FLIP ROBO

CAR PRICE PREDICTION PROJECT

Submitted by:

AAYUSHI LASHKARI

ACKNOWLEDGMENT

I would like to express my special thanks to my mentor Mr. Shubham Yadav who gave me his support and assistance throughout the project. I am thankful to flip robo technologies and Data Trained institute to give me guidelines and knowledge to complete this project. Links and websites that I preferred: https://towardsdatascience.com/oversampling-and-undersampling-5e2bbaf56dcf https://career-resource-center.udacity.com/portfolio/datasciencereports#:~:text=A%20data%20science%20report%20is,the%20legitimac y%20o f%20your%20process

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. 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.

- Conceptual Background of the Domain Problem

Data collection is the first and important part of this case now a days selling of used cars is most common and it is important to have knowledge of cars that what features one should include in the dataset. That feature people focus at the time of buying used cars.

- Review of Literature

Before starting this project one should have enough knowledge of cars and different categories of possible features and it should be helpful to know the correlation

Between different features.

- Motivation for the Problem Undertaken

Now a days used cars are very common it is important to know that what brand and features of car is giving weightage to the price of the car.

It will be helpful for both the clients and traders to understand the whole scenario.

Analytical Problem Framing

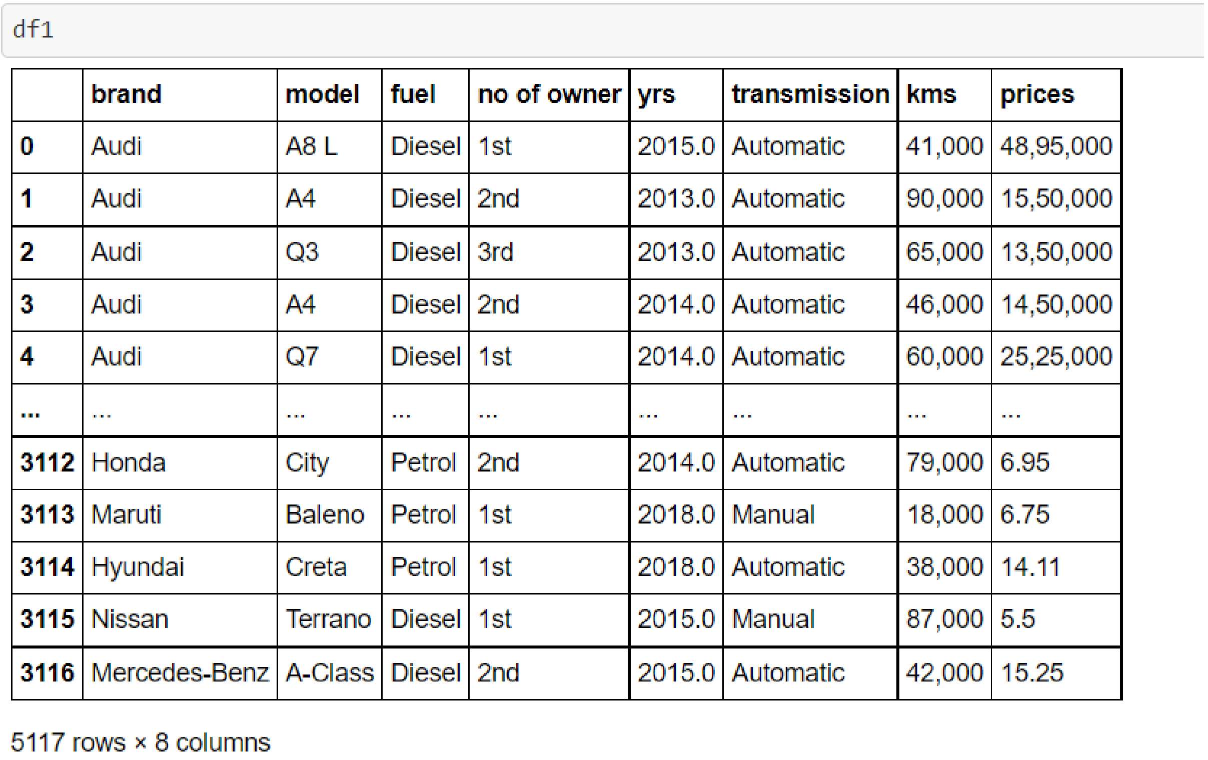
- ## Mathematical/ Analytical Modeling of the Problem

I have scraped data from olx , cardekho and car24 , number of features I have scraped are eight including the target variable. Independent variables are Name of the model , Name of the brand ,Number of owner , transmission type , fuel type , year of manufacture , Kilo meters and Prices of the car is target variable. Number of rows are 5116.

- ## Data Sources and their formats

I have collected data from different websites and the format of data for the same column is different for different websites.
For example: In prices of the cars in one website the data is in lakhs and in some with rupees symbol .
Also in number of owners in some websites it is in the form of "first owner" and for other "1st owner".

