299
1	1	maruti	Maruti Alto 800	LXI	2016	Petrol	Manual	16,134	1	bengaluru	3,19,099
2	2	maruti	Maruti Celerio	VXI AMT	2014	Petrol	Automatic	13,928	1	bengaluru	4,28,999
3	3	maruti	Maruti Ritz	VXI BS IV	2013	Petrol	Manual	41,507	1	bengaluru	4,18,899
4	4	maruti	Maruti Alto 800	LXI	2017	Petrol	Manual	10,742	2	bengaluru	3,51,799

```
df.shape
```

```
(8116, 11)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8116 entries, 0 to 8115
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Unnamed: 0         8116 non-null   int64
 1   Brand              8116 non-null   object
 2   Model              8116 non-null   object
 3   Variant            8116 non-null   object
 4   Make Year          8116 non-null   int64
 5   Fuel Type          8116 non-null   object
 6   Transmission       7836 non-null   object
 7   Kilometers Driven  8116 non-null   object
 8   Owner              8116 non-null   int64
 9   Location           8116 non-null   object
 10  Price              8116 non-null   object
dtypes: int64(3), object(8)
memory usage: 697.6+ KB
```

We can see there are 8116 rows and 11 columns. There are null values in the 'Transmission' column. We have int and object data types in our dataset. Price is our target variable. We have to convert 'Kilometers Driven' and 'Price' variables into int data types. And we will drop 'Unnamed: 0' since it gives no value to our dataset. We will do some data cleaning on our 'Fuel type', 'Model' and 'Variant' variables too. We will add a new variable 'Car Age' and delete the 'Make Year' variable.

```
df['Transmission']=df['Transmission'].fillna(df['Transmission'].mode()[0])
```

```
df['Kilometers Driven']=df['Kilometers Driven'].str.replace(',','')
df['Price']=df['Price'].str.replace(',','')
```

```
df.rename(columns={'Kilometers Driven':'KilometersDriven'},inplace=True)
```

```
df.drop(columns=['Unnamed: 0'],axis=1,inplace=True)
```

```
df['KilometersDriven']=df['KilometersDriven'].astype(int)
df['Price']=df['Price'].astype(int)
```

```
df['Fuel Type'].value_counts()
```

```
Petrol          5313
Diesel          2584
Petrol + CNG     211
Petrol + LPG       6
Electric           2
```

```
df['Fuel Type']=df['Fuel Type'].replace(['Petrol + CNG'], 'CNG')
df['Fuel Type']=df['Fuel Type'].replace(['Petrol + LPG'], 'LPG')
```

```
df['Model']=df['Model'].str.split(" ").str.slice(1,3).str.join(' ')
df['Variant']=df['Variant'].str.split(" ").str.slice(0,2).str.join(' ')
```

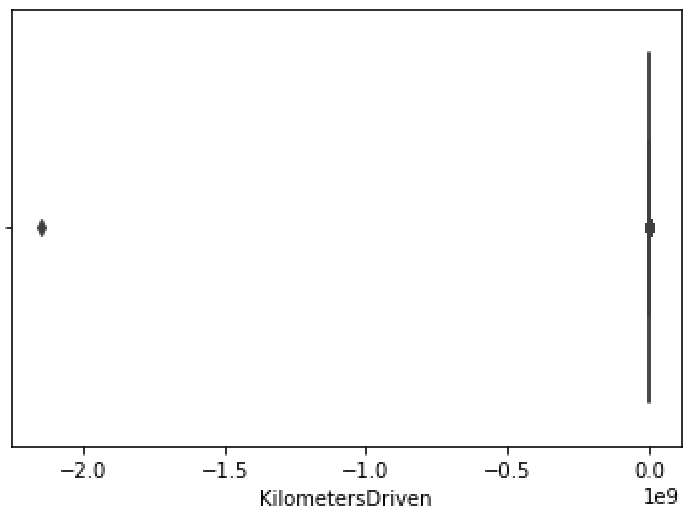
```
df['Current Year']=2021
df['Car Age']=df['Current Year']-df['Make Year']
df.drop(['Make Year'],axis=1,inplace=True)
df.drop(['Current Year'],axis=1,inplace=True)
df.head()
```

