CC-314 & CC-315 Mini - Project

Name: CHAVDA DIVYESH KETANKUMAR
Course: AIML
Roll No.: 07

Used Cars Price Prediction:

Data Understanding:

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
car1 = pd.read_csv(r"F:\Sem 6\ML Mini Project\train-data.csv")
car_dp = pd.read_csv(r"F:\Sem 6\ML Mini Project\test-data.csv")
```

In [3]:
```
car1.head()
```

Out[3]:

	Unnamed: 0	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	New_Price	Price
0	0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First	26.6 km/kg	998 CC	58.16 bhp	5.0	NaN	1.75
1	1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67 kmpl	1582 CC	126.2 bhp	5.0	NaN	12.50
2	2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	18.2 kmpl	1199 CC	88.7 bhp	5.0	8.61 Lakh	4.50
3	3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77 kmpl	1248 CC	88.76 bhp	7.0	NaN	6.00
4	4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.2 kmpl	1968 CC	140.8 bhp	5.0	NaN	17.74

In [4]:
```
car_dp.head()
```

Out[4]:

	Unnamed: 0	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	New_Price
0	0	Maruti Alto K10 LXI CNG	Delhi	2014	40929	CNG	Manual	First	32.26 km/kg	998 CC	58.2 bhp	4.0	NaN
1	1	Maruti Alto 800 2016-2019 LXI	Coimbatore	2013	54493	Petrol	Manual	Second	24.7 kmpl	796 CC	47.3 bhp	5.0	NaN
2	2	Toyota Innova Crysta Touring Sport 2.4 MT	Mumbai	2017	34000	Diesel	Manual	First	13.68 kmpl	2393 CC	147.8 bhp	7.0	25.27 Lakh
3	3	Toyota Etios Liva GD	Hyderabad	2012	139000	Diesel	Manual	First	23.59 kmpl	1364 CC	null bhp	5.0	NaN
4	4	Hyundai i20 Magna	Mumbai	2014	29000	Petrol	Manual	First	18.5 kmpl	1197 CC	82.85 bhp	5.0	NaN

In [5]:
```
car1.shape
```

Out[5]:
```
(6019, 14)
```

In [6]:
```
car_dp.shape
```

Out[6]:
```
(1234, 13)
```

In [7]:
```
car1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6019 entries, 0 to 6018
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Unnamed: 0         6019 non-null   int64  
 1   Name               6019 non-null   object 
 2   Location           6019 non-null   object 
 3   Year               6019 non-null   int64  
 4   Kilometers_Driven  6019 non-null   int64  
 5   Fuel_Type          6019 non-null   object 
 6   Transmission       6019 non-null   object 
 7   Owner_Type         6019 non-null   object 
 8   Mileage            6017 non-null   object 
 9   Engine             5983 non-null   object 
 10  Power              5983 non-null   object 
 11  Seats              5977 non-null   float64
 12  New_Price          824 non-null    object 
 13  Price              6019 non-null   float64
dtypes: float64(2), int64(3), object(9)
memory usage: 658.5+ KB
```

In [8]:
```
car_dp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1234 entries, 0 to 1233
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Unnamed: 0         1234 non-null   int64  
 1   Name               1234 non-null   object 
 2   Location           1234 non-null   object 
 3   Year               1234 non-null   int64  
 4   Kilometers_Driven  1234 non-null   int64  
 5   Fuel_Type          1234 non-null   object 
 6   Transmission       1234 non-null   object 
 7   Owner_Type         1234 non-null   object 
 8   Mileage            1234 non-null   object 
 9   Engine             1224 non-null   object 
 10  Power              1224 non-null   object 
 11  Seats              1223 non-null   float64
 12  New_Price          182 non-null    object 
dtypes: float64(1), int64(3), object(9)
memory usage: 125.5+ KB
```

Data Cleaning:

In [9]:
```
car1 = car1.drop(['Unnamed: 0','New_Price'],axis=1)
car_dp = car_dp.drop(['Unnamed: 0','New_Price'],axis=1)
```

In [10]:
```
car1['Location'].value_counts()
```

Out[10]:
```
Mumbai        790
Hyderabad     742
Kochi         651
Coimbatore    636
Pune          622
Delhi         554
Kolkata       535
Chennai       494
Jaipur        413
Bangalore     358
Ahmedabad     224
Name: Location, dtype: int64
```

In [11]:
```
car_dp['Location'].value_counts()
```

Out[11]:
```
Mumbai        159
Pune          143
Coimbatore    136
Hyderabad     134
Kochi         121
Kolkata       119
Delhi         106
Chennai        97
Jaipur         86
Bangalore      82
Ahmedabad      51
Name: Location, dtype: int64
```

In [12]:
```
car1['Fuel_Type'].value_counts()
```

Out[12]:
```
Diesel      3205
Petrol      2746
CNG           56
LPG           10
Electric       2
Name: Fuel_Type, dtype: int64
```

In [13]:
```
car_dp['Fuel_Type'].value_counts()
```

Out[13]:
```
Diesel    647
Petrol    579
CNG         6
LPG         2
Name: Fuel_Type, dtype: int64
```

In [14]:
```
i = car1[(car1['Fuel_Type']=='LPG') | (car1['Fuel_Type']=='Electric')].index
car1.drop(i,axis=0,inplace=True)
```

In [15]:
```
i = car_dp[(car_dp['Fuel_Type']=='LPG')].index
car_dp.drop(i,axis=0,inplace=True)
```

In [16]:
```
car1['Transmission'].value_counts()
```

Out[16]:
```
Manual       4289
Automatic    1718
Name: Transmission, dtype: int64
```

In [17]:
```
car_dp['Transmission'].value_counts()
```

Out[17]:
```
Manual       903
Automatic    329
Name: Transmission, dtype: int64
```

In [18]:
```
car1['Owner_Type'].value_counts()
```

Out[18]:
```
First             4919
Second             966
Third              113
Fourth & Above       9
Name: Owner_Type, dtype: int64
```

In [19]:
```
car_dp['Owner_Type'].value_counts()
```

Out[19]:
```
First             1022
Second             183
Third               24
Fourth & Above       3
Name: Owner_Type, dtype: int64
```

In [20]:
```
i = car1[(car1['Owner_Type']=='Fourth & Above')].index
car1.drop(i,axis=0,inplace=True)
```

In [21]:
```
i = car_dp[(car_dp['Owner_Type']=='Fourth & Above')].index
car_dp.drop(i,axis=0,inplace=True)
```

In [22]:
```
car1['Seats'].value_counts()
```

Out[22]:
```
5.0     4998
7.0      672
8.0      132
4.0       99
6.0       31
2.0       16
10.0       5
9.0        3
0.0        1
Name: Seats, dtype: int64
```

In [23]:
```
car_dp['Seats'].value_counts()
```

Out[23]:
```
5.0     1030
7.0      122
8.0       35
4.0       19
6.0        7
10.0       3
2.0        2
Name: Seats, dtype: int64
```

In [24]:
```
i = car1[(car1['Seats']==10.0) | (car1['Seats']==2.0) | (car1['Seats']==9.0) | (car1['Seats']==0.0)].index
car1.drop(i,axis=0,inplace=True)
```

In [25]:
```
i = car_dp[(car_dp['Seats']==10.0) | (car_dp['Seats']==2.0)].index
car_dp.drop(i,axis=0,inplace=True)
```

In [26]:
```
car1['Year'].value_counts()
```

Out[26]:
```
2014    794
2015    740
2016    738
2013    647
2017    586
2012    574
2011    461
2010    337
2018    297
2009    197
2008    170
2007    121
2019    101
2006     77
2005     56
2004     29
2003     17
2002     14
2001      7
2000      4
1998      4
1999      2
Name: Year, dtype: int64
```

In [27]:
```
car_dp['Year'].value_counts()
```

Out[27]:
```
2015    184
2016    144
2013    142
2014    128
2017    122
2011    113
2012    109
2010     64
2018     62
2009     53
2008     33
2007     23
2019     17
2006     11
2005      9
2004      4
2003      3
1996      1
2002      1
2000      1
Name: Year, dtype: int64
```

In [28]:
```
i = car1[(car1['Year']==2001) | (car1['Year']==2000)| (car1['Year']==1998) | (car1['Year']==1999)].index
car1.drop(i,axis=0,inplace=True)
```

In [29]:
```
i = car_dp[(car_dp['Year']==2000) | (car_dp['Year']==2002)| (car_dp['Year']==1996)].index
car_dp.drop(i,axis=0,inplace=True)
```

Shape after Cleaning:_

In [30]:
```
car1.shape
```

Out[30]:
```
(5956, 12)
```

In [31]:
```
car_dp.shape
```

Out[31]:
```
(1221, 11)
```

Data Preprocessing:

Transform Data type to numeric:_

In [32]:
```
import re
```

In [33]:
```
car1['Mileage'] = car1['Mileage'].str.replace('[^0-9.-]', '')
car1['Engine'] = car1['Engine'].str.replace('[^0-9.-]', '')
car1['Power'] = car1['Power'].str.replace('[^0-9.-]', '')

car1['Mileage'] = pd.to_numeric(car1['Mileage'])
car1['Engine'] = pd.to_numeric(car1['Engine'])
car1['Power'] = pd.to_numeric(car1['Power'])

car1.head()
```

Out[33]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price
0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First	26.60	998.0	58.16	5.0	1.75
1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67	1582.0	126.20	5.0	12.50
2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	18.20	1199.0	88.70	5.0	4.50
3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77	1248.0	88.76	7.0	6.00
4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.20	1968.0	140.80	5.0	17.74

In [34]:
```
car_dp['Mileage'] = car_dp['Mileage'].str.replace('[^0-9.-]', '')
car_dp['Engine'] = car_dp['Engine'].str.replace('[^0-9.-]', '')
car_dp['Power'] = car_dp['Power'].str.replace('[^0-9.-]', '')

car_dp['Mileage'] = pd.to_numeric(car_dp['Mileage'])
car_dp['Engine'] = pd.to_numeric(car_dp['Engine'])
car_dp['Power'] = pd.to_numeric(car_dp['Power'])

car_dp.head()
```

Out[34]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats
0	Maruti Alto K10 LXI CNG	Delhi	2014	40929	CNG	Manual	First	32.26	998.0	58.20	4.0
1	Maruti Alto 800 2016-2019 LXI	Coimbatore	2013	54493	Petrol	Manual	Second	24.70	796.0	47.30	5.0
2	Toyota Innova Crysta Touring Sport 2.4 MT	Mumbai	2017	34000	Diesel	Manual	First	13.68	2393.0	147.80	7.0
3	Toyota Etios Liva GD	Hyderabad	2012	139000	Diesel	Manual	First	23.59	1364.0	NaN	5.0
4	Hyundai i20 Magna	Mumbai	2014	29000	Petrol	Manual	First	18.50	1197.0	82.85	5.0

In [35]:
```
car1.isna().sum()
```

Out[35]:
```
Name                 0
Location             0
Year                 0
Kilometers_Driven    0
Fuel_Type            0
Transmission         0
Owner_Type           0
Mileage              0
Engine              34
Power              132
Seats               40
Price                0
dtype: int64
```

In [36]:
```
car_dp.isna().sum()
```

Out[36]:
```
Name                 0
Location             0
Year                 0
Kilometers_Driven    0
Fuel_Type            0
Transmission         0
Owner_Type           0
Mileage              0
Engine              10
Power               29
Seats               11
dtype: int64
```

In [37]:
```
print((car1['Mileage']==0).any())
print((car1['Power']==0).any())
print((car1['Kilometers_Driven']==0).any())
print((car1['Engine']==0).any())
```

```
True
False
False
False
```

In [38]:
```
print((car_dp['Mileage']==0).any())
print((car_dp['Power']==0).any())
print((car_dp['Kilometers_Driven']==0).any())
print((car_dp['Engine']==0).any())
```