Dataset after combining data of different websites.



```
df1
```

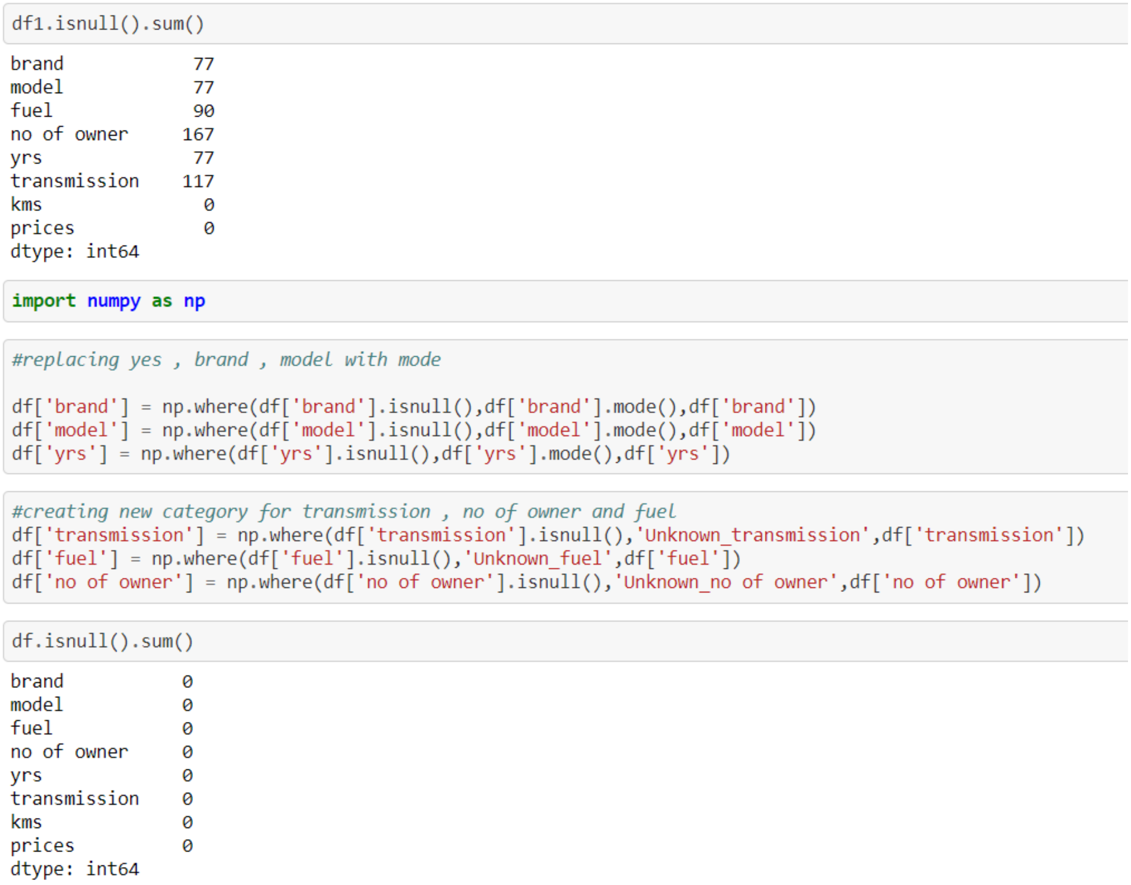
	brand	model	fuel	no of owner	yrs	transmission	kms	prices
0	Audi	A8 L	Diesel	1st	2015.0	Automatic	41,000	48,95,000
1	Audi	A4	Diesel	2nd	2013.0	Automatic	90,000	15,50,000
2	Audi	Q3	Diesel	3rd	2013.0	Automatic	65,000	13,50,000
3	Audi	A4	Diesel	2nd	2014.0	Automatic	46,000	14,50,000
4	Audi	Q7	Diesel	1st	2014.0	Automatic	60,000	25,25,000
...	...	...	...	...	...	...	...	...
3112	Honda	City	Petrol	2nd	2014.0	Automatic	79,000	6.95
3113	Maruti	Baleno	Petrol	1st	2018.0	Manual	18,000	6.75
3114	Hyundai	Creta	Petrol	1st	2018.0	Automatic	38,000	14.11
3115	Nissan	Terrano	Diesel	1st	2015.0	Manual	87,000	5.5
3116	Mercedes-Benz	A-Class	Diesel	2nd	2015.0	Automatic	42,000	15.25

5117 rows × 8 columns

- ## Data Preprocessing Done

Pre-processing Pipe Line is an important step towards the data modelling, To make data ready for prediction. Following Steps I followed for Pre Processing :

1. Making data in the same format for each column before combining the data into single dataset.
2. Changing the datatype of the columns or correcting the data types of the column.
3. Finding null values in the features and treat them accordingly.
4. Encoding of the data.
5. Scaling the data.



```
df1.isnull().sum()
```

```
brand            77
model            77
fuel             90
no of owner     167
yrs              77
transmission    117
kms               0
prices            0
dtype: int64
```

```
import numpy as np
```

