In [1]:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statistics
```

In [2]:

```python
# data loading

df = pd.read_csv('used_cars_data.csv')
df.head()
```

Out[2]:

	S.No.	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine
0	0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First	26.6 km/kg	998 CC
1	1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67 kmpl	1582 CC
2	2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	18.2 kmpl	1199 CC
3	3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77 kmpl	1248 CC
4	4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.2 kmpl	1968 CC

In [3]:

```python
# size
df.size
```

Out[3]:

```
101542
```

In [4]:

```python
# shape
df.shape
```

Out[4]:

```
(7253, 14)
```

In [5]:

```python
# describe
df.describe()
```

Out[5]:

	S.No.	Year	Kilometers_Driven	Seats	Price
count	7253.000000	7253.000000	7.253000e+03	7200.000000	6019.000000
mean	3626.000000	2013.365366	5.869906e+04	5.279722	9.479468
std	2093.905084	3.254421	8.442772e+04	0.811660	11.187917
min	0.000000	1996.000000	1.710000e+02	0.000000	0.440000
25%	1813.000000	2011.000000	3.400000e+04	5.000000	3.500000
50%	3626.000000	2014.000000	5.341600e+04	5.000000	5.640000
75%	5439.000000	2016.000000	7.300000e+04	5.000000	9.950000
max	7252.000000	2019.000000	6.500000e+06	10.000000	160.000000

In [6]:

```python
# info

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7253 entries, 0 to 7252
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   S.No.              7253 non-null   int64  
 1   Name               7253 non-null   object 
 2   Location           7253 non-null   object 
 3   Year               7253 non-null   int64  
 4   Kilometers_Driven  7253 non-null   int64  
 5   Fuel_Type          7253 non-null   object 
 6   Transmission       7253 non-null   object 
 7   Owner_Type         7253 non-null   object 
 8   Mileage            7251 non-null   object 
 9   Engine             7207 non-null   object 
 10  Power              7207 non-null   object 
 11  Seats              7200 non-null   float64
 12  New_Price          1006 non-null   object 
 13  Price              6019 non-null   float64
dtypes: float64(2), int64(3), object(9)
memory usage: 793.4+ KB
```

In [7]:

```python
# columns
df.columns
```

Out[7]:

```
Index(['S.No.', 'Name', 'Location', 'Year', 'Kilometers_Driven', 'Fuel_Type',
       'Transmission', 'Owner_Type', 'Mileage', 'Engine', 'Power', 'Seats',
       'New_Price', 'Price'],
      dtype='object')
```

In [8]:

```python
# Remove serial number column as it is not adding any value

df = df.drop(['S.No.'],axis = 1)
df.head()
```

Out[8]:

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

In [9]:

```
# unique values in each column

df.nunique()
```

Out[9]:

```
Name                 2041
Location               11
Year                   23
Kilometers_Driven    3660
Fuel_Type               5
Transmission            2
Owner_Type              4
Mileage               450
Engine                150
Power                 386
Seats                   9
New_Price             625
Price                1373
dtype: int64
```

In [10]:

```
# null values

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

Out[10]:

```
Name                    0
Location                0
Year                    0
Kilometers_Driven       0
Fuel_Type               0
Transmission            0
Owner_Type              0
Mileage                 2
Engine                 46
Power                  46
Seats                  53
New_Price            6247
Price                1234
dtype: int64
```

In [11]:

```
# null values in %

df.isnull().mean()*100
```

Out[11]:

```
Name                  0.000000
Location              0.000000
Year                  0.000000
Kilometers_Driven     0.000000
Fuel_Type             0.000000
Transmission          0.000000
Owner_Type            0.000000
Mileage               0.027575
Engine                0.634220
Power                 0.634220
Seats                 0.730732
New_Price            86.129877
Price                17.013650
dtype: float64
```

In [12]:

```python
# as 86% of data in column new_price is missing , we can drop that column

df = df.drop(['New_Price'],axis = 1)
df.head()
```

Out[12]:

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

In [13]:

```python
# feature generation  : extract brand and model name of car
df.iloc[0][0]
```

Out[13]:

```
'Maruti Wagon R LXI CNG'
```

In [14]:

```python
t = 'Maruti Wagon R LXI CNG'
t.split()[0]
```

Out[14]:

```
'Maruti'
```

In [15]:

```python
# create new column in data named "Brand"
df['Brand'] = df['Name'].apply(lambda x: x.split()[0])
# df['Brand']  = df.Name.str.split().str.get(0)
df.head()
```