```
True
False
False
False
```

In [39]:
```
car1.loc[(car1['Mileage']==0)]
```

Out[39]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price
14	Land Rover Freelander 2 TD4 SE	Pune	2012	85000	Diesel	Automatic	Second	0.0	2179.0	115.0	5.0	17.50
67	Mercedes-Benz C-Class Progressive C 220d	Coimbatore	2019	15369	Diesel	Automatic	First	0.0	1950.0	194.0	5.0	35.67
79	Hyundai Santro Xing XL	Hyderabad	2005	87591	Petrol	Manual	First	0.0	1086.0	NaN	5.0	1.30
194	Honda City 1.5 GXI	Ahmedabad	2007	60006	Petrol	Manual	First	0.0	NaN	NaN	NaN	2.95
229	Ford Figo Diesel	Bangalore	2015	70436	Diesel	Manual	First	0.0	1498.0	99.0	NaN	3.60
...	...	...	...	...	...	...	...	...	...	...	...	...
5529	Hyundai Santro LP - Euro II	Chennai	2005	105000	Petrol	Manual	First	0.0	999.0	NaN	5.0	1.75
5875	Mercedes-Benz C-Class Progressive C 220d	Ahmedabad	2019	4000	Diesel	Automatic	First	0.0	1950.0	194.0	5.0	35.00
5943	Mahindra Jeep MM 540 DP	Chennai	2002	75000	Diesel	Manual	First	0.0	2112.0	NaN	6.0	1.70
5972	Hyundai Santro Xing GL	Mumbai	2008	65000	Petrol	Manual	Second	0.0	1086.0	62.0	5.0	1.39
6011	Skoda Superb 3.6 V6 FSI	Hyderabad	2009	53000	Petrol	Automatic	First	0.0	3597.0	262.6	5.0	4.75

64 rows × 12 columns

In [40]:
```
car_dp.loc[(car_dp['Mileage']==0)]
```

Out[40]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats
71	Hyundai Santro Xing GL	Ahmedabad	2013	63831	Petrol	Manual	First	0.0	1086.0	62.0	5.0
74	Hyundai Santro Xing XL	Bangalore	2007	47000	Petrol	Manual	Second	0.0	1086.0	NaN	5.0
158	Mercedes-Benz M-Class ML 350 4Matic	Bangalore	2012	37000	Diesel	Automatic	First	0.0	2987.0	165.0	5.0
186	Hyundai Santro Xing GL	Ahmedabad	2007	78000	Petrol	Manual	First	0.0	1086.0	62.0	5.0
420	Hyundai Santro GLS I - Euro II	Bangalore	2011	43189	Petrol	Manual	First	0.0	999.0	NaN	5.0
472	Mercedes-Benz M-Class ML 350 4Matic	Coimbatore	2016	22177	Diesel	Automatic	First	0.0	2987.0	165.0	5.0
614	Mahindra TUV 300 P4	Kolkata	2016	27000	Diesel	Manual	First	0.0	NaN	NaN	NaN
678	Hyundai Santro Xing XL	Jaipur	2007	85000	Petrol	Manual	Second	0.0	1086.0	NaN	5.0
838	Land Rover Freelander 2 TD4 SE	Mumbai	2011	87000	Diesel	Automatic	First	0.0	2179.0	115.0	5.0
938	Honda Jazz 2020 Petrol	Kochi	2019	11574	Petrol	Manual	First	0.0	1199.0	88.7	NaN
1207	Hyundai Santro Xing GL	Ahmedabad	2014	41000	Petrol	Manual	First	0.0	1086.0	62.0	5.0

In [41]:
```
Mileage_mean=car1['Mileage'].mean()
Mileage_meant=car_dp['Mileage'].mean()
```

In [42]:
```
car1['Mileage']=np.where(car1['Mileage']==0, Mileage_mean, car1['Mileage'])
car_dp['Mileage']=np.where(car_dp['Mileage']==0, Mileage_meant, car_dp['Mileage'])
```

In [43]:
```
car1.loc[(car1['Engine'].isna())]
```

Out[43]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price
194	Honda City 1.5 GXI	Ahmedabad	2007	60006	Petrol	Manual	First	18.181439	NaN	NaN	NaN	2.95
208	Maruti Swift 1.3 VXi	Kolkata	2010	42001	Petrol	Manual	First	16.100000	NaN	NaN	NaN	2.11
733	Maruti Swift 1.3 VXi	Chennai	2006	97800	Petrol	Manual	Third	16.100000	NaN	NaN	NaN	1.75
749	Land Rover Range Rover 3.0 D	Mumbai	2008	55001	Diesel	Automatic	Second	18.181439	NaN	NaN	NaN	26.50
1294	Honda City 1.3 DX	Delhi	2009	55005	Petrol	Manual	First	12.800000	NaN	NaN	NaN	3.20
1327	Maruti Swift 1.3 ZXI	Hyderabad	2015	50295	Petrol	Manual	First	16.100000	NaN	NaN	NaN	5.80
1385	Honda City 1.5 GXI	Pune	2004	115000	Petrol	Manual	Second	18.181439	NaN	NaN	NaN	1.50
1460	Land Rover Range Rover Sport 2005 2012 Sport	Coimbatore	2008	69078	Petrol	Manual	First	18.181439	NaN	NaN	NaN	40.88
2074	Maruti Swift 1.3 LXI	Pune	2011	24255	Petrol	Manual	First	16.100000	NaN	NaN	NaN	3.15
2096	Hyundai Santro LP zipPlus	Coimbatore	2004	52146	Petrol	Manual	First	18.181439	NaN	NaN	NaN	1.93
2264	Toyota Etios Liva V	Pune	2012	24500	Petrol	Manual	Second	18.300000	NaN	NaN	NaN	2.95
2325	Maruti Swift 1.3 VXI ABS	Pune	2015	67000	Petrol	Manual	First	16.100000	NaN	NaN	NaN	4.70
2335	Maruti Swift 1.3 VXi	Mumbai	2007	55000	Petrol	Manual	Second	16.100000	NaN	NaN	NaN	1.75
2530	BMW 5 Series 520d Sedan	Kochi	2014	64158	Diesel	Automatic	First	18.480000	NaN	NaN	NaN	17.89
2542	Hyundai Santro GLS II - Euro II	Bangalore	2011	65000	Petrol	Manual	Second	18.181439	NaN	NaN	NaN	3.15
2623	BMW 5 Series 520d Sedan	Pune	2012	95000	Diesel	Automatic	Second	18.480000	NaN	NaN	NaN	18.00
2668	Maruti Swift 1.3 VXi	Kolkata	2014	32986	Petrol	Manual	First	16.100000	NaN	NaN	NaN	4.24
2780	Hyundai Santro GLS II - Euro II	Pune	2009	100000	Petrol	Manual	First	18.181439	NaN	NaN	NaN	1.60
2842	Hyundai Santro GLS II - Euro II	Bangalore	2012	43000	Petrol	Manual	First	18.181439	NaN	NaN	NaN	3.25
3272	BMW 5 Series 520d Sedan	Mumbai	2008	81000	Diesel	Automatic	Second	18.480000	NaN	NaN	NaN	10.50
3520	BMW 5 Series 520d Sedan	Delhi	2012	90000	Diesel	Automatic	First	18.480000	NaN	NaN	NaN	14.50
3522	Hyundai Santro GLS II - Euro II	Kochi	2012	66400	Petrol	Manual	First	18.181439	NaN	NaN	NaN	2.66
3810	Honda CR-V AT With Sun Roof	Kolkata	2013	27000	Petrol	Automatic	First	14.000000	NaN	NaN	NaN	11.99
4011	Fiat Punto 1.3 Emotion	Pune	2011	45271	Diesel	Manual	First	20.300000	NaN	NaN	NaN	2.60
4152	Land Rover Range Rover 3.0 D	Mumbai	2003	75000	Diesel	Automatic	Second	18.181439	NaN	NaN	NaN	16.11
4229	Hyundai Santro Xing XG	Bangalore	2005	79000	Petrol	Manual	Second	17.000000	NaN	NaN	NaN	1.65
4577	BMW 5 Series 520d Sedan	Delhi	2012	72000	Diesel	Automatic	Third	18.480000	NaN	NaN	NaN	13.85
4604	Honda Jazz Select Edition	Pune	2011	98000	Petrol	Manual	First	16.700000	NaN	NaN	NaN	3.15
4697	Fiat Punto 1.2 Dynamic	Kochi	2017	17941	Petrol	Manual	First	15.700000	NaN	NaN	NaN	3.93
4712	Hyundai Santro Xing XG	Pune	2003	80000	Petrol	Manual	Second	17.000000	NaN	NaN	NaN	0.90
4952	Fiat Punto 1.4 Emotion	Kolkata	2010	47000	Petrol	Manual	First	14.600000	NaN	NaN	NaN	1.49
5015	Maruti Swift 1.3 VXi	Delhi	2006	63000	Petrol	Manual	First	16.100000	NaN	NaN	NaN	1.60
5185	Maruti Swift 1.3 LXI	Delhi	2012	52000	Petrol	Manual	First	16.100000	NaN	NaN	NaN	3.65
5270	Honda City 1.5 GXI	Bangalore	2002	53000	Petrol	Manual	Second	18.181439	NaN	NaN	NaN	1.85

In [44]:
```
car_dp.loc[(car_dp['Engine'].isna())]
```

Out[44]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats
23	Skoda Laura 1.8 TSI Ambition	Bangalore	2009	72000	Petrol	Manual	Second	17.500000	NaN	NaN	NaN
522	Toyota Etios Liva Diesel TRD Sportivo	Bangalore	2012	56600	Diesel	Manual	First	23.590000	NaN	NaN	NaN
525	Hyundai i20 new Sportz AT 1.4	Bangalore	2012	58000	Petrol	Automatic	Second	15.000000	NaN	NaN	NaN
614	Mahindra TUV 300 P4	Kolkata	2016	27000	Diesel	Manual	First	18.227715	NaN	NaN	NaN
624	BMW 5 Series 520d Sedan	Bangalore	2009	150000	Diesel	Automatic	Second	18.480000	NaN	NaN	NaN
632	Maruti Swift 1.3 VXi	Kolkata	2015	36009	Petrol	Manual	First	16.100000	NaN	NaN	NaN
658	Fiat Punto 1.4 Emotion	Jaipur	2010	65000	Petrol	Manual	Third	14.600000	NaN	NaN	NaN
666	Maruti Swift 1.3 VXi	Pune	2010	115000	Petrol	Manual	Second	16.100000	NaN	NaN	NaN
861	BMW 5 Series 520d Sedan	Chennai	2009	95000	Diesel	Automatic	Second	18.480000	NaN	NaN	NaN
883	Toyota Etios Liva V	Kochi	2012	59311	Petrol	Manual	First	18.300000	NaN	NaN	NaN

In [45]:
```
car1 = car1[car1.Engine.notnull()]
car_dp = car_dp[car_dp.Engine.notnull()]
```

In [46]:
```
car1.loc[(car1['Power'].isna())]
```

Out[46]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price
76	Ford Fiesta 1.4 SXi TDCi	Jaipur	2008	111111	Diesel	Manual	First	17.800000	1399.0	NaN	5.0	2.00
79	Hyundai Santro Xing XL	Hyderabad	2005	87591	Petrol	Manual	First	18.181439	1086.0	NaN	5.0	1.30
89	Hyundai Santro Xing XO	Hyderabad	2007	73745	Petrol	Manual	First	17.000000	1086.0	NaN	5.0	2.10
120	Hyundai Santro Xing XL eRLX Euro III	Mumbai	2005	102000	Petrol	Manual	Second	17.000000	1086.0	NaN	5.0	0.85
143	Hyundai Santro Xing XO eRLX Euro II	Kochi	2008	80759	Petrol	Manual	Third	17.000000	1086.0	NaN	5.0	1.67
...	...	...	...	...	...	...	...	...	...	...	...	...
5873	Hyundai Santro Xing XO eRLX Euro II	Pune	2006	47200	Petrol	Manual	Second	17.000000	1086.0	NaN	5.0	1.20
5893	Maruti Estilo LXI	Chennai	2008	51000	Petrol	Manual	Second	19.500000	1061.0	NaN	NaN	1.75
5925	Skoda Laura Classic 1.8 TSI	Pune	2010	85000	Petrol	Manual	First	17.500000	1798.0	NaN	5.0	2.85
5943	Mahindra Jeep MM 540 DP	Chennai	2002	75000	Diesel	Manual	First	18.181439	2112.0	NaN	6.0	1.70
5985	Toyota Etios GD	Delhi	2013	70000	Diesel	Manual	First	23.590000	1364.0	NaN	5.0	3.88

98 rows × 12 columns

In [47]:
```
car_dp.loc[(car_dp['Power'].isna())]
```

Out[47]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats
3	Toyota Etios Liva GD	Hyderabad	2012	139000	Diesel	Manual	First	23.590000	1364.0	NaN	5.0
74	Hyundai Santro Xing XL	Bangalore	2007	47000	Petrol	Manual	Second	18.227715	1086.0	NaN	5.0
122	Toyota Etios Liva GD	Jaipur	2012	121134	Diesel	Manual	First	23.590000	1364.0	NaN	5.0
143	Ford Fiesta 1.4 SXI Duratorq	Jaipur	2008	135000	Diesel	Manual	First	17.800000	1399.0	NaN	5.0
191	Tata Indica DLS	Chennai	2006	140000	Diesel	Manual	Second	13.500000	1405.0	NaN	5.0
259	Toyota Etios GD	Bangalore	2012	90000	Diesel	Manual	First	23.590000	1364.0	NaN	5.0
367	Nissan Teana 230jM	Coimbatore	2009	67038	Petrol	Automatic	First	9.100000	2349.0	NaN	5.0
409	Nissan Teana 230jM	Chennai	2008	63288	Petrol	Automatic	First	9.100000	2349.0	NaN	5.0
420	Hyundai Santro GLS I - Euro II	Bangalore	2011	43189	Petrol	Manual	First	18.227715	999.0	NaN	5.0
426	Maruti Swift VDI BSIV W ABS	Coimbatore	2016	69564	Diesel	Manual	First	17.800000	1248.0	NaN	5.0
572	Ford Fiesta 1.4 SXi TDCi	Pune	2009	134000	Diesel	Manual	First	17.800000	1399.0	NaN	5.0
606	Maruti Swift VDI BSIV	Kolkata	2012	72000	Diesel	Manual	First	17.800000	1248.0	NaN	5.0
678	Hyundai Santro Xing XL	Jaipur	2007	85000	Petrol	Manual	Second	18.227715	1086.0	NaN	5.0
704	Ford Fiesta 1.4 SXi TDCi	Kolkata	2009	39408	Diesel	Manual	First	17.800000	1399.0	NaN	5.0
872	Toyota Etios GD	Kolkata	2013	60000	Diesel	Manual	First	23.590000	1364.0	NaN	5.0
877	Toyota Etios Liva GD	Hyderabad	2013	86000	Diesel	Manual	First	23.590000	1364.0	NaN	5.0
928	Toyota Etios Liva G	Delhi	2012	77800	Petrol	Manual	First	18.300000	1197.0	NaN	5.0
1073	Hyundai Santro Xing XG AT eRLX Euro III	Coimbatore	2007	64168	Petrol	Automatic	First	17.000000	1086.0	NaN	5.0
1126	Toyota Etios Liva G	Kolkata	2012	37212	Petrol	Manual	First	18.300000	1197.0	NaN	5.0

In [48]:
```
Power_mean=car1['Power'].mean()
Power_meant=car_dp['Power'].mean()
```

In [49]:
```
car1['Power']=np.where(car1['Power'].isna(),Power_mean,car1['Power'])
car_dp['Power']=np.where(car_dp['Power'].isna(),Power_meant,car_dp['Power'])
```

In [50]:
```
car1.isnull().sum()
```

Out[50]:
```
Name                0
Location            0
Year                0
Kilometers_Driven   0
Fuel_Type           0
Transmission        0
Owner_Type          0
Mileage             0
Engine              0
Power               0
Seats               6
Price               0
dtype: int64
```

In [51]:
```
car_dp.isnull().sum()
```

Out[51]:
```
Name                0
Location            0
Year                0
Kilometers_Driven   0
Fuel_Type           0
Transmission        0
Owner_Type          0
Mileage             0
Engine              0
Power               0
Seats               1
dtype: int64
```

In [52]:
```
car1=car1.dropna(axis=0)
car1.head()
```

Out[52]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price
0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First	26.60	998.0	58.16	5.0	1.75
1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67	1582.0	126.20	5.0	12.50
2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	18.20	1199.0	88.70	5.0	4.50
3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77	1248.0	88.76	7.0	6.00
4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.20	1968.0	140.80	5.0	17.74

In [53]:
```
car_dp=car_dp.dropna(axis=0)
car_dp.head()
```

Out[53]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats
0	Maruti Alto K10 LXI CNG	Delhi	2014	40929	CNG	Manual	First	32.26	998.0	58.200000	4.0
1	Maruti Alto 800 2016-2019 LXI	Coimbatore	2013	54493	Petrol	Manual	Second	24.70	796.0	47.300000	5.0
2	Toyota Innova Crysta Touring Sport 2.4 MT	Mumbai	2017	34000	Diesel	Manual	First	13.68	2393.0	147.800000	7.0
3	Toyota Etios Liva GD	Hyderabad	2012	139000	Diesel	Manual	First	23.59	1364.0	110.321992	5.0
4	Hyundai i20 Magna	Mumbai	2014	29000	Petrol	Manual	First	18.50	1197.0	82.850000	5.0

In [54]:
```
car1['Seats']=car1['Seats'].astype(int)
car_dp['Seats']=car_dp['Seats'].astype(int)
```

In [55]:
```
car1.dtypes
```

Out[55]:
```
Name                object
Location            object
Year                 int64
Kilometers_Driven    int64
Fuel_Type           object
Transmission        object
Owner_Type          object
Mileage            float64
Engine             float64
Power              float64
Seats                int32
Price              float64
dtype: object
```

In [56]:
```
car_dp.dtypes
```

Out[56]:
```
Name                object
Location            object
Year                 int64
Kilometers_Driven    int64
Fuel_Type           object
Transmission        object
Owner_Type          object
Mileage            float64
Engine             float64
Power              float64
Seats                int32
dtype: object
```

In [57]:
```
car1.isnull().sum()
```

Out[57]:
```
Name                0
Location            0
Year                0
Kilometers_Driven   0
Fuel_Type           0
Transmission        0
Owner_Type          0
Mileage             0
Engine              0
Power               0
Seats               0
Price               0
dtype: int64
```

In [58]:
```
car_dp.isnull().sum()
```

Out[58]:
```
Name                0
Location            0
Year                0
Kilometers_Driven   0
Fuel_Type           0
Transmission        0
Owner_Type          0
Mileage             0
Engine              0
Power               0
Seats               0
dtype: int64
```

Shape after Preprocessing:_

In [59]:
```
car1.shape
```

Out[59]:
```
(5916, 12)
```

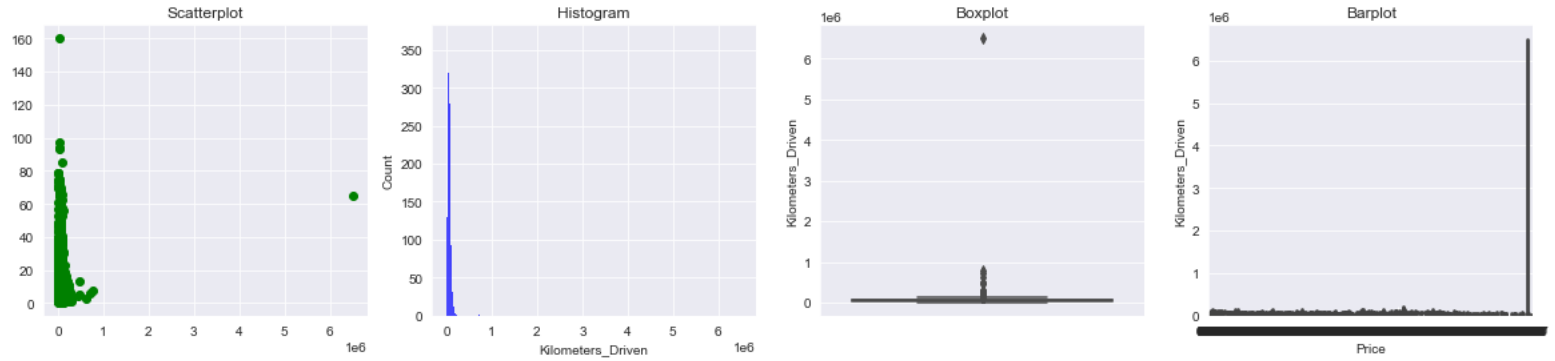
In [60]:
```
car_dp.shape
```

Out[60]:
```
(1210, 11)
```

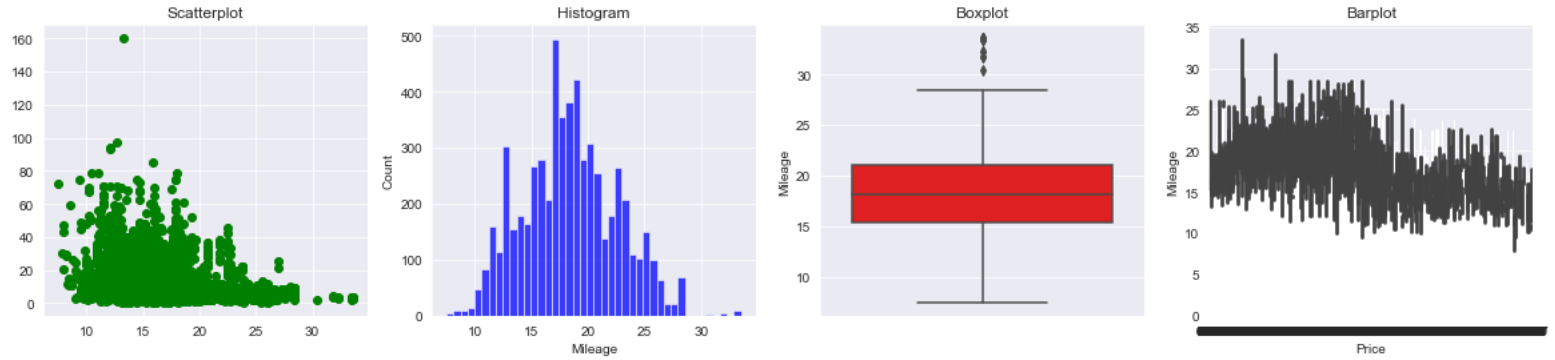
EDA:

In [61]:
```
def EDA_plots(data, feature, target):
    sns.set_style("darkgrid")
    plt.figure(figsize=(20, 4))