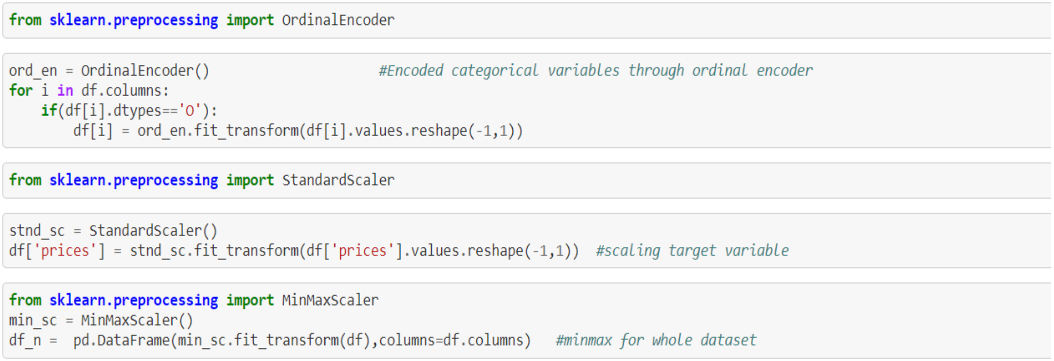
```
#replacing yes , brand , model with mode

df['brand'] = np.where(df['brand'].isnull(),df['brand'].mode(),df['brand'])
df['model'] = np.where(df['model'].isnull(),df['model'].mode(),df['model'])
df['yrs'] = np.where(df['yrs'].isnull(),df['yrs'].mode(),df['yrs'])
```

```
#creating new category for transmission , no of owner and fuel
df['transmission'] = np.where(df['transmission'].isnull(),'Unknown_transmission',df['transmission'])
df['fuel'] = np.where(df['fuel'].isnull(),'Unknown_fuel',df['fuel'])
df['no of owner'] = np.where(df['no of owner'].isnull(),'Unknown_no of owner',df['no of owner'])
```

```
df.isnull().sum()
```

```
brand           0
model           0
fuel            0
no of owner     0
yrs             0
transmission    0
kms             0
prices          0
dtype: int64
```



```
from sklearn.preprocessing import OrdinalEncoder
```

```
ord_en = OrdinalEncoder()                    #Encoded categorical variables through ordinal encoder
for i in df.columns:
    if(df[i].dtypes=='O'):
        df[i] = ord_en.fit_transform(df[i].values.reshape(-1,1))
```

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

```
stnd_sc = StandardScaler()
df['prices'] = stnd_sc.fit_transform(df['prices'].values.reshape(-1,1))  #scaling target variable
```

```
from sklearn.preprocessing import MinMaxScaler
min_sc = MinMaxScaler()
df_n =  pd.DataFrame(min_sc.fit_transform(df),columns=df.columns)   #minmax for whole dataset
```

- Data Inputs- Logic- Output Relationships

In this case the target variable is continuous and two
Variables are numeric except that :
Year and distance (km).
Where Year is negatively correlated and Km is positively correlated with the price of the cars.

- Hardware and Software Requirements and Tools Used

Tool: Jupyter NoteBook 6.1.4
• Web-based interactive computing notebook Environment.
Software Requirement:
• The client environment may be Windows, macOS, or Linux. Hardware Requirement:
• CPU:
2 x 64-bit, 2.8 GHz, 8.00 GT/s CPUs or better.
• Memory:
minimum RAM size of 32 GB, or 16 GB RAM with 1600 MHz DDR3 installed, for a typical installation with 50 regular users. Libraries:
• Pandas: For reading CSV file, Converting dataset into a data frame, handling date datatype, and more.
• Seaborn and matplotlib: For EDA and Visualization

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

Most time consuming step in this case is to make the data in the similar format like number of owner of one website is "first owner " form and for other website it is "1st" owner. And removing unwanted things like km from the distance And rupees sigh from the prices. Then also some * symbols were still there that I have replaced with the numeric values only.
Also in the transmission and the brand name some unwanted was there that I have replaced with new category.

- Testing of Identified Approaches (Algorithms)

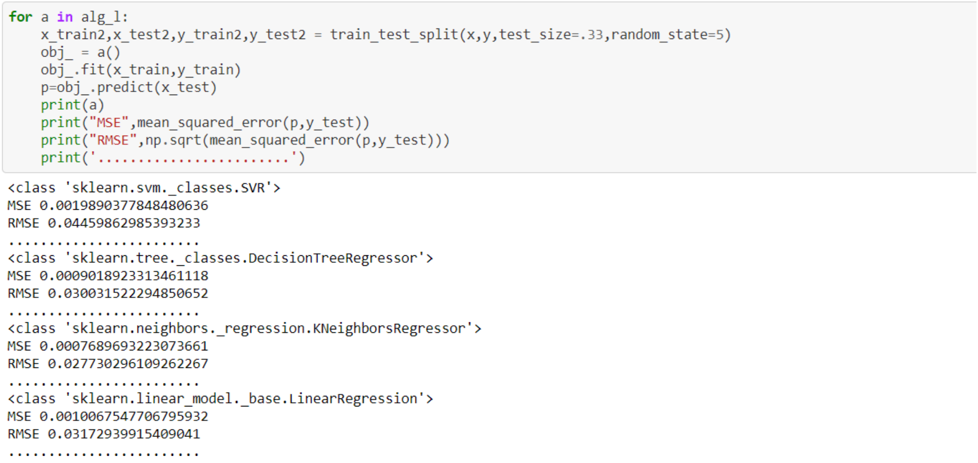
Following are the algorithms I have used in this problem :