	Brand	Model	Variant	Fuel Type	Transmission	KilometersDriven	Owner	Location	Price	Car Age
0	maruti	Wagon R	VXI	Petrol	Manual	32725	1	bengaluru	339299	10
1	maruti	Alto 800	LXI	Petrol	Manual	16134	1	bengaluru	319099	5
2	maruti	Celerio	VXI AMT	Petrol	Automatic	13928	1	bengaluru	428999	7
3	maruti	Ritz	VXI BS	Petrol	Manual	41507	1	bengaluru	418899	8
4	maruti	Alto 800	LXI	Petrol	Manual	10742	2	bengaluru	351799	4

```
from sklearn.preprocessing import LabelEncoder

lab_enc=LabelEncoder()

df1=lab_enc.fit_transform(df['Brand'])
df2=lab_enc.fit_transform(df['Model'])
df3=lab_enc.fit_transform(df['Variant'])
df5=lab_enc.fit_transform(df['Fuel Type'])
df6=lab_enc.fit_transform(df['Transmission'])
df7=lab_enc.fit_transform(df['Location'])


df['Brand']=df1
df['Model']=df2
df['Variant']=df3
df['Fuel Type']=df5
df['Transmission']=df6
df['Location']=df7
```

```python
sns.boxplot(df['KilometersDriven'])
```

```
<AxesSubplot:xlabel='KilometersDriven'>
```




```python
#Find the IQR (inter quantile range) to identify outliers

q1=df.quantile(0.25) #1st quantile

q3=df.quantile(0.75) #3rd quantile

#IQR
iqr=q3-q1
iqr
```

```python
index=np.where(df['KilometersDriven']>(q3.KilometersDriven)+(1.5*iqr.KilometersDriven))
df=df.drop(df.index[index])
print('Shape:',df.shape)
df.reset_index()
```

```
Shape: (7897, 10)
```

```python
index=np.where(df['KilometersDriven']<(q1.KilometersDriven)-(1.5*iqr.KilometersDriven))
df=df.drop(df.index[index])
print('Shape:',df.shape)
df.reset_index()
```

```
Shape: (7896, 10)
```

```python
df['KilometersDriven']=np.sqrt(df['KilometersDriven'])
```

```python
df.skew()
```

```
Brand               0.422427
Model               0.168025
Variant            -0.363732
Fuel Type          -0.759906
Transmission       -2.261232
KilometersDriven    0.042875
Owner               2.212310
Location           -0.398069
Price               2.766919
Car Age             0.616359
dtype: float64
```

We have converted all the categorical data into numeric variables. Then, we have checked for outliers and skewness which is only present in KilometersDriven since it is the only numerical variable in our dataset.

Hardware and Software Requirements

The hardware utilized for this project is a laptop with high-end specifications and a steady internet connection. When it came to the software, I utilized anaconda navigator and Jupyter notebook to conduct my Python programming and analysis.

Microsoft Excel is required to use an excel file. In Jupyter notebook, I utilized several Python libraries to complete this project, which I have listed below with appropriate substantiation:

1. Pandas - It is a library that is used to read data, visualize it, and analyze it.
2. NumPy- utilized for dealing with arrays and different mathematical methods.

3. Seaborn- a visualization tool for plotting many sorts of plots.
4. Matplotlib- It provides an object-oriented API for embedding plots into applications.

MODELS DEVELOPMENT AND EVALUATION

Testing of Identified Approaches (Algorithms)

I have used 5 algorithms to train and test my dataset. They are:

- Decision Tree Regressor
- Random Forest Regressor
- K-Neighbors Regressor
- Ada Boost Regressor
- Gradient Boosting Regressor

Run and evaluate selected models

```
X=df.drop(columns=['Price'],axis=1)
y=df['Price']
```

```
#Standardizing

scaler=StandardScaler()
X_scaler=scaler.fit_transform(X)

#Checking multicollinearity by vif

vif=pd.DataFrame()
vif['score']=[variance_inflation_factor(X_scaler,i) for i in range(X_scaler.shape[1])]
vif['Features']=X.columns

vif
```