    plt.subplot(1, 4, 1)
    plt.scatter(data[feature],data[target],color = 'g')
    plt.title('Scatterplot')

    plt.subplot(1, 4, 2)
    sns.histplot(data[feature], color = 'b')
    plt.title('Histogram')

    plt.subplot(1, 4, 3)
    sns.boxplot(y=data[feature],color = 'r')
    plt.title('Boxplot')

    plt.subplot(1, 4, 4)
    sns.barplot(x = target, y = feature, data = data)
    plt.title('Barplot')

    plt.show()
```

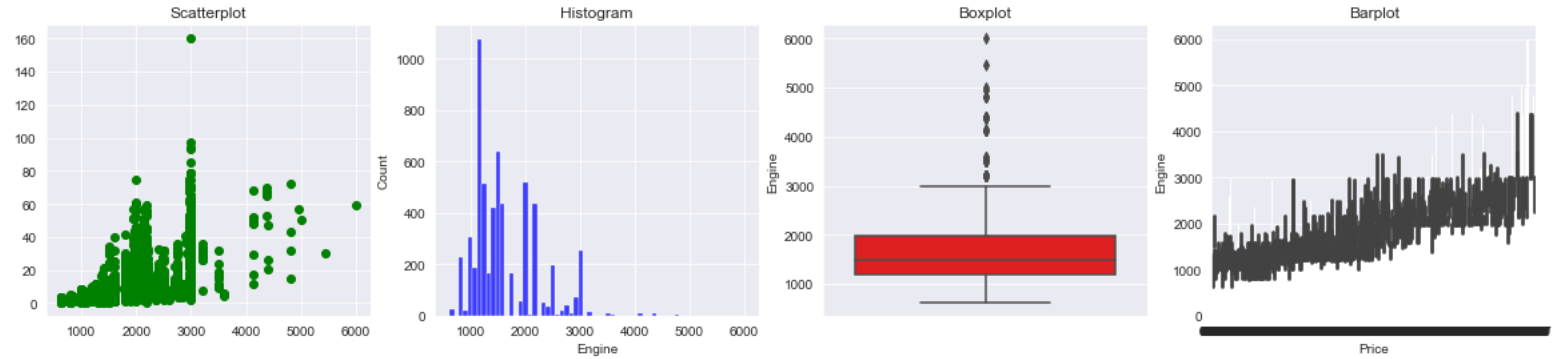
In [62]:
```
EDA_plots(car1, 'Kilometers_Driven', 'Price')
```



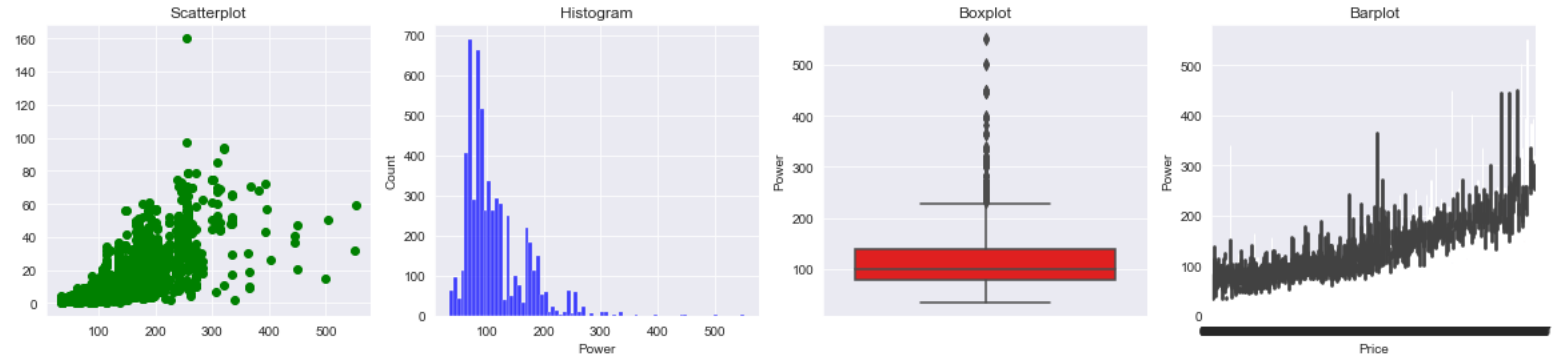
In [63]:
```
EDA_plots(car1, 'Mileage', 'Price')
```



In [64]:
```
EDA_plots(car1, 'Engine', 'Price')
```



In [65]:
```
EDA_plots(car1, 'Power', 'Price')
```



Shape with Outliers:_

In [66]:
```
car1.shape
```

Out[66]: (5916, 12)

In [67]:
```
car_dp.shape
```

Out[67]: (1210, 11)

Handle Outliers:_

In [68]:
```
q3k = car1['Kilometers_Driven'].quantile(0.75)
q1k = car1['Kilometers_Driven'].quantile(0.25)