1. Linear Regression.

2. Support Vector Regressor.
3. Decision Tree Regressor.
4. Kneighbors Regressor

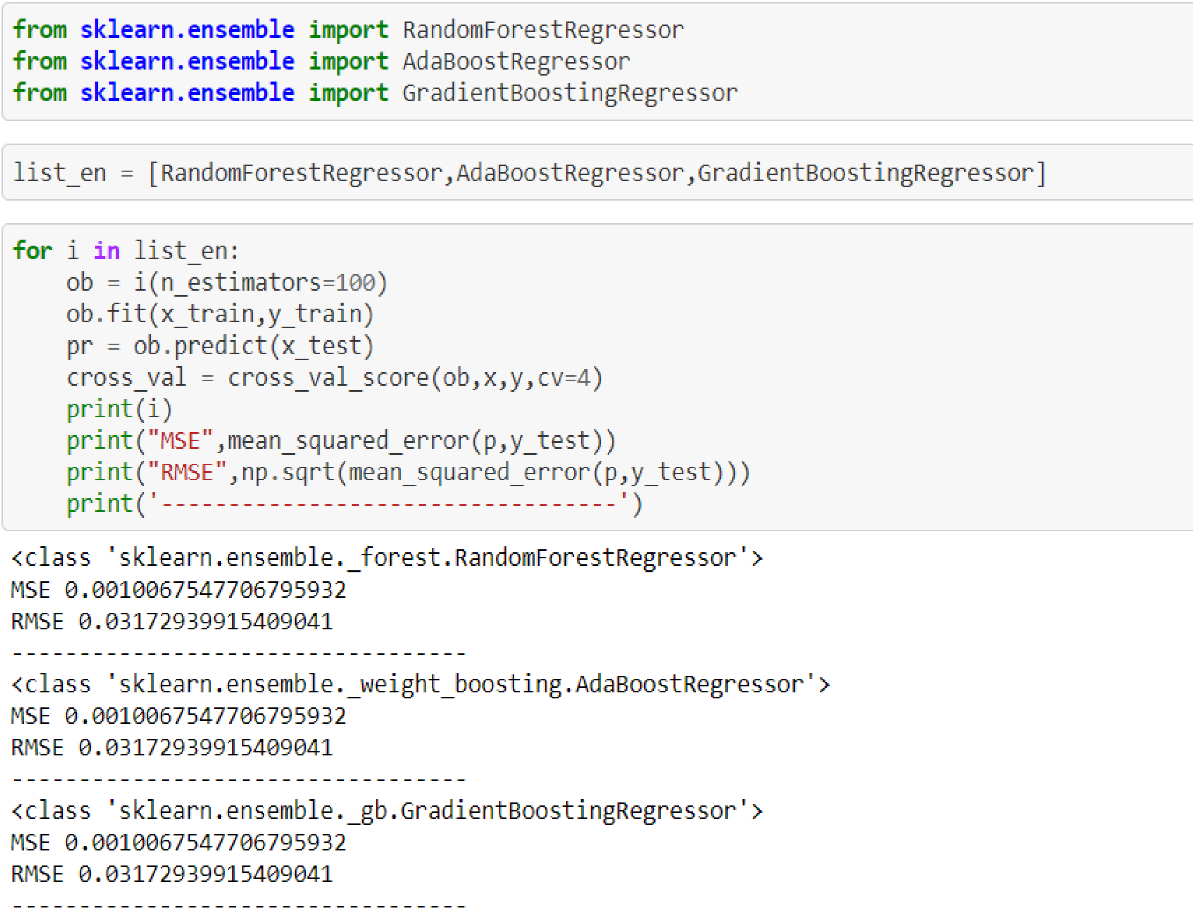
Different Ensemble Algorithm:

1. Random Forest
2. Gradient Descent
3. Adaboost Regressor



```
for a in alg_l:
    x_train2,x_test2,y_train2,y_test2 = train_test_split(x,y,test_size=.33,random_state=5)
    obj_ = a()
    obj_.fit(x_train,y_train)
    p=obj_.predict(x_test)
    print(a)
    print("MSE",mean_squared_error(p,y_test))
    print("RMSE",np.sqrt(mean_squared_error(p,y_test)))
    print('........................')
```

```
<class 'sklearn.svm._classes.SVR'>
MSE 0.0019890377848480636
RMSE 0.04459862985393233
........................
<class 'sklearn.tree._classes.DecisionTreeRegressor'>
MSE 0.0009018923313461118
RMSE 0.030031522294850652
........................
<class 'sklearn.neighbors._regression.KNeighborsRegressor'>
MSE 0.0007689693223073661
RMSE 0.027730296109262267
........................
<class 'sklearn.linear_model._base.LinearRegression'>
MSE 0.0010067547706795932
RMSE 0.03172939915409041
........................
```



```
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn.ensemble import GradientBoostingRegressor
```

```
list_en = [RandomForestRegressor,AdaBoostRegressor,GradientBoostingRegressor]
```

```
for i in list_en:
    ob = i(n_estimators=100)
    ob.fit(x_train,y_train)
    pr = ob.predict(x_test)
    cross_val = cross_val_score(ob,x,y,cv=4)
    print(i)
    print("MSE",mean_squared_error(p,y_test))
    print("RMSE",np.sqrt(mean_squared_error(p,y_test)))
    print('----------------------------------')
```

```
<class 'sklearn.ensemble._forest.RandomForestRegressor'>
MSE 0.0010067547706795932
RMSE 0.03172939915409041
----------------------------------
<class 'sklearn.ensemble._weight_boosting.AdaBoostRegressor'>
MSE 0.0010067547706795932
RMSE 0.03172939915409041
----------------------------------
<class 'sklearn.ensemble._gb.GradientBoostingRegressor'>
MSE 0.0010067547706795932
RMSE 0.03172939915409041
----------------------------------
```