	score	Features
0	1.112037	Brand
1	1.112863	Model
2	1.126585	Variant
3	1.344592	Fuel Type
4	1.008990	Transmission
5	1.892226	KilometersDriven
6	1.100555	Owner
7	1.010398	Location
8	1.623047	Car Age

```
maxAccu=0
maxRs=0
for i in range(1,200):
    X_train,x_test,Y_train,y_test=train_test_split(X_scaler,y,test_size=0.25,random_state=i)
    mod=DecisionTreeRegressor()
    mod.fit(X_train,Y_train)
    pred=mod.predict(x_test)
    acc=r2_score(y_test,pred)
    if acc>maxAccu:
        maxAccu=acc
        maxRs=i
print("Best accuracy is:",maxAccu,"on Random State",maxRs)
```

```
Best accuracy is: 0.956310827593007 on Random State 152
```

```
X_train,x_test,Y_train,y_test=train_test_split(X_scaler,y,test_size=0.25,random_state=152)
```

```
DTR=DecisionTreeRegressor()
DTR.fit(X_train,Y_train)
pred=DTR.predict(x_test)
print(r2_score(y_test,pred))
```

```
0.9557916219536722
```

```
RFR=RandomForestRegressor()
RFR.fit(X_train,Y_train)
pred=RFR.predict(x_test)
print(r2_score(y_test,pred))
```

```
0.9496894340916135
```

```
from sklearn.neighbors import KNeighborsRegressor
knn=KNeighborsRegressor()
knn.fit(X_train,Y_train)
pred=knn.predict(x_test)
print(r2_score(y_test,pred))
```

```
0.664721177973995
```

```
from sklearn.ensemble import AdaBoostRegressor
ada=AdaBoostRegressor()
ada.fit(X_train,Y_train)
pred=ada.predict(x_test)
print(r2_score(y_test,pred))
```

```
0.2082364754711412
```

```
from sklearn.ensemble import GradientBoostingRegressor
gbc=GradientBoostingRegressor()
gbc.fit(X_train,Y_train)
pred=gbc.predict(x_test)
print(r2_score(y_test,pred))
```

```
0.8596503597902009
```

We have separated the independent and dependent variables and standardized our dataset to look for multicollinearity. We don't find any multicollinearity in our dataset. We proceed further to check the best random state for our dataset and use that random state to find the accuracy using all 5 algorithms mentioned earlier.

Key Metrics for success in solving problem under consideration

```
from sklearn.model_selection import cross_val_score
print(cross_val_score(DTR,X_scaler,y,cv=5).mean())
print(cross_val_score(RFR,X_scaler,y,cv=5).mean())
print(cross_val_score(knn,X_scaler,y,cv=5).mean())
print(cross_val_score(ada,X_scaler,y,cv=5).mean())
print(cross_val_score(gbc,X_scaler,y,cv=5).mean())
```

```
0.8016441993510671
0.8847541612102676
0.48397066087602614
-0.013534065156029996
0.7852188681677597
```

By performing cross validation and comparing the r2 score, we find Random Forest Regressor to be our best model. Now, we will be performing hyperparameter tuning on our best model and see if we can increase our accuracy and also check if our model is over fitting or under fitting.

```
parameters={'n_estimators':[200,300,400,500,600,700],
            'max_features':['auto','sqrt','log2'],
            'criterion':['squared_error','absolute_error','poisson'],
            'max_depth':[5,10,15,20,25],
             'min_samples_leaf': [1, 2, 4, 6],
            'n_jobs':[-1]}
```

```
GCV=GridSearchCV(RandomForestRegressor(),parameters,cv=5)
```

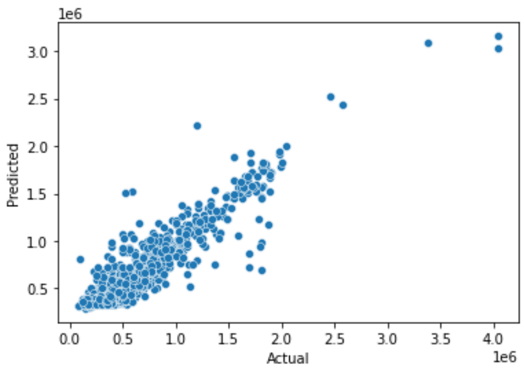
```
GCV.fit(X_train,Y_train)
```

```
Final_model=RandomForestRegressor(min_samples_leaf=1,n_estimators=500,max_features='sqrt',criterion='poisson',max_depth=25,n_jobs=-1)
Final_model.fit(X_train,Y_train)
pred=Final_model.predict(x_test)
acc=r2_score(y_test,pred)
print('Accuracy:',acc*100)
```

```
Accuracy: 85.15541452647427
```

```
plt.figure()
sns.scatterplot(y_test, pred)
plt.xlabel('Actual')
plt.ylabel('Predicted')
```

```
Text(0, 0.5, 'Predicted')
```