iqrk = q3k - q1k
iqrk
```

Out[68]: 38636.75

In [69]:
```
def outlier_km(car1):
    km = car1['Kilometers_Driven']
    if km > (q3k + 1.5 * iqrk) or km < (q1k - 1.5 * iqrk):
        return True
    else:
        return False
```

In [70]:
```
q3kt = car_dp['Kilometers_Driven'].quantile(0.75)
q1kt = car_dp['Kilometers_Driven'].quantile(0.25)

iqrkt = q3kt - q1kt
iqrkt
```

Out[70]: 40000.0

In [71]:
```
def outlier_kmt(car_dp):
    km = car_dp['Kilometers_Driven']
    if km > (q3kt + 1.5 * iqrkt) or km < (q1kt - 1.5 * iqrkt):
        return True
    else:
        return False
```

In [72]:
```
q3p = car1['Power'].quantile(0.75)
q1p = car1['Power'].quantile(0.25)

iqrp = q3p - q1p
iqrp
```

Out[72]: 60.099999999999994

In [73]:
```
iqrp = 60.1
iqrp
```

Out[73]: 60.1

In [74]:
```
def outlier_pw(car1):
    pw=car1['Power']
    if pw > (q3p + 1.5 * iqrp) or pw < (q1p - 1.5 * iqrp):
        return True
    else:
        return False
```

In [75]:
```
q3pt = car1['Power'].quantile(0.75)
q1pt = car1['Power'].quantile(0.25)

iqrpt = q3pt - q1pt
iqrpt
```

Out[75]: 60.099999999999994

In [76]:
```
def outlier_pwt(car_dp):
    pw=car_dp['Power']
    if pw > (q3pt + 1.5 * iqrp) or pw < (q1pt - 1.5 * iqrp):
        return True
    else:
        return False
```

In [77]:
```
q3e = car1['Engine'].quantile(0.75)
q1e = car1['Engine'].quantile(0.25)

iqre = q3e - q1e
iqre
```

Out[77]: 770.0

In [78]:
```
def outlier_eng(car1):
    eng=car1['Engine']
    if eng > (q3e + 1.5 * iqre) or eng < (q1e - 1.5 * iqre):
        return True
    else:
        return False
```

In [79]:
```
q3et = car_dp['Engine'].quantile(0.75)
q1et = car_dp['Engine'].quantile(0.25)

iqret = q3et - q1et
iqret
```

Out[79]: 770.0

In [80]:
```
def outlier_engt(car_dp):
    eng=car_dp['Engine']
    if eng > (q3et + 1.5 * iqret) or eng < (q1et - 1.5 * iqret):
        return True
    else:
        return False
```

In [81]:
```
q3m = car1['Mileage'].quantile(0.75)
q1m = car1['Mileage'].quantile(0.25)

iqrm = q3m - q1m
iqrm
```

Out[81]: 5.700000000000001

In [82]:
```
iqrm = 5.7
iqrm
```

Out[82]: 5.7

In [83]:
```
def outlier_mile(car1):
    mile=car1['Mileage']
    if mile > (q3m + 1.5 * iqrm) or mile < (q1m - 1.5 * iqrm):
        return True
    else:
        return False
```

In [84]:
```
q3mt = car_dp['Mileage'].quantile(0.75)
q1mt = car_dp['Mileage'].quantile(0.25)

iqrmt = q3mt - q1mt
iqrmt
```

Out[84]: 5.700000000000001

In [85]:
```
def outlier_milet(car_dp):
    mile=car_dp['Mileage']
    if mile > (q3mt + 1.5 * iqrm) or mile < (q1mt - 1.5 * iqrm):
        return True
    else:
        return False
```

In [86]:
```
car1['outlier_KM'] = car1.apply(outlier_km, axis = 1)
car1['outlier_Power'] = car1.apply(outlier_pw, axis = 1)
car1['outlier_Engine'] = car1.apply(outlier_eng, axis = 1)
car1['outlier_Mileage'] = car1.apply(outlier_mile, axis = 1)

car1.head()
```

Out[86]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price	outlier_KM	outlier_Power	outlier_Engine	outlier_Mileage
0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First	26.60	998.0	58.16	5	1.75	False	False	False	False
1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67	1582.0	126.20	5	12.50	False	False	False	False
2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	18.20	1199.0	88.70	5	4.50	False	False	False	False
3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77	1248.0	88.76	7	6.00	False	False	False	False
4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.20	1968.0	140.80	5	17.74	False	False	False	False

In [87]:
```
car_dp['outlier_KM'] = car_dp.apply(outlier_kmt, axis = 1)
car_dp['outlier_Power'] = car_dp.apply(outlier_pwt, axis = 1)
car_dp['outlier_Engine'] = car_dp.apply(outlier_engt, axis = 1)
car_dp['outlier_Mileage'] = car_dp.apply(outlier_milet, axis = 1)