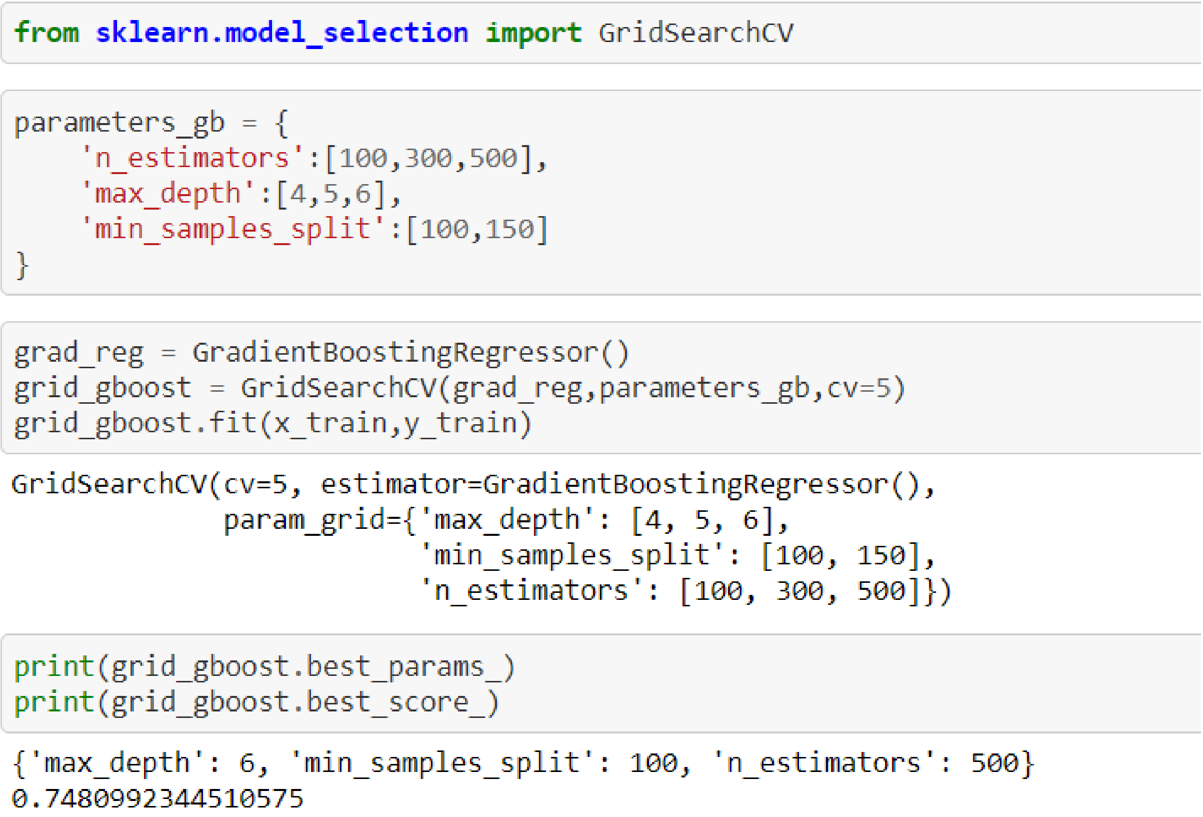
- Run and Evaluate selected models

Selected Model is Hyper parametric tuning model of gradient descent regressor.

With r2 score :0.74

MSE : 0.007

RMSE : 0.267



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

```
parameters_gb = {
    'n_estimators':[100,300,500],
    'max_depth':[4,5,6],
    'min_samples_split':[100,150]
}
```

```
grad_reg = GradientBoostingRegressor()
grid_gboost = GridSearchCV(grad_reg,parameters_gb,cv=5)
grid_gboost.fit(x_train,y_train)
```