Out[15]:

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

In [16]:

```python
# create new column in data named "model"
#df['Model'] = df['Name'].apply(lambda x: x.split()[1])
#df.head()
```

In [17]:

```
# create new column in data named "model"
df['Model_1'] = df.Name.str.split().str.get(1) + df.Name.str.split().str.get(2)
df.head()
```

Out[17]:

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

In [18]:

```
# droping repeated column model

#df = df.drop(['Model'],axis = 1)
```

In [19]:

```
# data cleaning

df.Brand.unique()
```

Out[19]:

```
array(['Maruti', 'Hyundai', 'Honda', 'Audi', 'Nissan', 'Toyota',
       'Volkswagen', 'Tata', 'Land', 'Mitsubishi', 'Renault',
       'Mercedes-Benz', 'BMW', 'Mahindra', 'Ford', 'Porsche', 'Datsun',
       'Jaguar', 'Volvo', 'Chevrolet', 'Skoda', 'Mini', 'Fiat', 'Jeep',
       'Smart', 'Ambassador', 'Isuzu', 'ISUZU', 'Force', 'Bentley',
       'Lamborghini', 'Hindustan', 'OpelCorsa'], dtype=object)
```

In [20]:

```
# replace with proper names

df['Brand'].replace({'Mini':'Minicooper',"ISUZU":'Isuzu','Land':'Landrover'},inp
```

In [21]:

```
df.Brand.unique()
```

Out[21]:

```
array(['Maruti', 'Hyundai', 'Honda', 'Audi', 'Nissan', 'Toyota',
       'Volkswagen', 'Tata', 'Landrover', 'Mitsubishi', 'Renault',
       'Mercedes-Benz', 'BMW', 'Mahindra', 'Ford', 'Porsche', 'Datsun',
       'Jaguar', 'Volvo', 'Chevrolet', 'Skoda', 'Minicooper', 'Fiat',
       'Jeep', 'Smart', 'Ambassador', 'Isuzu', 'Force', 'Bentley',
       'Lamborghini', 'Hindustan', 'OpelCorsa'], dtype=object)
```

In [22]:

```
# extract numerical value from mileage

df.iloc[0][7].split(' ')[0]
```

Out[22]:

```
'26.6'
```

In [24]:

```python
df['Mileage_num'] = df.Mileage.str.split().str.get(0)
df['Engine_num'] = df.Engine.str.split().str.get(0)
df['Power_num'] = df.Power.str.split().str.get(0)
df.head()
```

Out[24]:

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

In [27]:

```python
# create new feature as car age

from datetime import date

df['Car_age'] = date.today().year-df['Year']
df.head()
```

Out[27]:

ters_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price	Brand	Model_1	Mileag
72000	CNG	Manual	First	26.6 km/kg	998 CC	58.16 bhp	5.0	1.75	Maruti	WagonR	
41000	Diesel	Manual	First	19.67 kmpl	1582 CC	126.2 bhp	5.0	12.50	Hyundai	Creta1.6	
46000	Petrol	Manual	First	18.2 kmpl	1199 CC	88.7 bhp	5.0	4.50	Honda	JazzV	
87000	Diesel	Manual	First	20.77 kmpl	1248 CC	88.76 bhp	7.0	6.00	Maruti	ErtigaVDI	
40670	Diesel	Automatic	Second	15.2 kmpl	1968 CC	140.8 bhp	5.0	17.74	Audi	A4New	

In [31]:

```python
# extract some insights using describe command
df.describe()
```

Out[31]:

	Year	Kilometers_Driven	Seats	Price	Car_age
count	7253.000000	7.253000e+03	7200.000000	6019.000000	7253.000000
mean	2013.365366	5.869906e+04	5.279722	9.479468	9.634634
std	3.254421	8.442772e+04	0.811660	11.187917	3.254421
min	1996.000000	1.710000e+02	0.000000	0.440000	4.000000
25%	2011.000000	3.400000e+04	5.000000	3.500000	7.000000
50%	2014.000000	5.341600e+04	5.000000	5.640000	9.000000
75%	2016.000000	7.300000e+04	5.000000	9.950000	12.000000
max	2019.000000	6.500000e+06	10.000000	160.000000	27.000000