car_dp.head()
```

Out[87]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	outlier_KM	outlier_Power	outlier_Engine	outlier_Mileage
0	Maruti Alto K10 LXI CNG	Delhi	2014	40929	CNG	Manual	First	32.26	998.0	58.200000	4	False	False	False	True
1	Maruti Alto 800 2016-2019 LXI	Coimbatore	2013	54493	Petrol	Manual	Second	24.70	796.0	47.300000	5	False	False	False	False
2	Toyota Innova Crysta Touring Sport 2.4 MT	Mumbai	2017	34000	Diesel	Manual	First	13.68	2393.0	147.800000	7	False	False	False	False
3	Toyota Etios Liva GD	Hyderabad	2012	139000	Diesel	Manual	First	23.59	1364.0	110.321992	5	True	False	False	False
4	Hyundai i20 Magna	Mumbai	2014	29000	Petrol	Manual	First	18.50	1197.0	82.850000	5	False	False	False	False

In [88]:
```
def outliers(car1):
    if (car1['outlier_KM']==False) and (car1['outlier_Power']==False) and (car1['outlier_Engine']==False) and (car1['outlier_Mileage']==False):
        return False
    else:
        return True
```

In [89]:
```
def outlierst(car_dp):
    if (car_dp['outlier_KM']==False) and (car_dp['outlier_Power']==False) and (car_dp['outlier_Engine']==False) and (car_dp['outlier_Mileage']==False):
        return False
    else:
        return True
```

In [90]:
```
car1['Is Outlier'] = car1.apply(outliers, axis = 1)
car_dp['Is Outlier'] = car_dp.apply(outlierst, axis = 1)
```

Outlier Removal:_

In [91]:
```
car2 = car1.loc[car1['Is Outlier'] == False]
car2
```

Out[91]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price	outlier_KM	outlier_Power	outlier_Engine	outlier_Mileage	Is Outlier
0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First	26.60	998.0	58.16	5	1.75	False	False	False	False	False
1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67	1582.0	126.20	5	12.50	False	False	False	False	False
2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	18.20	1199.0	88.70	5	4.50	False	False	False	False	False
3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77	1248.0	88.76	7	6.00	False	False	False	False	False
4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.20	1968.0	140.80	5	17.74	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
6014	Maruti Swift VDI	Delhi	2014	27365	Diesel	Manual	First	28.40	1248.0	74.00	5	4.75	False	False	False	False	False
6015	Hyundai Xcent 1.1 CRDi S	Jaipur	2015	100000	Diesel	Manual	First	24.40	1120.0	71.00	5	4.00	False	False	False	False	False
6016	Mahindra Xylo D4 BSIV	Jaipur	2012	55000	Diesel	Manual	Second	14.00	2498.0	112.00	8	2.90	False	False	False	False	False
6017	Maruti Wagon R VXI	Kolkata	2013	46000	Petrol	Manual	First	18.90	998.0	67.10	5	2.65	False	False	False	False	False
6018	Chevrolet Beat Diesel	Hyderabad	2011	47000	Diesel	Manual	First	25.44	936.0	57.60	5	2.50	False	False	False	False	False

5463 rows × 17 columns

In [92]:
```
car_dp = car_dp.loc[car_dp['Is Outlier'] == False]
car_dp
```

Out[92]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	outlier_KM	outlier_Power	outlier_Engine	outlier_Mileage	Is Outlier
1	Maruti Alto 800 2016-2019 LXI	Coimbatore	2013	54493	Petrol	Manual	Second	24.70	796.0	47.30	5	False	False	False	False	False
2	Toyota Innova Crysta Touring Sport 2.4 MT	Mumbai	2017	34000	Diesel	Manual	First	13.68	2393.0	147.80	7	False	False	False	False	False
4	Hyundai i20 Magna	Mumbai	2014	29000	Petrol	Manual	First	18.50	1197.0	82.85	5	False	False	False	False	False
5	Mahindra XUV500 W8 2WD	Coimbatore	2016	85609	Diesel	Manual	Second	16.00	2179.0	140.00	7	False	False	False	False	False
6	Toyota Fortuner 4x2 AT TRD Sportivo	Pune	2015	59000	Diesel	Automatic	First	12.55	2982.0	168.70	7	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1229	Volkswagen Vento Diesel Trendline	Hyderabad	2011	89411	Diesel	Manual	First	20.54	1598.0	103.60	5	False	False	False	False	False
1230	Volkswagen Polo GT TSI	Mumbai	2015	59000	Petrol	Automatic	First	17.21	1197.0	103.60	5	False	False	False	False	False
1231	Nissan Micra Diesel XV	Kolkata	2012	28000	Diesel	Manual	First	23.08	1461.0	63.10	5	False	False	False	False	False
1232	Volkswagen Polo GT TSI	Pune	2013	52262	Petrol	Automatic	Third	17.20	1197.0	103.60	5	False	False	False	False	False
1233	Mercedes-Benz E-Class 2009-2013 E 220 CDI Avan...	Kochi	2014	72443	Diesel	Automatic	First	10.00	2148.0	170.00	5	False	False	False	False	False

1128 rows × 16 columns

In [93]:
```
dlst = ['outlier_KM','outlier_Power','outlier_Engine','outlier_Mileage','Is Outlier']
```

In [94]:
```
def outlier_rm(data):
    for i in dlst:
        del data[i]
```

In [95]:
```
outlier_rm(car2)
outlier_rm(car_dp)
```

Shape after Outlier Removal:_

In [96]:
```
car2.shape
```

Out[96]: (5463, 12)

In [97]:
```
car_dp.shape
```

Out[97]: (1128, 11)

In [98]:
```
car2.head()
```

Out[98]:

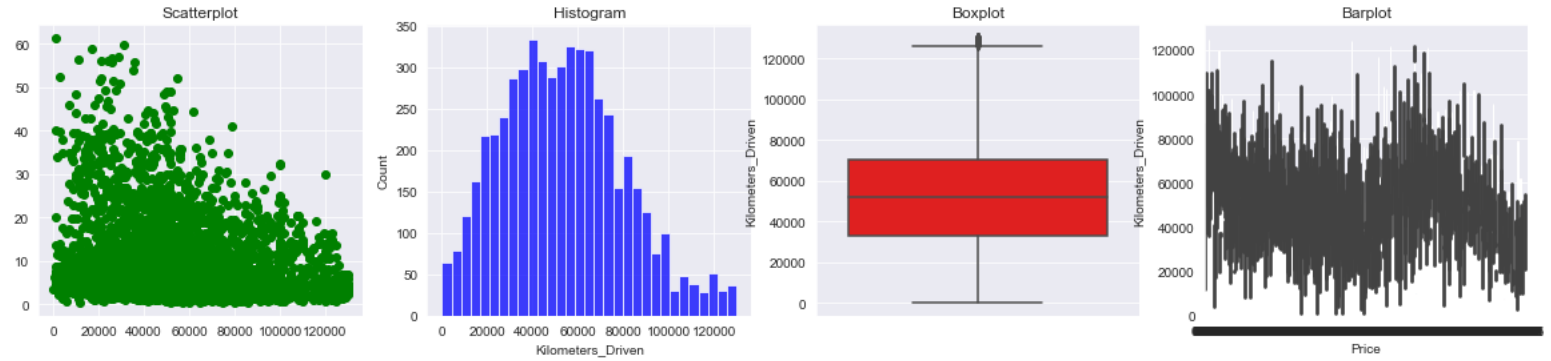
	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price
0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First	26.60	998.0	58.16	5	1.75
1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67	1582.0	126.20	5	12.50
2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	18.20	1199.0	88.70	5	4.50
3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77	1248.0	88.76	7	6.00
4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.20	1968.0	140.80	5	17.74

In [99]:
```
car_dp.head()
```

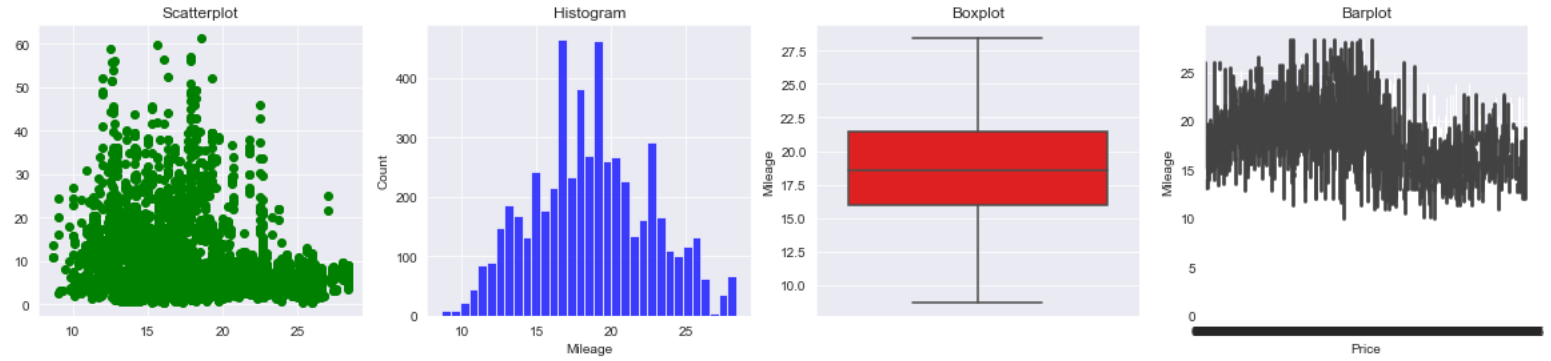
Out[99]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats
1	Maruti Alto 800 2016-2019 LXI	Coimbatore	2013	54493	Petrol	Manual	Second	24.70	796.0	47.30	5
2	Toyota Innova Crysta Touring Sport 2.4 MT	Mumbai	2017	34000	Diesel	Manual	First	13.68	2393.0	147.80	7
4	Hyundai i20 Magna	Mumbai	2014	29000	Petrol	Manual	First	18.50	1197.0	82.85	5
5	Mahindra XUV500 W8 2WD	Coimbatore	2016	85609	Diesel	Manual	Second	16.00	2179.0	140.00	7
6	Toyota Fortuner 4x2 AT TRD Sportivo	Pune	2015	59000	Diesel	Automatic	First	12.55	2982.0	168.70	7

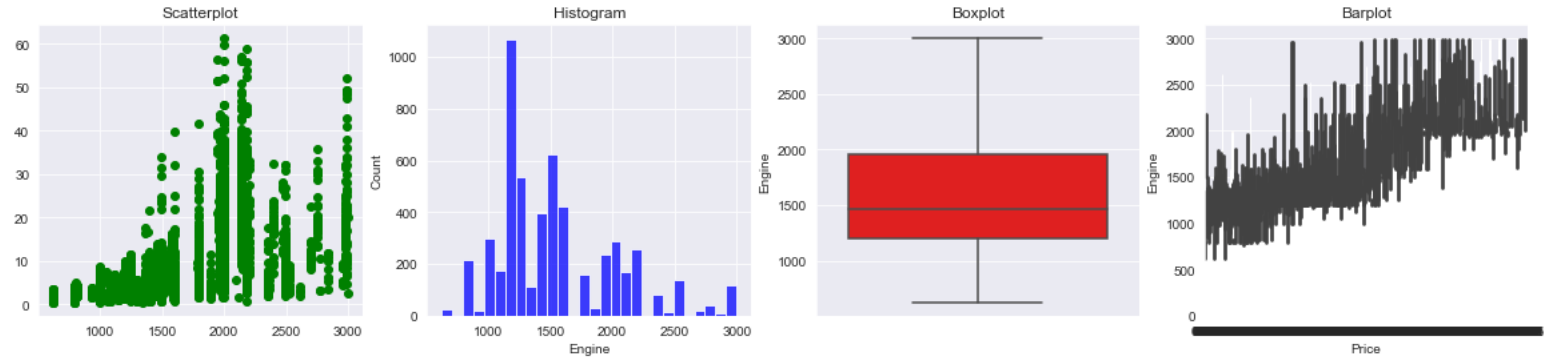
In [100]:
```
EDA_plots(car2, 'Kilometers_Driven', 'Price')
```



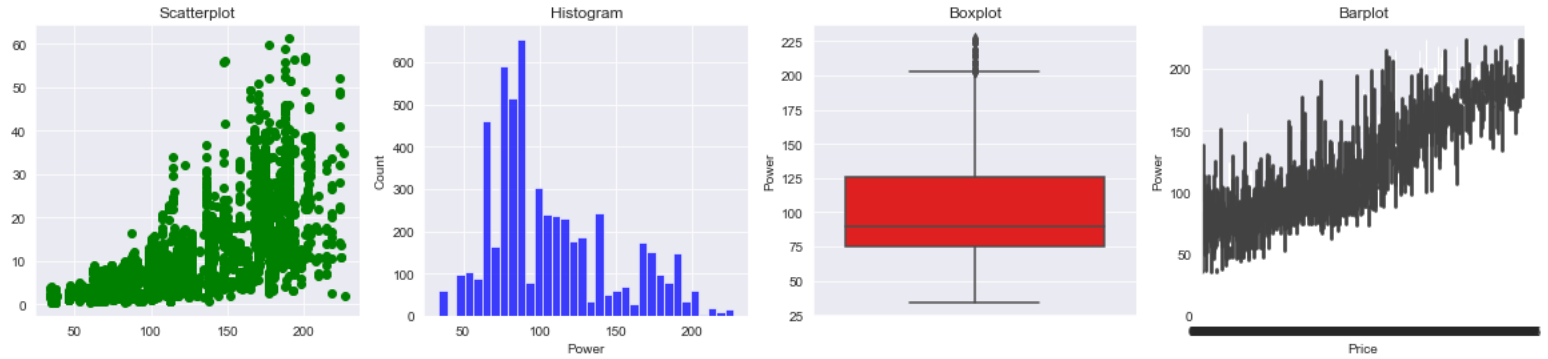
In [101]:
```
EDA_plots(car2, 'Mileage', 'Price')
```



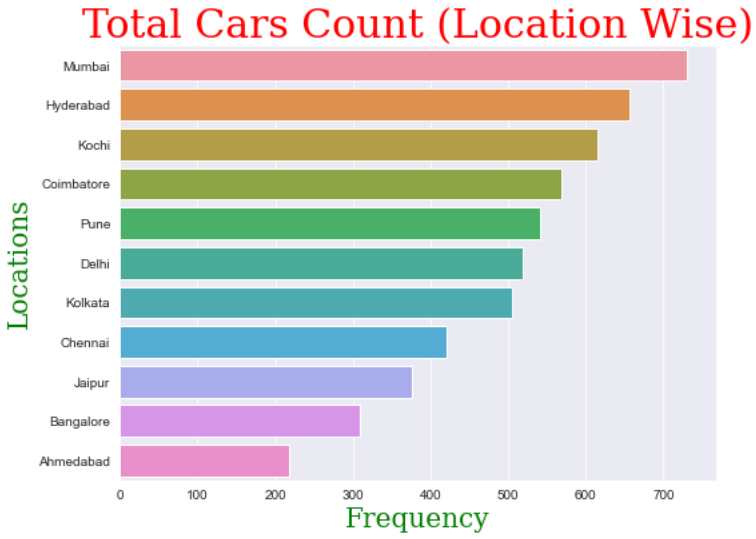
In [102]:
```
EDA_plots(car2, 'Engine', 'Price')
```



In [103]:
```
EDA_plots(car2, 'Power', 'Price')
```

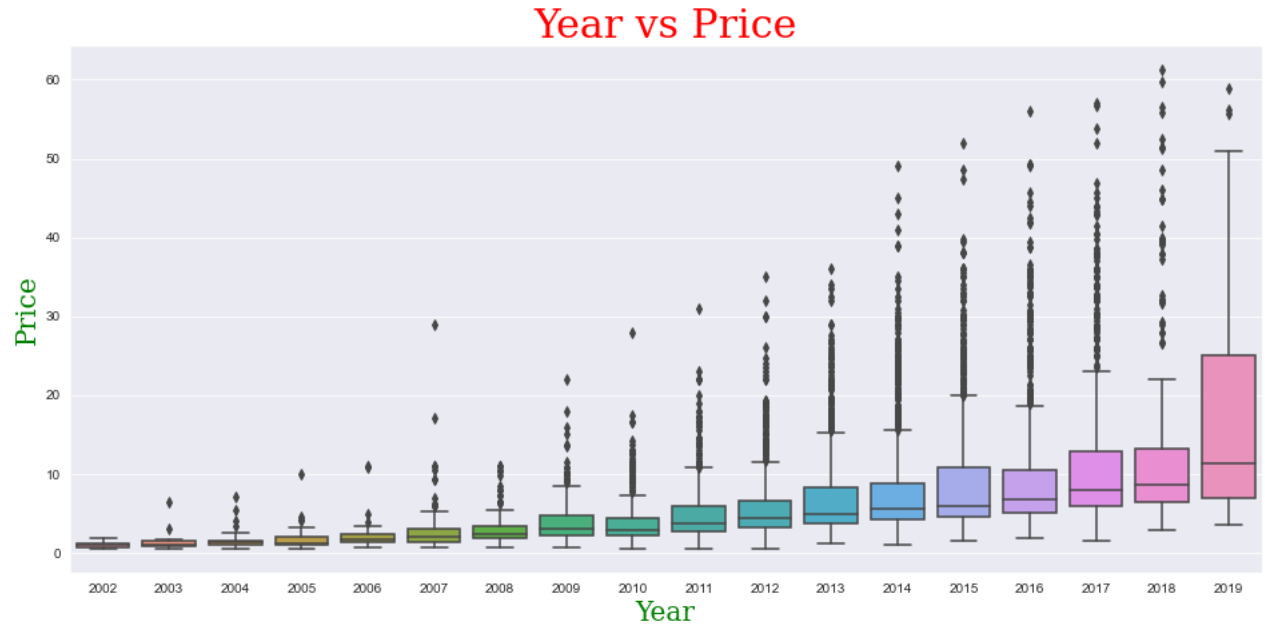


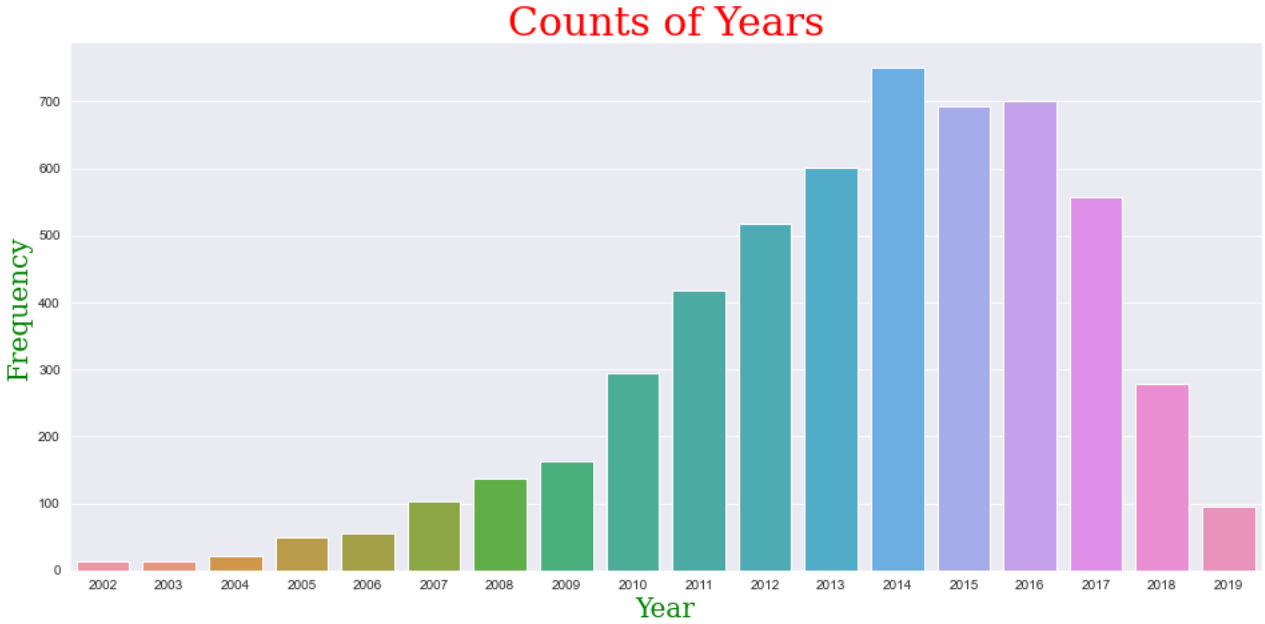
In [104]:
```
plt.figure(figsize=(8,6))