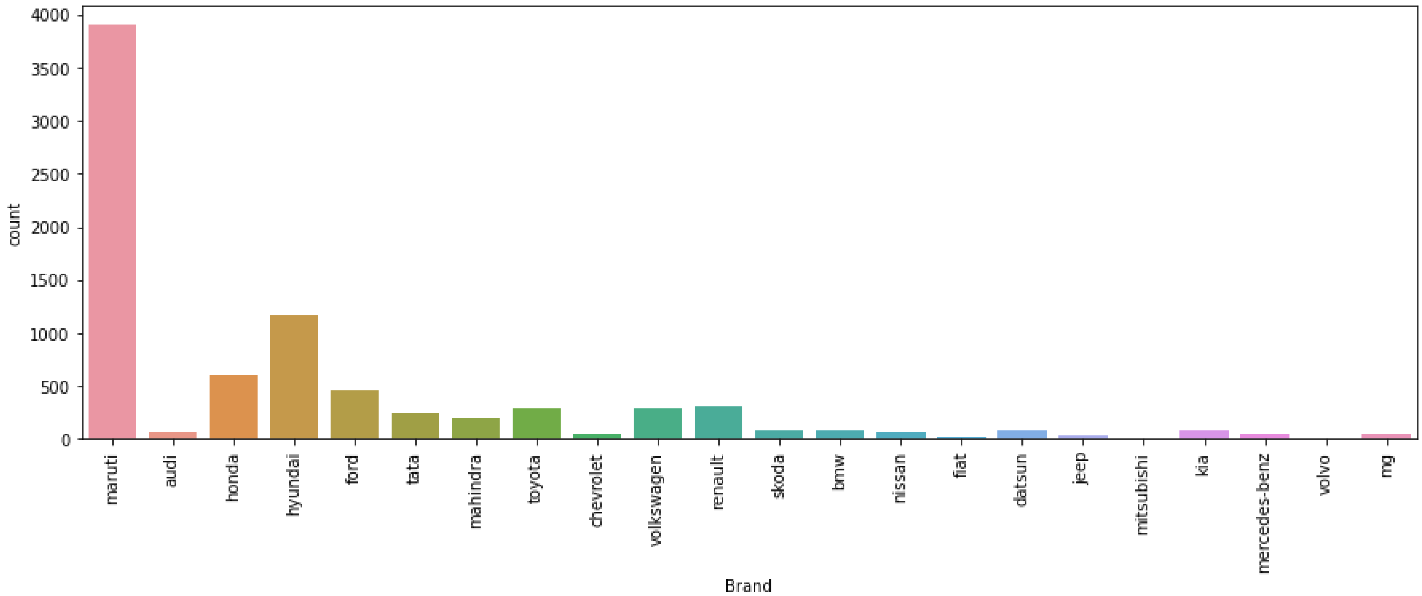



```
import joblib
joblib.dump(Final_model,"UsedCarPrice.pkl")
```

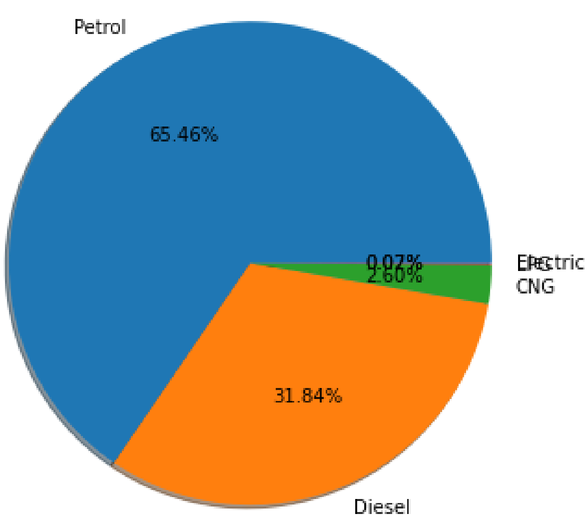
```
['UsedCarPrice.pkl']
```

Visualizations

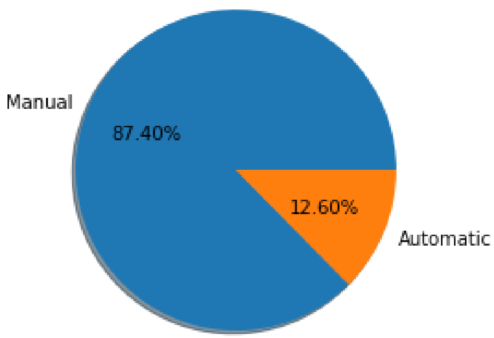
```
plt.figure(figsize=(15,5))
sns.countplot(df['Brand'])
plt.xticks(rotation = 90)
```