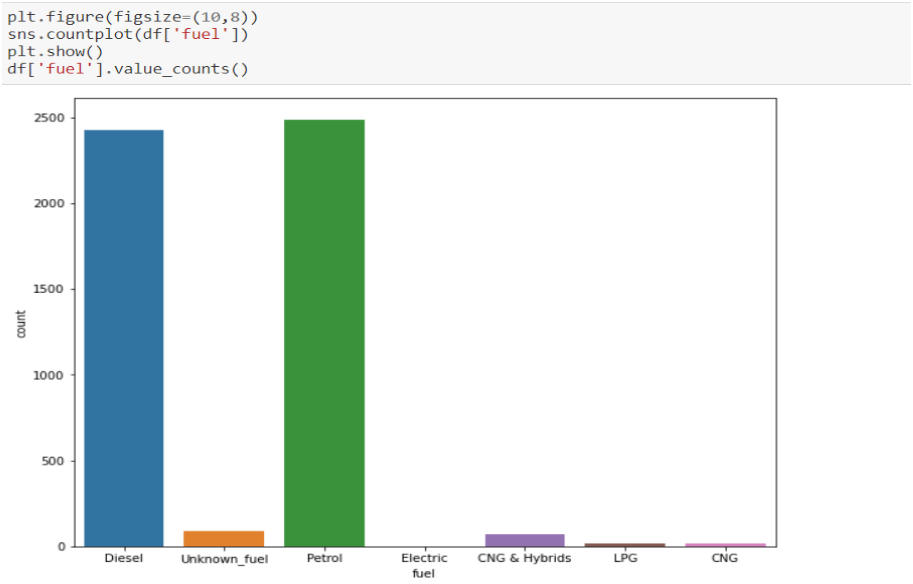
```
GridSearchCV(cv=5, estimator=GradientBoostingRegressor(),
             param_grid={'max_depth': [4, 5, 6],
                         'min_samples_split': [100, 150],
                         'n_estimators': [100, 300, 500]})
```

```
print(grid_gboost.best_params_)
print(grid_gboost.best_score_)
```

```
{'max_depth': 6, 'min_samples_split': 100, 'n_estimators': 500}
0.7480992344510575
```

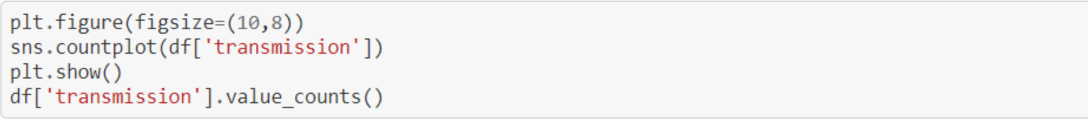
- Visualizations

1. Fuel :

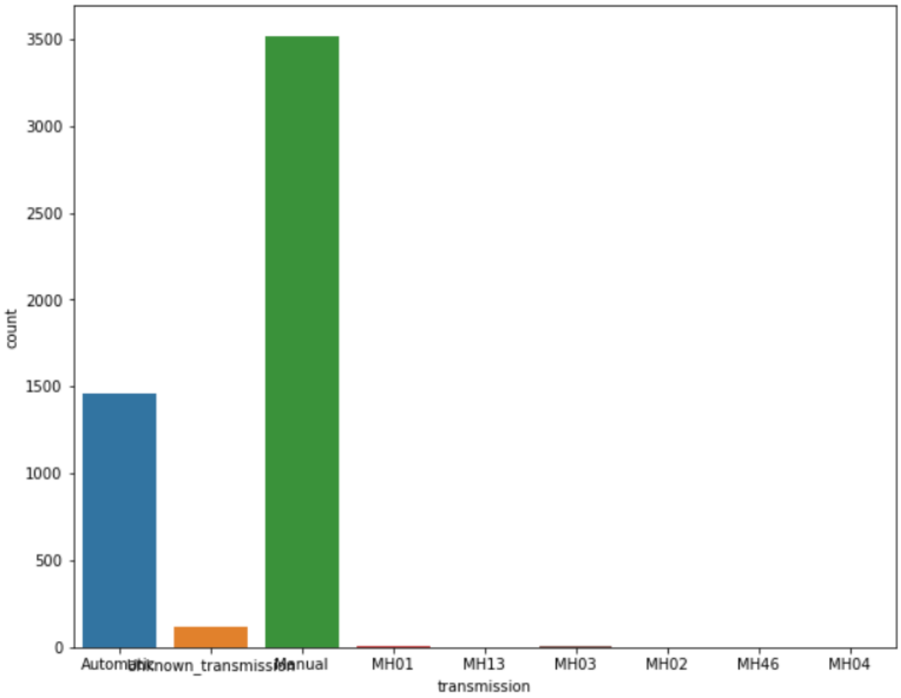


```
plt.figure(figsize=(10,8))
sns.countplot(df['fuel'])
plt.show()
df['fuel'].value_counts()
```

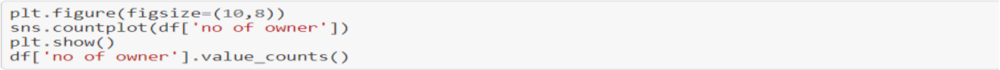
2. Transmission



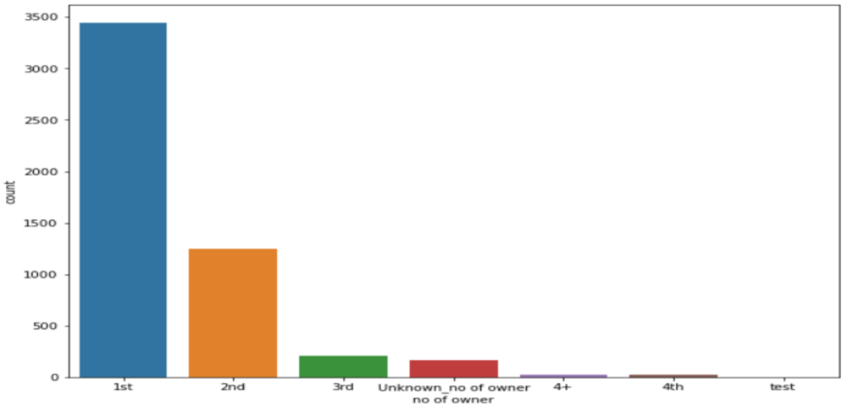
```
plt.figure(figsize=(10,8))
sns.countplot(df['transmission'])
plt.show()
df['transmission'].value_counts()
```