In [ ]:

```python
# we have data from year 1996 - 2019 cars
# on an average 5 seaters car are more in number
# on an average we have cars with 58000km run
# looking at price 160k , we can say that its an outlier or data entry issue
```

In [29]:

```python
df.describe(include = 'all')
```

Out[29]:

ocation	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seat
7253	7253.000000	7.253000e+03	7253	7253	7253	7251	7207	7207	7200.00000
11	NaN	NaN	5	2	4	450	150	386	Nal
Mumbai	NaN	NaN	Diesel	Manual	First	17.0 kmpl	1197 CC	74 bhp	Nal
949	NaN	NaN	3852	5204	5952	207	732	280	Nal
NaN	2013.365366	5.869906e+04	NaN	NaN	NaN	NaN	NaN	NaN	5.27972
NaN	3.254421	8.442772e+04	NaN	NaN	NaN	NaN	NaN	NaN	0.81166
NaN	1996.000000	1.710000e+02	NaN	NaN	NaN	NaN	NaN	NaN	0.00000
NaN	2011.000000	3.400000e+04	NaN	NaN	NaN	NaN	NaN	NaN	5.00000
NaN	2014.000000	5.341600e+04	NaN	NaN	NaN	NaN	NaN	NaN	5.00000
NaN	2016.000000	7.300000e+04	NaN	NaN	NaN	NaN	NaN	NaN	5.00000
NaN	2019.000000	6.500000e+06	NaN	NaN	NaN	NaN	NaN	NaN	10.00000

In [33]:

```python
# converting str col to float

df['Mileage_num'] = df['Mileage_num'].astype(float)
df['Engine_num'] = df['Engine_num'].astype(float)
#df['Power_num'] = df['Power_num'].astype(float)
```

In [34]:

```
df.describe()
```

Out[34]:

	Year	Kilometers_Driven	Seats	Price	Mileage_num	Engine_num	Car_age
count	7253.000000	7.253000e+03	7200.000000	6019.000000	7251.000000	7207.000000	7253.000000
mean	2013.365366	5.869906e+04	5.279722	9.479468	18.141580	1616.573470	9.634634
std	3.254421	8.442772e+04	0.811660	11.187917	4.562197	595.285137	3.254421
min	1996.000000	1.710000e+02	0.000000	0.440000	0.000000	72.000000	4.000000
25%	2011.000000	3.400000e+04	5.000000	3.500000	15.170000	1198.000000	7.000000
50%	2014.000000	5.341600e+04	5.000000	5.640000	18.160000	1493.000000	9.000000
75%	2016.000000	7.300000e+04	5.000000	9.950000	21.100000	1968.000000	12.000000
max	2019.000000	6.500000e+06	10.000000	160.000000	33.540000	5998.000000	27.000000

In [ ]:

```
# 0 mileage car is something weird
# 0 seat car is something weird
# check car with 27 years age
```

In [ ]:

```

```

In [35]:

```
# lets seepearate numerical and categorical features

cat_features = df.select_dtypes(include = ['object']).columns
print("Categorical features : ", cat_features)
```

```
Categorical features :  Index(['Name', 'Location', 'Fuel_Type', 'Transmission', 'Ow
ner_Type',
       'Mileage', 'Engine', 'Power', 'Brand', 'Model_1', 'Power_num'],
      dtype='object')
```

In [36]:

```
numerical_features = df.select_dtypes(include = [np.number]).columns # anything
print("numerical features : ", numerical_features)
```

```
numerical features :  Index(['Year', 'Kilometers_Driven', 'Seats', 'Price', 'Mileag
e_num',
       'Engine_num', 'Car_age'],
      dtype='object')
```

In [39]:

```
# converting series data to list
numerical_features = numerical_features.tolist()
type(numerical_features)
```