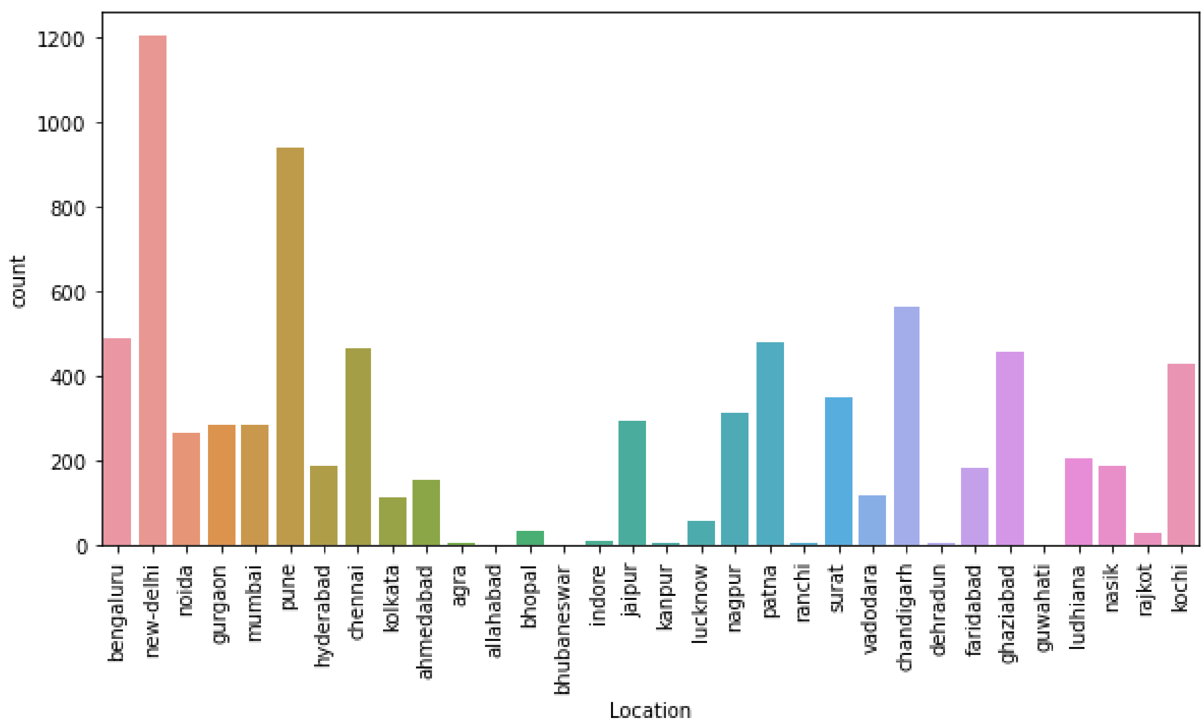

```
labels='Petrol','Diesel','CNG','LPG','Electric'

fig,ax=plt.subplots()
ax.pie(df['Fuel Type'].value_counts(),labels=labels,autopct='%1.2f%%',shadow=True,radius=1.5)

plt.show()
```



```python
labels='Manual','Automatic'

fig,ax=plt.subplots()
ax.pie(df['Transmission'].value_counts(),labels=labels,autopct='%1.2f%%',shadow=True)

plt.show()
```

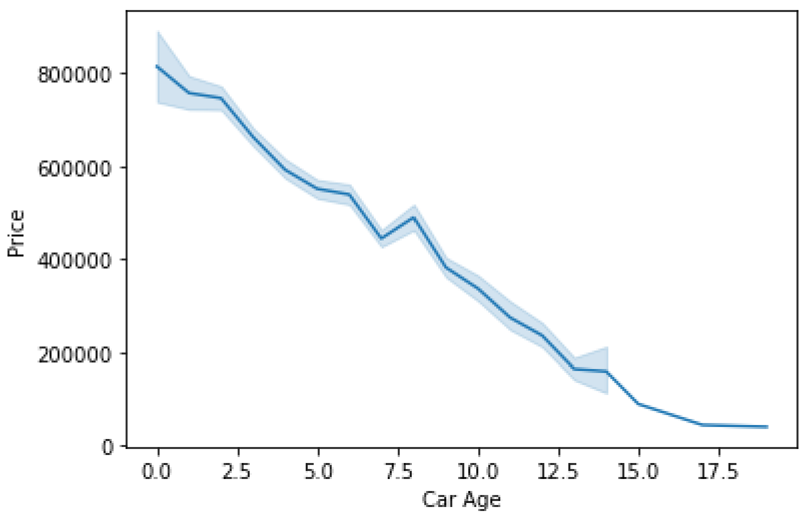



```python
plt.figure(figsize=(10,5))
sns.countplot(df['Location'])
plt.xticks(rotation = 90)
```