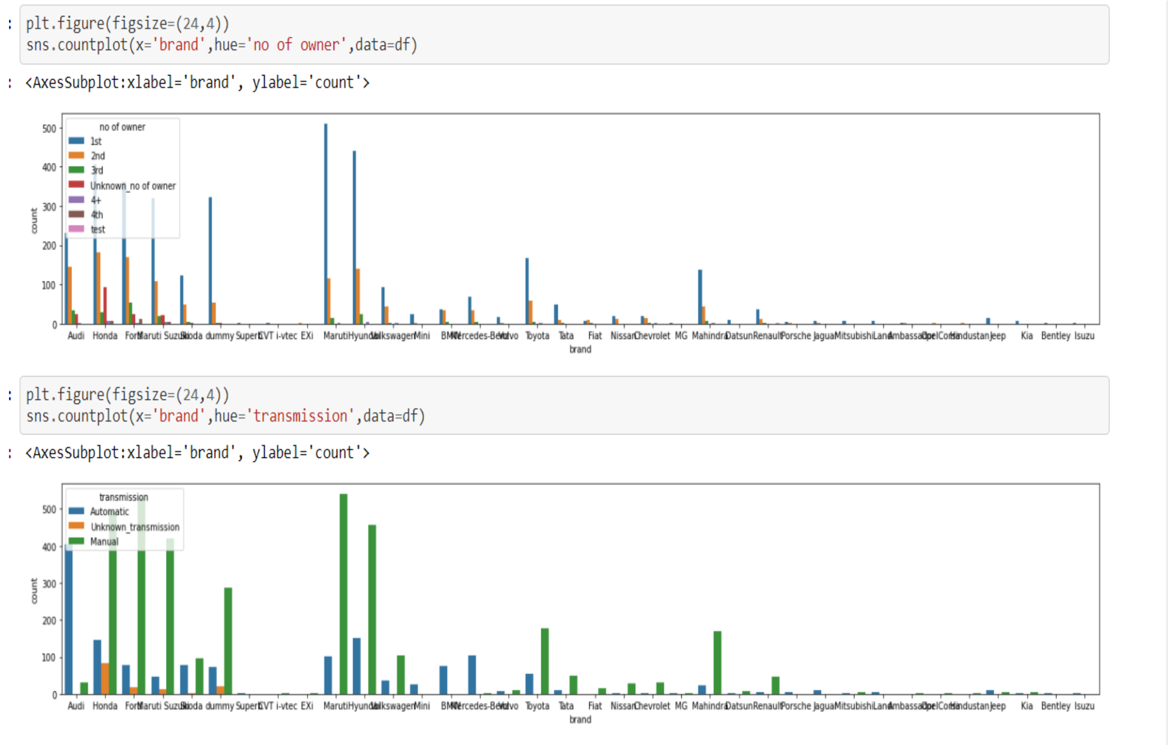
3. Number of owner:



```
plt.figure(figsize=(10,8))
sns.countplot(df['no of owner'])
plt.show()
df['no of owner'].value_counts()
```



4. Brand with Transmission and Number of owner:



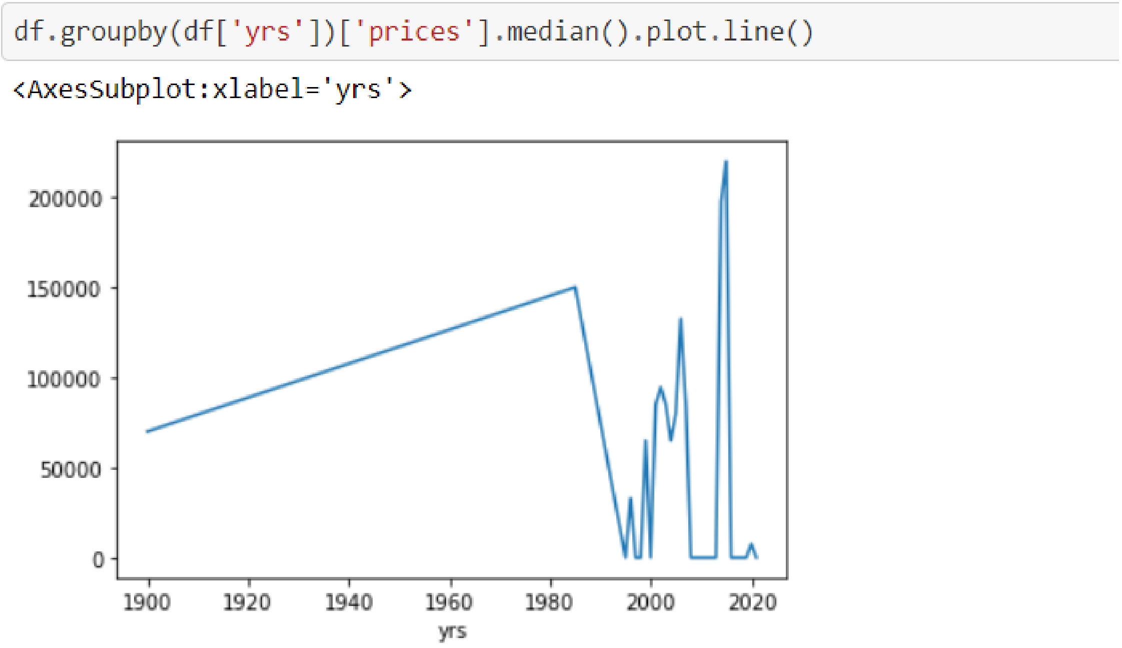
```
: plt.figure(figsize=(24,4))
  sns.countplot(x='brand',hue='no of owner',data=df)
```

```
: <AxesSubplot:xlabel='brand', ylabel='count'>
```

```
: plt.figure(figsize=(24,4))
  sns.countplot(x='brand',hue='transmission',data=df)
```

```
: <AxesSubplot:xlabel='brand', ylabel='count'>
```