font1={'family':'serif','color':'red','size':30}
font2={'family':'serif','color':'g','size':20}

loc_cars = car2['Location'].value_counts()
sns.barplot(x=loc_cars, y=loc_cars.index, data=car2)

plt.xlabel("Frequency",fontdict=font2)
plt.ylabel("Locations",fontdict=font2)
plt.title("Total Cars Count (Location Wise)",fontdict=font1)

plt.show()
```



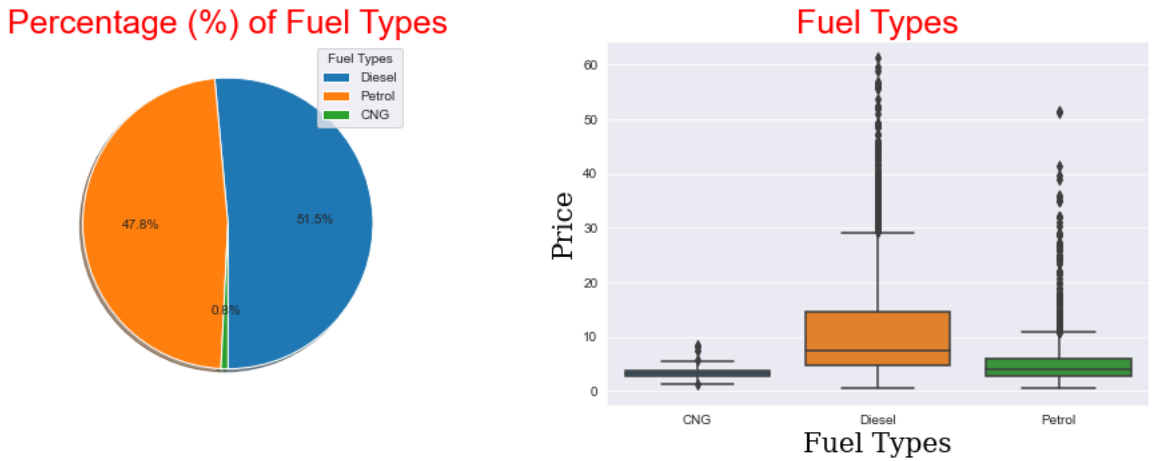
In [105]:

```
plt.figure(figsize=(16,16))

font1={'family':'serif','color':'red','size':30}
font2={'family':'serif','color':'g','size':20}

plt.subplot(2,1,1)
sns.boxplot(data=car2,x="Year",y="Price")
plt.xlabel("Year", fontdict=font2)
plt.ylabel("Price", fontdict=font2)
plt.title("Year vs Price", fontdict=font1)


plt.subplot(2,1,2)
sns.countplot(data=car2,x="Year")
plt.xlabel("Year", fontdict=font2)
plt.ylabel("Frequency", fontdict=font2)
plt.title('Counts of Years', fontdict=font1)

plt.show()
```

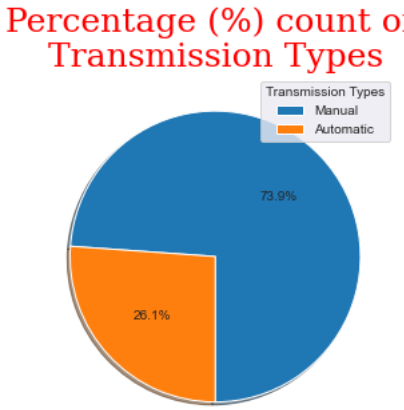

Year vs Price
Price
Year
0
10
20
30
40
50
60
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019


Counts of Years
Frequency
Year
0
100
200
300
400
500
600
700
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019

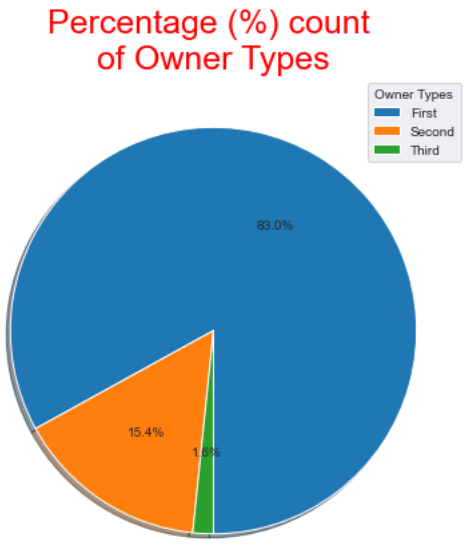
In [106]:
```
plt.figure(figsize=(16,5))

font2={'family':'serif','color':'k','size':20}

plt.subplot(1,2,1)
labels =car2['Fuel_Type'].value_counts(sort = True).index
sizes = car2['Fuel_Type'].value_counts(sort = True)
plt.pie(sizes, autopct='%1.1f%%', shadow=True, startangle=270)
plt.title('Percentage (%) of Fuel Types',size = 25,c="r")
plt.legend(title='Fuel Types', labels=labels)

plt.subplot(1,2,2)
sns.boxplot(data=car2, x="Fuel_Type", y="Price")
plt.xlabel("Fuel Types", fontdict=font2)
plt.ylabel("Price", fontdict=font2)
plt.title("Fuel Types", size=25, c="r")

plt.show()
```

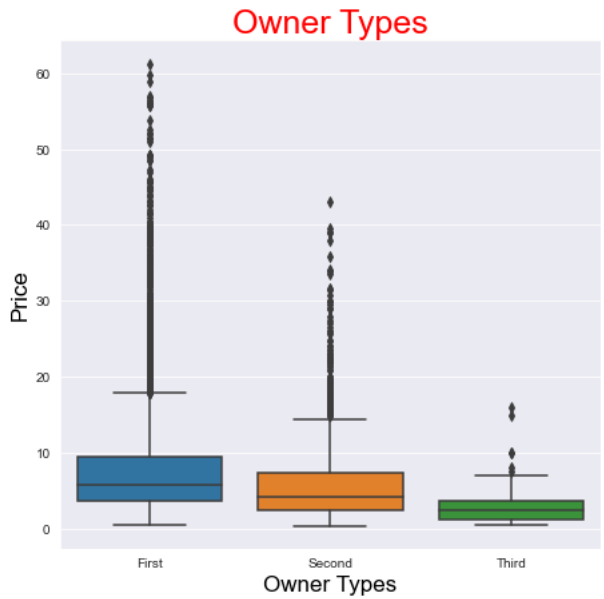


In [107]:
```
plt.figure(figsize=(5,5))

font1={'family':'serif','color':'red','size':25}

labels =car2['Transmission'].value_counts(sort = True).index
sizes = car2['Transmission'].value_counts(sort = True)

plt.pie(sizes, autopct='%1.1f%%', shadow=True, startangle=270)
plt.title('Percentage (%) count of \nTransmission Types', fontdict=font1)
plt.legend(title='Transmission Types', labels=labels)

plt.show()
```

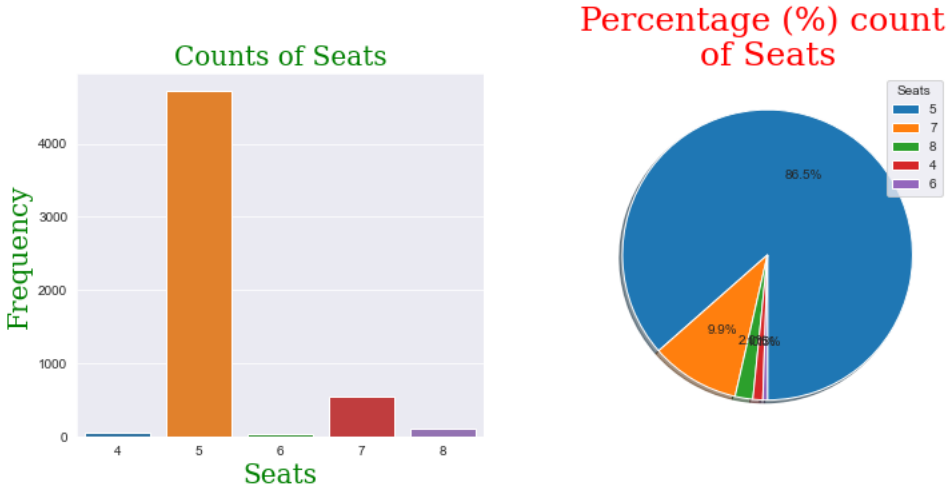


In [108]:
```
plt.figure(figsize=(16,7))

plt.subplot(1,2,1)
labels =car2['Owner_Type'].value_counts(sort = True).index
sizes = car2['Owner_Type'].value_counts(sort = True)
plt.pie(sizes, autopct='%1.1f%%', shadow=True, startangle=270)
plt.title('Percentage (%) count \nof Owner Types',size = 25,c="r")
plt.legend(title='Owner Types', labels=labels)

plt.subplot(1,2,2)
sns.boxplot(data=car2,x="Owner_Type",y="Price")
plt.xlabel("Owner Types",size=17,c="k")
plt.ylabel("Price",size=17,c="k")
plt.title("Owner Types",size=25,c="red")

plt.show()
```

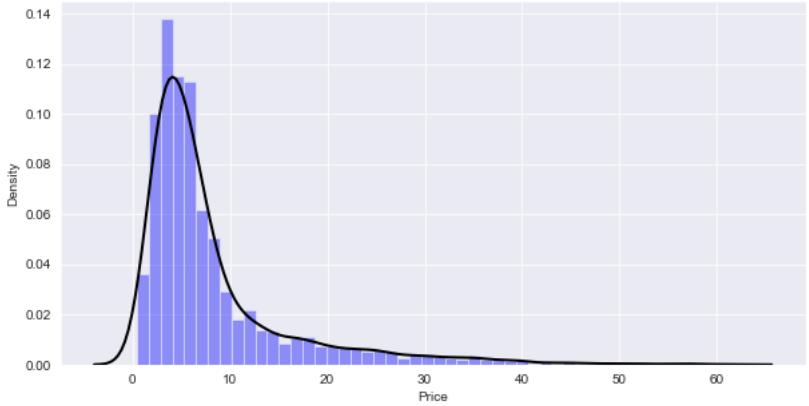




In [109]:
```
plt.figure(figsize=(12,5))

font1={'family':'serif','color':'red','size':25}
font2={'family':'serif','color':'g','size':20}

plt.subplot(1,2,1)
sns.countplot(data=car2,x="Seats")
plt.xlabel("Seats", fontdict=font2)
plt.ylabel("Frequency", fontdict=font2)
plt.title('Counts of Seats', fontdict=font2)

plt.subplot(1,2,2)
labels = car2['Seats'].value_counts(sort = True).index
sizes = car2['Seats'].value_counts(sort = True)
plt.pie(sizes, autopct='%1.1f%%', shadow=True, startangle=270)
plt.title('Percentage (%) count \nof Seats', fontdict=font1)
plt.legend(title='Seats', labels=labels)

plt.show()
```



In [110]:
```
plt.figure(figsize=(10, 5))
sns.distplot(car2['Price'],color='b',kde_kws={'linewidth':2,'color':'k'})
plt.show()
```



Shape after EDA:_

In [111]:
```
car2.shape
```

Out[111]: (5463, 12)

In [112]:
```
car_dp.shape
```

Out[112]: (1128, 11)

Feature Engineering:

Encoding of Categorical features:_

In [113]:
```
print(car2['Fuel_Type'].unique())
print(car2['Transmission'].unique())
print(car2['Owner_Type'].unique())
print(car2['Location'].unique())
```

```
['CNG' 'Diesel' 'Petrol']
['Manual' 'Automatic']
['First' 'Second' 'Third']
['Mumbai' 'Pune' 'Chennai' 'Coimbatore' 'Jaipur' 'Kochi' 'Kolkata' 'Delhi'
 'Bangalore' 'Hyderabad' 'Ahmedabad']
```

In [114]:
```
print(car_dp['Fuel_Type'].unique())
print(car_dp['Transmission'].unique())
print(car_dp['Owner_Type'].unique())
print(car_dp['Location'].unique())
```