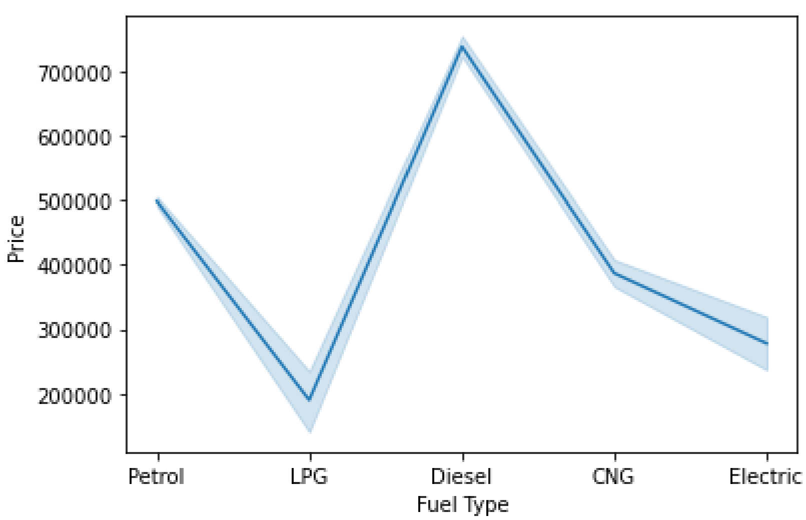
```
1  sns.lineplot(x='Car Age', y='Price', data=df)
```

<AxesSubplot:xlabel='Car Age', ylabel='Price'>



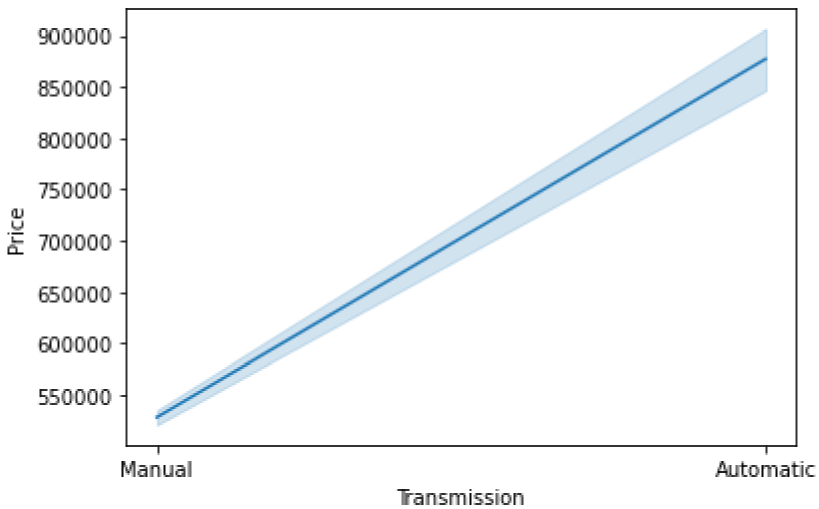

```
1  sns.lineplot(x='Fuel Type', y='Price', data=df)
```

<AxesSubplot:xlabel='Fuel Type', ylabel='Price'>



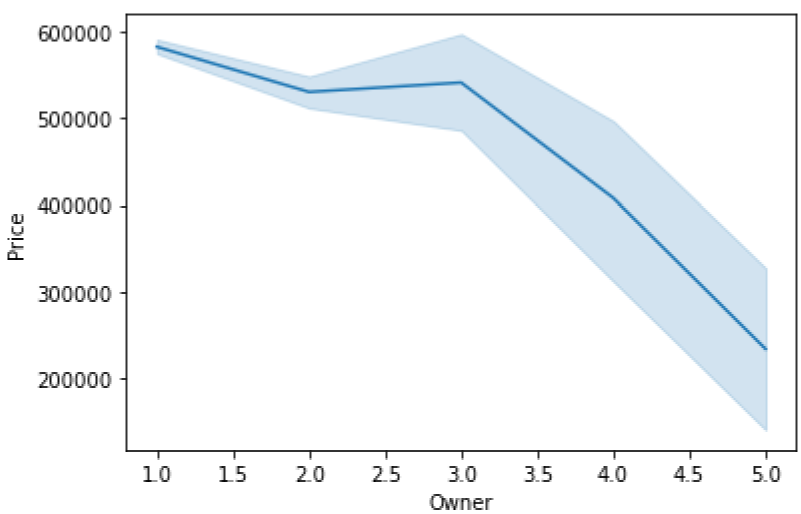
```
sns.lineplot(x='Transmission',y='Price',data=df)
```

```
<AxesSubplot:xlabel='Transmission', ylabel='Price'>
```