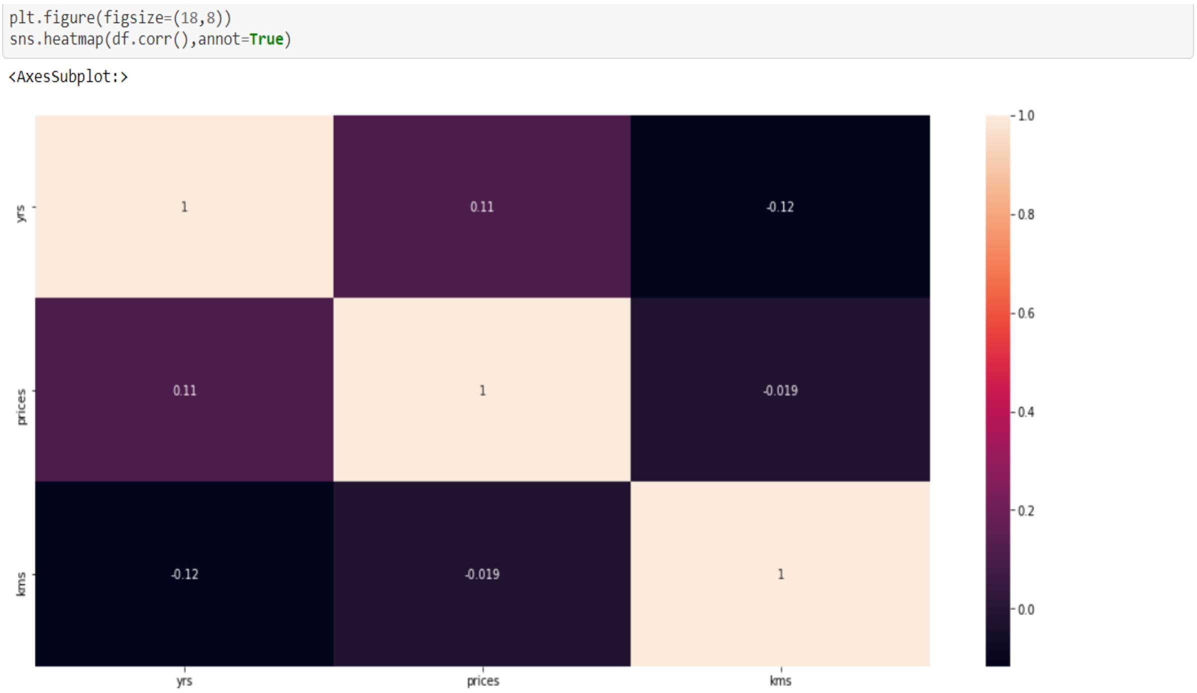
5. Price with year:



```
df.groupby(df['yrs'])['prices'].median().plot.line()
```

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

6. Multivariate Analysis:



```
plt.figure(figsize=(18,8))
sns.heatmap(df.corr(),annot=True)
```

```
<AxesSubplot:>
```

Interpretation of the Results

The results are obvious the fuel type is most frequent is diesel and petrol . And transmission is manual.

Year is negatively correlated with price and Km is positively correlated

With prices.

Audi , Ford and Honda are the brands that provides diesel .

Audi provides automatic transmission.

CONCLUSION

- ## Key Findings and Conclusions of the Study

1. The key step of this problem is to collect the data and make that in the same format and the unwanted values.
2. Key finding is that the most time consuming thing is data cleaning and data collection.
3. Data cleaning includes formatting data in same and correct format , find null values and treat them.
4. As the common scenario the price of the car depends on the year and km and brand.
5. Audi gives some different insights that it's most cars
Are diesel and with automatic transmission.

- ## Learning Outcomes of the Study in respect of Data Science

As I have scraped the data by my own it will be a good learning to know how to fetch the data and what data will be useful for which output.
To understand which steps to take while work on the data from different sources.
As unnecessary symbols in the data and removing that is time taking task.
To separate the unwanted data like km and symbol of currency etc.

- ## Limitations of this work and Scope for Future Work

The size of the dataset is important because of the large dataset and

Diversified data the model can learn do well for that.

My dataset contains 5000 rows and if the dataset size increase it will

be good for the model building and should have given good results.