Out[39]:

```
list
```

In [43]:

```
# analysis on numerical features
numerical_features
```

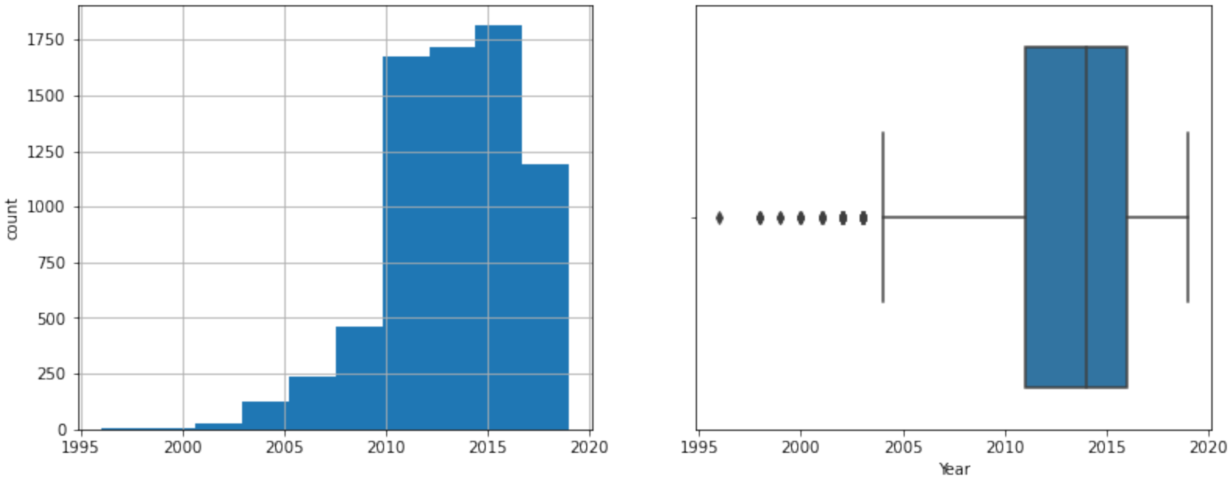
Out[43]:

```
['Year',
 'Kilometers_Driven',
 'Seats',
 'Price',
 'Mileage_num',
 'Engine_num',
 'Car_age']
```

In [48]:

```
for i in numerical_features:
    print(i)
    print('Skewness : ' , round(df[i].skew(),3))
    plt.figure(figsize = (13,5))
    plt.subplot(1,2,1)
    df[i].hist()
    plt.ylabel('count')
    plt.subplot(1,2,2)
    sns.boxplot(x = df[i])
    plt.show()
```

```
Year
Skewness :  -0.84
```



In [ ]:

In [52]:

```
# categorical data plots
df['Fuel_Type'].value_counts()
```

Out[52]:

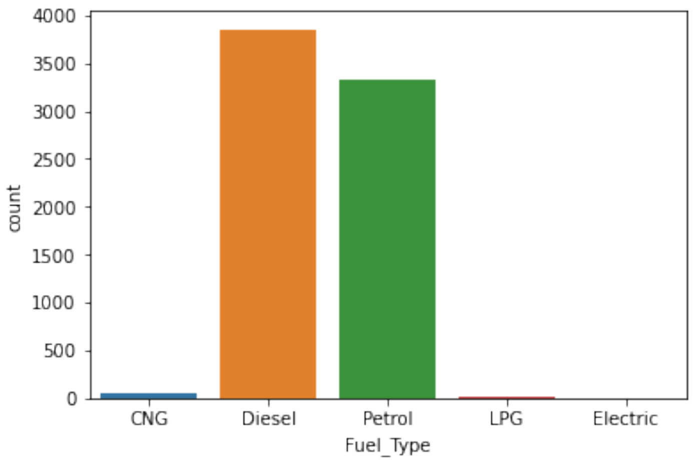
```
Diesel      3852
Petrol      3325
CNG           62
LPG           12
Electric       2
Name: Fuel_Type, dtype: int64
```

In [57]:

```
# countplot of different fuel types
sns.countplot(x = 'Fuel_Type',data = df)
```

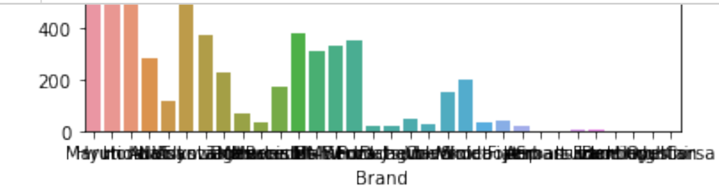
Out[57]:

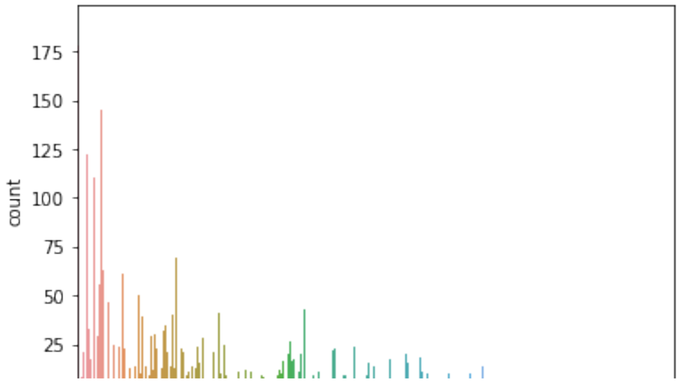
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe0e5debc40>
```



In [58]:

```
for i in cat_features:
    sns.countplot(x = i,data = df)
    plt.show()
```





In [59]:

```
# 5 categorical features we can plot properly
```

In [ ]:

```
# applying log transform on skewed data can improve skewness
```

In [ ]:

In [60]:

```
# treating null values
```

In [61]:

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

Out[61]:

```
Name                    0
Location                0
Year                    0
Kilometers_Driven       0
Fuel_Type               0
Transmission            0
Owner_Type              0
Mileage                 2
Engine                 46
Power                  46
Seats                  53
Price                1234
Brand                   0
Model_1                 1
Mileage_num             2
Engine_num             46
Power_num              46
Car_age                 0
dtype: int64
```

In [62]:

```
df.dropna(subset = ['Mileage','Mileage_num'],inplace = True)
```

In [63]:

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

Out[63]:

```
Name                    0
Location                0
Year                    0
Kilometers_Driven       0
Fuel_Type               0
Transmission            0
Owner_Type              0
Mileage                 0
Engine                 46
Power                  46
Seats                  53
Price                1234
Brand                   0
Model_1                 1
Mileage_num             0
Engine_num             46
Power_num              46
Car_age                 0
dtype: int64
```

In [64]:

```
df['Engine_num'].fillna(df['Engine_num'].mean,inplace = True)
df['Power_num'].fillna(df['Power_num'].mean,inplace = True)
```

In [65]:

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

Out[65]:

```
Name                    0
Location                0
Year                    0
Kilometers_Driven       0
Fuel_Type               0
Transmission            0
Owner_Type              0
Mileage                 0
Engine                 46
Power                  46
Seats                  53
Price                1234
Brand                   0
Model_1                 1
Mileage_num             0
Engine_num              0
Power_num               0
Car_age                 0
dtype: int64
```

In [ ]:

```
# for imputing price null values :
# segg data brand wise (by filtering )and then impute
```

In [69]:

```
df_audi = df[df['Brand'] == 'Audi']
df_audi.head()
```

Out[69]:

Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price	Brand	Mc
2013	40670	Diesel	Automatic	Second	15.2 kmpl	1968 CC	140.8 bhp	5.0	17.74	Audi	A
2015	55985	Petrol	Automatic	First	13.53 kmpl	1984 CC	177.01 bhp	5.0	23.50	Audi	A6
2010	35000	Diesel	Automatic	First	12.4 kmpl	2698 CC	179.5 bhp	5.0	11.50	Audi	
2015	13648	Diesel	Automatic	First	17.11 kmpl	1968 CC	174.33 bhp	5.0	21.43	Audi	
2012	65664	Diesel	Automatic	First	16.55 kmpl	1968 CC	140 bhp	5.0	13.50	Audi	

In [70]:

```
df_audi.isnull().sum()
```

Out[70]:

```
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                49
Brand                 0
Model_1               0
Mileage_num           0
Engine_num            0
Power_num             0
Car_age               0
dtype: int64
```

In [71]:

```
df_audi['Price'].fillna(df_audi['Price'].mean,inplace = True)
```

```
/Users/kunalshriwas/opt/anaconda3/lib/python3.8/site-packages/pandas/core/generic.p
y:6245: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/
user_guide/indexing.html#returning-a-view-versus-a-copy
(https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  self._update_inplace(new_data)
```

In [72]:

```
df_audi.isnull().sum()
```

Out[72]:

```
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
Brand                0
Model_1              0
Mileage_num          0
Engine_num           0
Power_num            0
Car_age              0
dtype: int64
```

In [ ]:

```

```

In [ ]:

```

```