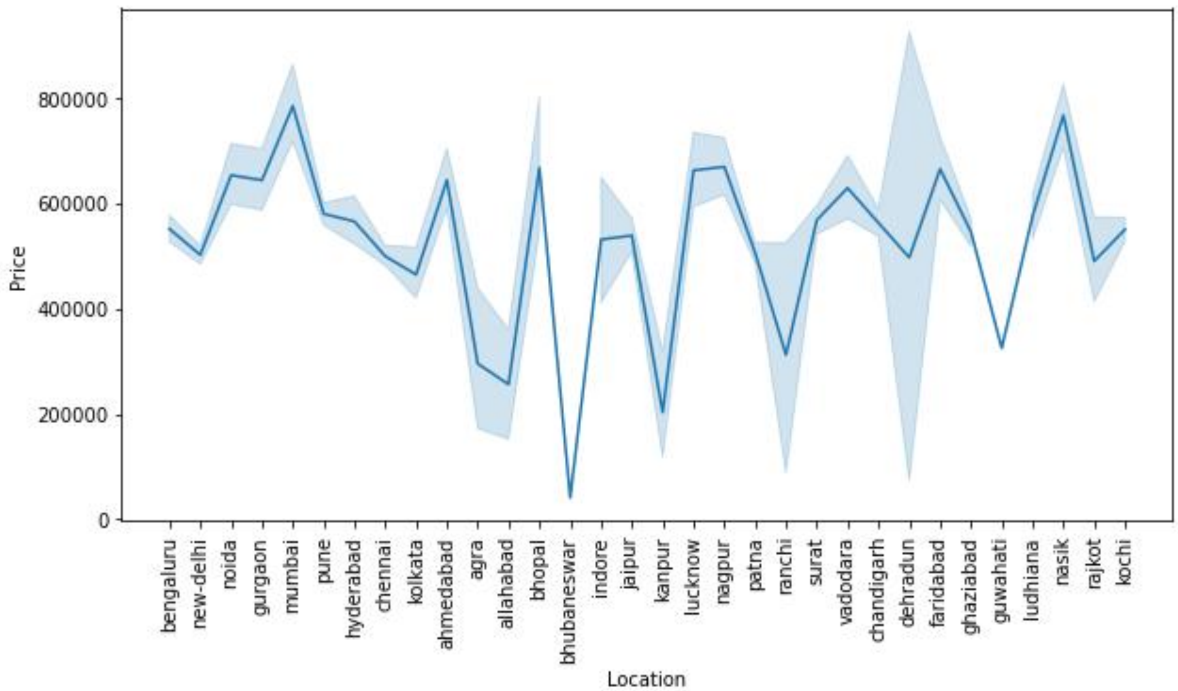



```
sns.lineplot(x='Owner',y='Price',data=df)
```

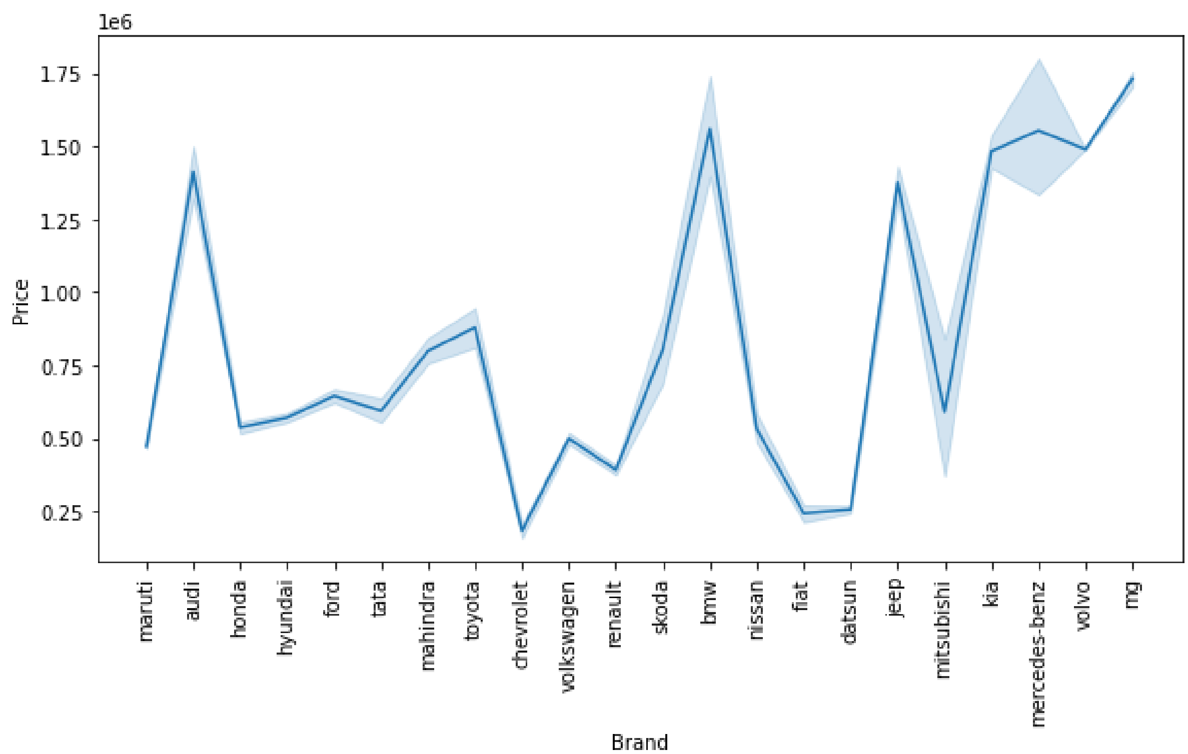
```
<AxesSubplot:xlabel='Owner', ylabel='Price'>
```




```
plt.figure(figsize=(10,5))
sns.lineplot(x='Location',y='Price',data=df)
plt.xticks(rotation = 90)
```




```
plt.figure(figsize=(10,5))
sns.lineplot(x='Brand',y='Price',data=df)
plt.xticks(rotation = 90)
```



```
corr=df.corr()
corr.shape
```

```
(10, 10)
```

```
plt.figure(figsize=(15,10))

sns.heatmap(corr,cbar=True,square=True,cbar_kws={'shrink':0.82},fmt='.2f',annot=True,annot_kws={'size':10})
plt.show()
```