```
['Petrol' 'Diesel' 'CNG']
['Manual' 'Automatic']
['Second' 'First' 'Third']
['Coimbatore' 'Mumbai' 'Pune' 'Jaipur' 'Chennai' 'Hyderabad' 'Kochi'
 'Delhi' 'Bangalore' 'Kolkata' 'Ahmedabad']
```

In [115]:
```
fuel_map = {'Petrol':1, 'Diesel' : 2, 'CNG' : 3}
trans_map = {'Manual':1, 'Automatic' : 2}
owner_map = {'First':1, 'Second':2, 'Third':3}
loc_map = {'Mumbai':1, 'Pune':2, 'Chennai':3, 'Coimbatore':4, 'Jaipur':5, 'Kochi':6, 'Kolkata':7, 'Delhi':8,
           'Bangalore':9, 'Hyderabad':10, 'Ahmedabad':11}
```

In [116]:
```
car2['Fuel'] = car2.Fuel_Type.map(fuel_map)
car2['Transmission_Type'] = car2.Transmission.map(trans_map)
car2['Owner'] = car2.Owner_Type.map(owner_map)
car2['Locations'] = car2.Location.map(loc_map)
```

In [117]:
```
car3=car2.drop(['Fuel_Type','Transmission','Owner_Type','Location'], axis=1)
car3.head()
```

Out[117]:

	Name	Year	Kilometers_Driven	Mileage	Engine	Power	Seats	Price	Fuel	Transmission_Type	Owner	Locations
0	Maruti Wagon R LXI CNG	2010	72000	26.60	998.0	58.16	5	1.75	3	1	1	1
1	Hyundai Creta 1.6 CRDi SX Option	2015	41000	19.67	1582.0	126.20	5	12.50	2	1	1	2
2	Honda Jazz V	2011	46000	18.20	1199.0	88.70	5	4.50	1	1	1	3
3	Maruti Ertiga VDI	2012	87000	20.77	1248.0	88.76	7	6.00	2	1	1	3
4	Audi A4 New 2.0 TDI Multitronic	2013	40670	15.20	1968.0	140.80	5	17.74	2	2	2	4

In [118]:
```
car_dp['Fuel'] = car_dp.Fuel_Type.map(fuel_map)
car_dp['Transmission_Type'] = car_dp.Transmission.map(trans_map)
car_dp['Owner'] = car_dp.Owner_Type.map(owner_map)
car_dp['Locations'] = car_dp.Location.map(loc_map)
```

In [119]:
```
car_dp=car_dp.drop(['Fuel_Type','Transmission','Owner_Type','Location'], axis=1)
car_dp.head()
```

Out[119]:

	Name	Year	Kilometers_Driven	Mileage	Engine	Power	Seats	Fuel	Transmission_Type	Owner	Locations
1	Maruti Alto 800 2016-2019 LXI	2013	54493	24.70	796.0	47.30	5	1	1	2	4
2	Toyota Innova Crysta Touring Sport 2.4 MT	2017	34000	13.68	2393.0	147.80	7	2	1	1	1
4	Hyundai i20 Magna	2014	29000	18.50	1197.0	82.85	5	1	1	1	1
5	Mahindra XUV500 W8 2WD	2016	85609	16.00	2179.0	140.00	7	2	1	2	4
6	Toyota Fortuner 4x2 AT TRD Sportivo	2015	59000	12.55	2982.0	168.70	7	2	2	1	2

In [120]:
```
car3.isna().sum()
```

Out[120]:
```
Name                0
Year                0
Kilometers_Driven   0
Mileage             0
Engine              0
Power               0
Seats               0
Price               0
Fuel                0
Transmission_Type   0
Owner               0
Locations           0
dtype: int64
```

In [121]:
```
car_dp.isna().sum()
```

Out[121]:
```
Name                0
Year                0
Kilometers_Driven   0
Mileage             0
Engine              0
Power               0
Seats               0
Fuel                0
Transmission_Type   0
Owner               0
Locations           0
dtype: int64
```

In [122]:
```
car3['Current_Year'] = 2023
car3['No_Year'] = car3['Current_Year'] - car3['Year']
```

In [123]:
```
car_dp['Current_Year'] = 2023
car_dp['No_Year'] = car_dp['Current_Year'] - car_dp['Year']
```

In [124]:
```
car3 = car3.drop(['Year','Current_Year'],axis = 1)
car3.head()
```

Out[124]:

	Name	Kilometers_Driven	Mileage	Engine	Power	Seats	Price	Fuel	Transmission_Type	Owner	Locations	No_Year
0	Maruti Wagon R LXI CNG	72000	26.60	998.0	58.16	5	1.75	3	1	1	1	13
1	Hyundai Creta 1.6 CRDi SX Option	41000	19.67	1582.0	126.20	5	12.50	2	1	1	2	8
2	Honda Jazz V	46000	18.20	1199.0	88.70	5	4.50	1	1	1	3	12
3	Maruti Ertiga VDI	87000	20.77	1248.0	88.76	7	6.00	2	1	1	3	11
4	Audi A4 New 2.0 TDI Multitronic	40670	15.20	1968.0	140.80	5	17.74	2	2	2	4	10

In [125]:
```
car_dp = car_dp.drop(['Year','Current_Year'],axis = 1)
car_dp.head()
```

Out[125]:

	Name	Kilometers_Driven	Mileage	Engine	Power	Seats	Fuel	Transmission_Type	Owner	Locations	No_Year
1	Maruti Alto 800 2016-2019 LXI	54493	24.70	796.0	47.30	5	1	1	2	4	10
2	Toyota Innova Crysta Touring Sport 2.4 MT	34000	13.68	2393.0	147.80	7	2	1	1	1	6
4	Hyundai i20 Magna	29000	18.50	1197.0	82.85	5	1	1	1	1	9
5	Mahindra XUV500 W8 2WD	85609	16.00	2179.0	140.00	7	2	1	2	4	7
6	Toyota Fortuner 4x2 AT TRD Sportivo	59000	12.55	2982.0	168.70	7	2	2	1	2	8

In [126]:
```
car3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5463 entries, 0 to 6018
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Name               5463 non-null   object 
 1   Kilometers_Driven  5463 non-null   int64  
 2   Mileage            5463 non-null   float64
 3   Engine             5463 non-null   float64
 4   Power              5463 non-null   float64
 5   Seats              5463 non-null   int32  
 6   Price              5463 non-null   float64
 7   Fuel               5463 non-null   int64  
 8   Transmission_Type  5463 non-null   int64  
 9   Owner              5463 non-null   int64  
 10  Locations          5463 non-null   int64  
 11  No_Year            5463 non-null   int64  
dtypes: float64(4), int32(1), int64(6), object(1)
memory usage: 533.5+ KB
```

In [127]:
```
car_dp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1128 entries, 1 to 1233
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Name               1128 non-null   object 
 1   Kilometers_Driven  1128 non-null   int64  
 2   Mileage            1128 non-null   float64
 3   Engine             1128 non-null   float64
 4   Power              1128 non-null   float64
 5   Seats              1128 non-null   int32  
 6   Fuel               1128 non-null   int64  
 7   Transmission_Type  1128 non-null   int64  
 8   Owner              1128 non-null   int64  
 9   Locations          1128 non-null   int64  
 10  No_Year            1128 non-null   int64  
dtypes: float64(3), int32(1), int64(6), object(1)
memory usage: 101.3+ KB
```

In [128]:
```
car3.describe()
```

Out[128]:

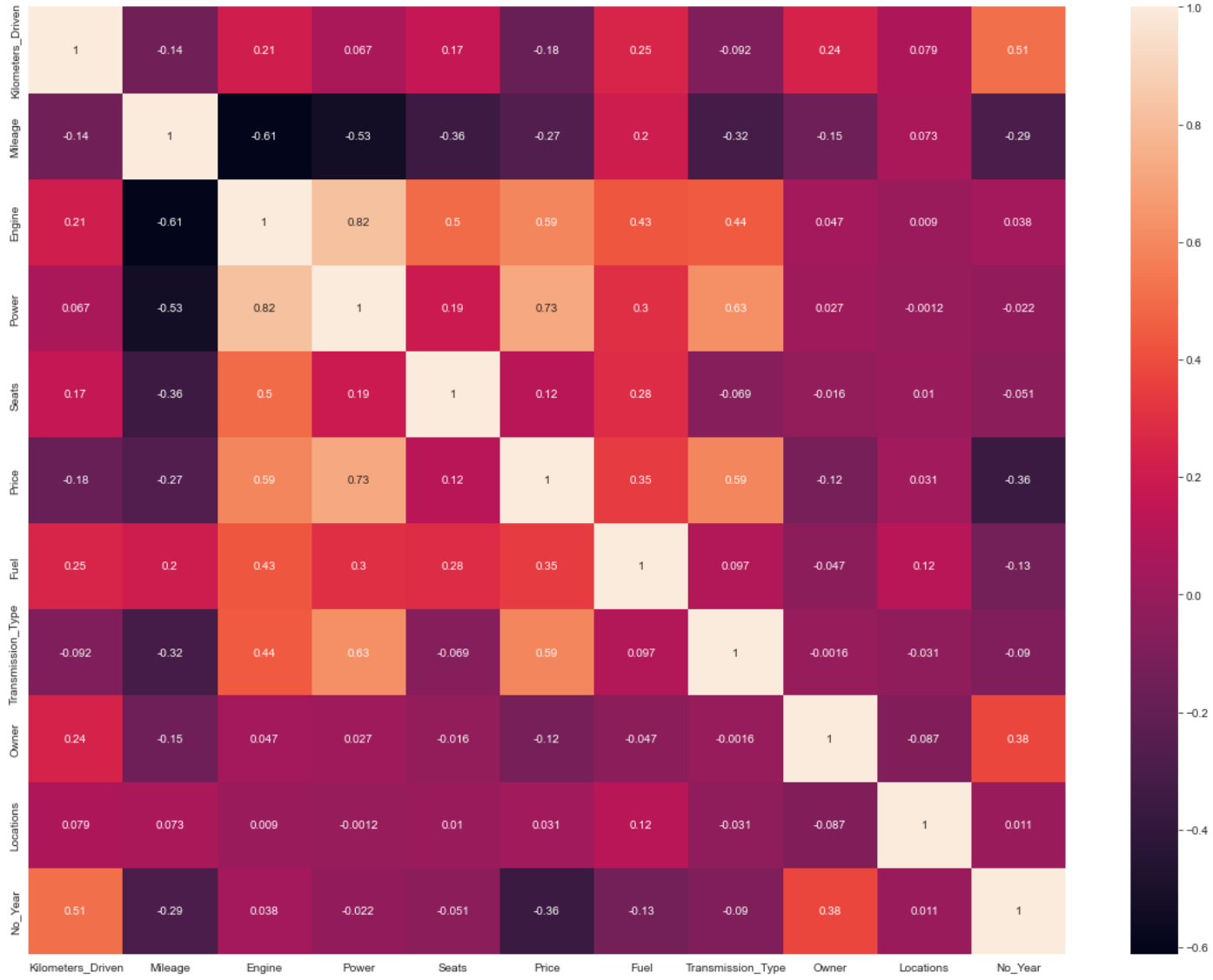
	Kilometers_Driven	Mileage	Engine	Power	Seats	Price	Fuel	Transmission_Type	Owner	Locations	No_Year
count	5463.000000	5463.000000	5463.000000	5463.000000	5463.000000	5463.000000	5463.000000	5463.000000	5463.000000	5463.000000	5463.000000
mean	53416.087315	18.659366	1538.704009	106.221104	5.251876	8.237910	1.529929	1.261212	1.186161	5.558667	9.455061
std	26793.696681	3.956665	485.066710	40.175087	0.726319	8.130409	0.514324	0.439335	0.429094	3.133749	3.106811
min	171.000000	8.700000	624.000000	34.200000	4.000000	0.440000	1.000000	1.000000	1.000000	1.000000	4.000000
25%	33000.000000	16.000000	1197.000000	75.000000	5.000000	3.500000	1.000000	1.000000	1.000000	3.000000	7.000000
50%	52000.000000	18.600000	1461.000000	90.000000	5.000000	5.500000	2.000000	1.000000	1.000000	6.000000	9.000000
75%	70348.000000	21.400000	1956.000000	126.240000	5.000000	9.000000	2.000000	2.000000	1.000000	8.000000	11.000000
max	130002.000000	28.400000	2999.000000	227.000000	8.000000	61.250000	3.000000	2.000000	3.000000	11.000000	21.000000

In [129]:
```
car3.corr()
```

Out[129]:

	Kilometers_Driven	Mileage	Engine	Power	Seats	Price	Fuel	Transmission_Type	Owner	Locations	No_Year
Kilometers_Driven	1.000000	-0.139803	0.213296	0.066842	0.167937	-0.181735	0.254234	-0.092238	0.242134	0.079410	0.510467
Mileage	-0.139803	1.000000	-0.611627	-0.533892	-0.358507	-0.270736	0.201598	-0.318430	-0.153100	0.073444	-0.292662
Engine	0.213296	-0.611627	1.000000	0.817930	0.499041	0.591247	0.433187	0.443334	0.047133	0.008958	0.037550
Power	0.066842	-0.533892	0.817930	1.000000	0.188660	0.729184	0.299040	0.625479	0.026984	-0.001182	-0.021924
Seats	0.167937	-0.358507	0.499041	0.188660	1.000000	0.121838	0.283679	-0.068522	-0.015954	0.010398	-0.050722
Price	-0.181735	-0.270736	0.591247	0.729184	0.121838	1.000000	0.345189	0.592920	-0.121046	0.031069	-0.363740
Fuel	0.254234	0.201598	0.433187	0.299040	0.283679	0.345189	1.000000	0.097061	-0.047234	0.124231	-0.134101
Transmission_Type	-0.092238	-0.318430	0.443334	0.625479	-0.068522	0.592920	0.097061	1.000000	-0.001605	-0.031279	-0.089785
Owner	0.242134	-0.153100	0.047133	0.026984	-0.015954	-0.121046	-0.047234	-0.001605	1.000000	-0.086753	0.377561
Locations	0.079410	0.073444	0.008958	-0.001182	0.010398	0.031069	0.124231	-0.031279	-0.086753	1.000000	0.011361
No_Year	0.510467	-0.292662	0.037550	-0.021924	-0.050722	-0.363740	-0.134101	-0.089785	0.377561	0.011361	1.000000

In [130]:
```
plt.figure(figsize=(20, 15))
sns.heatmap(car3.corr(),cmap='rocket', annot=True)
plt.show()
```



In [131]:
```
car_train = car3.drop(['Name'], axis=1)
car_dp= car_dp.drop(['Name'], axis=1)
```

In [132]:
```
car_train.isna().sum()
```

Out[132]:
```
Kilometers_Driven    0
Mileage              0
Engine               0
Power                0
Seats                0
Price                0
Fuel                 0
Transmission_Type    0
Owner                0
Locations            0
No_Year              0
dtype: int64
```

In [133]:
```
car_dp.isna().sum()
```

Out[133]:
```
Kilometers_Driven    0
Mileage              0
Engine               0
Power                0
Seats                0
Fuel                 0
Transmission_Type    0
Owner                0
Locations            0
No_Year              0
dtype: int64
```

Shape after Feature Engineering:_

In [134]:
```
car_train.shape
```

Out[134]:
```
(5463, 11)
```

In [135]:
```
car_dp.shape
```

Out[135]:
```
(1128, 10)
```

Model Training & Testing:

In [136]:
```
X = car_train.drop(['Price'], axis=1)
y = car_train.Price
```

In [137]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x = sc.fit_transform(X)
```