	Brand	Model	Variant	Fuel Type	Transmission	KilometersDriven	Owner	Location	Price	Car Age
Brand	1.00	0.17	0.26	0.03	0.02	-0.00	-0.05	-0.01	-0.06	-0.05
Model	0.17	1.00	0.23	-0.14	0.02	0.01	-0.03	0.03	0.01	0.12
Variant	0.26	0.23	1.00	0.01	0.03	0.00	-0.06	0.02	-0.16	-0.02
Fuel Type	0.03	-0.14	0.01	1.00	-0.01	-0.01	-0.05	-0.08	-0.27	-0.10
Transmission	0.02	0.02	0.03	-0.01	1.00	-0.00	-0.07	-0.02	-0.33	0.03
KilometersDriven	-0.00	0.01	0.00	-0.01	-0.00	1.00	-0.02	0.02	0.02	-0.06
Owner	-0.05	-0.03	-0.06	-0.05	-0.07	-0.02	1.00	-0.02	-0.06	0.28
Location	-0.01	0.03	0.02	-0.08	-0.02	0.02	-0.02	1.00	0.01	0.04
Price	-0.06	0.01	-0.16	-0.27	-0.33	0.02	-0.06	0.01	1.00	-0.36
Car Age	-0.05	0.12	-0.02	-0.10	0.03	-0.06	0.28	0.04	-0.36	1.00

From the above data visualizations we can conclude that:

- Maruti, Hyundai and Honda are the most sold cars.

- Petrol engine cars are most sold followed by diesel engine and CNG.
- Manual Transmission cars are most sold.
- States with most sold cars are New Delhi, Pune and Chandigarh.
- Car price decreases with increase in car age.
- Diesel cars are sold at higher prices whereas LPG is sold at lower price.
- Automatic cars are sold at higher prices as compared to manual cars.
- Car price decreases with increase in number of previous owners.
- Cars in Mumbai, Nasik, Bhopal and Nagpur are sold at higher price whereas Bhubaneswar has the cheapest price for used cars.
- Mg, BMW, Mercedes-benz are sold at higher price whereas Chevrolet, Fiat and Datsun are sold at lower price.
- There is no co-linearity issues in any of the variables.

Conclusion

Key findings and conclusion of the study

In this project, I sought to demonstrate how used vehicle prices change and what variables contribute to the fluctuation of automobile prices. The Random forest regressor model did a good job of forecasting prices.

Learning Outcomes of the study in respect of Data Science

This study has proved the significance of data modeling and prediction. I was able to analyze and explain several hidden insights regarding the data using various strong visualization tools. I was able to eliminate extraneous columns and outliers from our dataset using data cleaning, which would have resulted in over fitting or under fitting of our model.

Limitations of this work and scope for future work

As with every undertaking, there is always space for improvement. Because of the nature of this project, various algorithms may be connected as modules, and their findings can be pooled to maximize the accuracy of the final output. This model may be enhanced further by incorporating more algorithms. The output of these algorithms, however, must be in the same format as the others. Once that criterion is met, the modules may be easily added as shown in the code. This gives the project a high level of adaptability and scalability.