In [138]:
```
print("Shape of Input features: {}".format(X.shape))
print("Shape of Output features: {}".format(y.shape))
```

```
Shape of Input features: (5463, 10)
Shape of Output features: (5463,)
```

In [139]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state=100)
```

In [140]:
```
print("Training Input shape\t: {}".format(x_train.shape))
print("Testing Input shape\t: {}".format(x_test.shape))
print("Training Output shape\t: {}".format(y_train.shape))
print("Testing Output shape\t: {}".format(y_test.shape))
```

```
Training Input shape    : (4370, 10)
Testing Input shape     : (1093, 10)
Training Output shape   : (4370,)
Testing Output shape    : (1093,)
```

Linear Regression:

In [141]:
```
from sklearn.linear_model import LinearRegression
```

In [142]:
```
model1= LinearRegression().fit(x_train,y_train)
```

In [143]:
```
print(model1.intercept_)
```

```
8.184959797559056
```

In [144]:
```
print(model1.coef_)
```

```
[-0.94595808 -0.61633929  0.18823032  4.07301327 -0.33761692  1.46191295
  1.47463284 -0.05466604  0.28517595 -2.24118147]
```

In [145]:
```
print("Accuracy on Traing set\t:",model1.score(x_train,y_train))
print("Accuracy on Testing set\t:",model1.score(x_test,y_test))
```

```
Accuracy on Traing set  : 0.6886369714825684
Accuracy on Testing set : 0.7210770846514873
```

In [146]:
```
y_pred = model1.predict(x_test)
```

In [147]:
```
from sklearn.metrics import r2_score,mean_squared_error,mean_absolute_error

def errors(actual, pred):

    MAE=mean_absolute_error(actual, pred)
    MSE=mean_squared_error(actual, pred)
    RMSE=np.sqrt(MSE)
    RSQ=r2_score(actual, pred)

    print('Mean Absolute Error\t:', MAE)
    print('Mean Squared Error\t:', MSE)
    print('Root Mean Squared Error\t:', RMSE)
    print('R Squared Error\t\t:', RSQ)

def error_values(actual,pred):

    MAE=mean_absolute_error(actual, pred)
    MSE=mean_squared_error(actual, pred)
    RMSE=np.sqrt(MSE)
    RSQ=r2_score(actual, pred)

    return [MAE,MSE,RMSE,RSQ]
```

In [148]:
```
errors(y_test, y_pred)
```

```
Mean Absolute Error     : 2.877007148004066
Mean Squared Error      : 19.612547172711277
Root Mean Squared Error : 4.428605556234522
R Squared Error         : 0.7210770846514873
```

In [149]:
```
LRE = error_values(y_test, y_pred)
```

Polynomial Regression:

In [150]:
```
from sklearn.preprocessing import PolynomialFeatures
```

Degree = 2_

In [151]:
```
poly2 = PolynomialFeatures(degree=2)
```

In [152]:
```
x_train_poly2 = poly2.fit_transform(x_train)
x_test_poly2 = poly2.fit_transform(x_test)
```

In [153]:
```
model2 = LinearRegression().fit(x_train_poly2, y_train)
```

In [154]:
```
print(model2.intercept_)
```

```
-18235368696.720196
```

In [155]:
```
print(model2.coef_)
```

...

In [156]:
```
print("Accuracy on Traing set\t:",model2.score(x_train_poly2,y_train))
print("Accuracy on Testing set\t:",model2.score(x_test_poly2,y_test))
```

```
Accuracy on Traing set  : 0.8527732698908015
Accuracy on Testing set : 0.8732052378271983
```

In [157]:
```
y_pred_poly2 = model2.predict(x_test_poly2)
```

In [158]:
```
errors(y_test, y_pred_poly2)
```

```
Mean Absolute Error     : 1.8072632136087514
Mean Squared Error      : 8.915611151057913
Root Mean Squared Error : 2.9859020665550826
R Squared Error         : 0.8732052378271983
```

In [159]:
```
P2E = error_values(y_test, y_pred_poly2)
```

Degree = 3_

In [160]:
```
poly3 = PolynomialFeatures(degree=3)
```

In [161]:
```
x_train_poly3 = poly3.fit_transform(x_train)
x_test_poly3 = poly3.fit_transform(x_test)
```

In [162]:
```
model3=LinearRegression().fit(x_train_poly3, y_train)
```

In [163]:
```
print(model3.intercept_)
```

```
349985429653.7382
```

In [164]:
```
print(model3.coef_)
```

...

In [165]:
```
print("Accuracy on Traing set\t:",model3.score(x_train_poly3,y_train))
print("Accuracy on Testing set\t:",model3.score(x_test_poly3,y_test))
```

```
Accuracy on Traing set  : 0.8944746053809776
Accuracy on Testing set : 0.8837285177885276
```

In [166]:
```
y_pred_poly3 = model3.predict(x_test_poly3)
```

In [167]:
```
errors(y_test, y_pred_poly3)
```

```
Mean Absolute Error     : 1.709151297496855
Mean Squared Error      : 8.175663612522625
Root Mean Squared Error : 2.8593117375554953
R Squared Error         : 0.8837285177885276
```

In [168]:
```
P3E = error_values(y_test, y_pred_poly3)
```

Decision Tree:

In [169]:
```
from sklearn.tree import DecisionTreeRegressor
```

In [170]:
```
dt = DecisionTreeRegressor(random_state = 100).fit(x_train, y_train)
```

In [171]:
```
print("Accuracy on Traing set\t:",dt.score(x_train,y_train))
print("Accuracy on Testing set\t:",dt.score(x_test,y_test))
```

```
Accuracy on Traing set  : 0.9999910699612427
Accuracy on Testing set : 0.8337587885054981
```

In [172]:
```
y_pred_dt = dt.predict(x_test)
```

In [173]:
```
errors(y_test, y_pred_dt)
```

```
Mean Absolute Error     : 1.7094739249771271
Mean Squared Error      : 11.689299885635863
Root Mean Squared Error : 3.4189618140066824
R Squared Error         : 0.8337587885054981
```

In [174]:
```
DTE = error_values(y_test, y_pred_dt)
```

Random Forest:

In [175]:
```
from sklearn.ensemble import RandomForestRegressor
```

In [176]:
```
rf = RandomForestRegressor(n_estimators=100).fit(x_train, y_train)
```

In [177]:
```
print("Accuracy on Traing set\t:",rf.score(x_train,y_train))
print("Accuracy on Testing set\t:",rf.score(x_test,y_test))
```

```
Accuracy on Traing set  : 0.9871966760781558
Accuracy on Testing set : 0.9290969085989838
```

In [178]:
```
y_pred_rf = rf.predict(x_test)
```

In [179]:
```
errors(y_test, y_pred_rf)
```

```
Mean Absolute Error     : 1.2183631259530348
Mean Squared Error      : 4.985571813115301
Root Mean Squared Error : 2.2328394060288574
R Squared Error         : 0.9290969085989838
```

In [180]:
```
RFE = error_values(y_test, y_pred_rf)
```

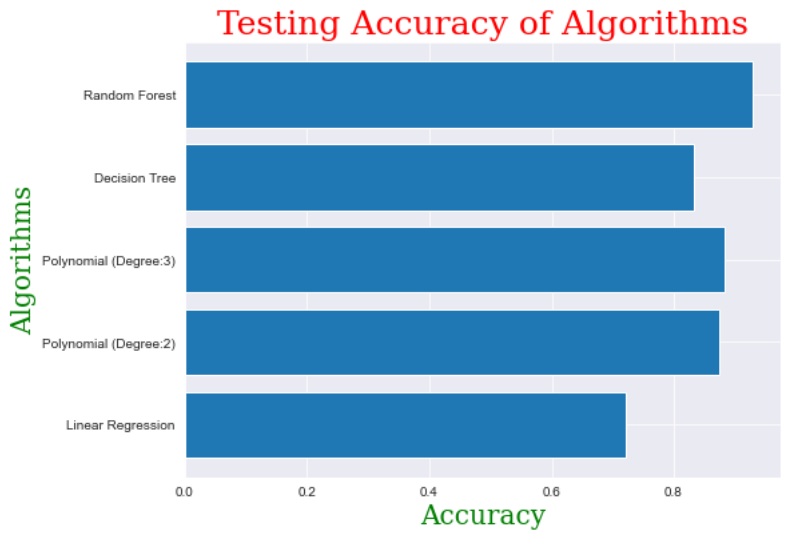
Comparing different Algorithms' Testing Accuracy_

In [181]:
```
data =  ['Linear Regression', 0.7211], ['Polynomial (Degree:2)', 0.8732], ['Polynomial (Degree:3)', 0.8837], ['Decision Tree', 0.8338], ['Random Forest',0.9291]
acc_df = pd.DataFrame(data, columns=['Algorithms','Testing Accuracy'])
acc_df
```

Out[181]:

	Algorithms	Testing Accuracy
0	Linear Regression	0.7211
1	Polynomial (Degree:2)	0.8732
2	Polynomial (Degree:3)	0.8837
3	Decision Tree	0.8338
4	Random Forest	0.9291

In [182]:
```
plt.figure(figsize=(8,6))
font1={'family':'serif','color':'red','size':25}
font2={'family':'serif','color':'g','size':20}

plt.barh(acc_df['Algorithms'],acc_df['Testing Accuracy'])

plt.xlabel("Accuracy",fontdict=font2)
plt.ylabel("Algorithms",fontdict=font2)
plt.title("Testing Accuracy of Algorithms",fontdict=font1)
plt.show()
```



Comparing different Algorithms' Errors_

In [183]:
```
ERROR = [LRE, P2E, P3E, DTE, RFE]
COL = ['MAE', 'MSE', 'RMSE', 'R-SQ']
ERRORS = ['Linear Regression','Polynomial (Degree:2)','Polynomial (Degree:3)','Decision Tree','Random Forest']
comparison_df = pd.DataFrame(ERROR, index = ERRORS, columns=COL)
comparison_df
```

Out[183]:

	MAE	MSE	RMSE	R-SQ
Linear Regression	2.877007	19.612547	4.428606	0.721077
Polynomial (Degree:2)	1.807263	8.915611	2.985902	0.873205
Polynomial (Degree:3)	1.709151	8.175664	2.859312	0.883729
Decision Tree	1.709474	11.689300	3.418962	0.833759
Random Forest	1.218363	4.985572	2.232839	0.929097

Model Deployment:

In [184]:
```
car_dp.head()
```

Out[184]:

	Kilometers_Driven	Mileage	Engine	Power	Seats	Fuel	Transmission_Type	Owner	Locations	No_Year
1	54493	24.70	796.0	47.30	5	1	1	2	4	10
2	34000	13.68	2393.0	147.80	7	2	1	1	1	6
4	29000	18.50	1197.0	82.85	5	1	1	1	1	9
5	85609	16.00	2179.0	140.00	7	2	1	2	4	7
6	59000	12.55	2982.0	168.70	7	2	2	1	2	8

In [185]:
```
cars = sc.fit_transform(car_dp)
```

In [186]:
```
pred = rf.predict(cars)
```

In [187]:
```
pred[0:10]
```

Out[187]:
```
array([ 2.5128, 17.5676,  4.4913, 12.1062, 21.945 ,  3.3427,  3.7466,
       10.5574, 15.2207,  4.4924])
```