Importing Library

In [1]:
```
import os
```

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Changing the working directory

In [3]:
```
os.chdir("E:\ML Projects")
```

Importing Dataset

In [4]:
```
df=pd.read_csv("E:/Machine Learning/Project CSV files/Toyota.csv")
df
```

Out[4]:

	Unnamed: 0	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	0	13500	23.0	46986	Diesel	90	1.0	0	2000	three	1165
1	1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165
2	2	13950	24.0	41711	Diesel	90	NaN	0	2000	3	1165
3	3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165
4	4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...	...
1431	1431	7500	NaN	20544	Petrol	86	1.0	0	1300	3	1025
1432	1432	10845	72.0	??	Petrol	86	0.0	0	1300	3	1015
1433	1433	8500	NaN	17016	Petrol	86	0.0	0	1300	3	1015
1434	1434	7250	70.0	??	NaN	86	1.0	0	1300	3	1015
1435	1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114

1436 rows × 11 columns

In [5]:
```
df=pd.read_csv("E:/Machine Learning/Project CSV files/Toyota.csv" , index_col = 0 , na
df
```

Out[5]:

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986.0	Diesel	90.0	1.0	0	2000	three	1165
1	13750	23.0	72937.0	Diesel	90.0	1.0	0	2000	3	1165
2	13950	24.0	41711.0	Diesel	90.0	NaN	0	2000	3	1165
3	14950	26.0	48000.0	Diesel	90.0	0.0	0	2000	3	1165
4	13750	30.0	38500.0	Diesel	90.0	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1431	7500	NaN	20544.0	Petrol	86.0	1.0	0	1300	3	1025
1432	10845	72.0	NaN	Petrol	86.0	0.0	0	1300	3	1015
1433	8500	NaN	17016.0	Petrol	86.0	0.0	0	1300	3	1015
1434	7250	70.0	NaN	NaN	86.0	1.0	0	1300	3	1015
1435	6950	76.0	1.0	Petrol	110.0	0.0	0	1600	5	1114

1436 rows × 10 columns

Creating copy of original data

In [6]:
```
df2 = df.copy()
df2
```

Out[6]:

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986.0	Diesel	90.0	1.0	0	2000	three	1165
1	13750	23.0	72937.0	Diesel	90.0	1.0	0	2000	3	1165
2	13950	24.0	41711.0	Diesel	90.0	NaN	0	2000	3	1165
3	14950	26.0	48000.0	Diesel	90.0	0.0	0	2000	3	1165
4	13750	30.0	38500.0	Diesel	90.0	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1431	7500	NaN	20544.0	Petrol	86.0	1.0	0	1300	3	1025
1432	10845	72.0	NaN	Petrol	86.0	0.0	0	1300	3	1015
1433	8500	NaN	17016.0	Petrol	86.0	0.0	0	1300	3	1015
1434	7250	70.0	NaN	NaN	86.0	1.0	0	1300	3	1015
1435	6950	76.0	1.0	Petrol	110.0	0.0	0	1600	5	1114

1436 rows × 10 columns

Understanding Analysing Of The DataSet

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

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1436 entries, 0 to 1435
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Price       1436 non-null   int64  
 1   Age         1336 non-null   float64
 2   KM          1421 non-null   float64
 3   FuelType    1336 non-null   object 
 4   HP          1430 non-null   float64
 5   MetColor    1286 non-null   float64
 6   Automatic   1436 non-null   int64  
 7   CC          1436 non-null   int64  
 8   Doors       1436 non-null   object 
 9   Weight      1436 non-null   int64  
dtypes: float64(4), int64(4), object(2)
memory usage: 123.4+ KB
```

In [8]:
```
df.corr()
```

```
C:\Users\akash\AppData\Local\Temp\ipykernel_10356\1134722465.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.corr()
```

Out[8]:

	Price	Age	KM	HP	MetColor	Automatic	CC	Weight
Price	1.000000	-0.878407	-0.574720	0.309902	0.112041	0.033081	0.165067	0.581198
Age	-0.878407	1.000000	0.512735	-0.157904	-0.099659	0.032573	-0.120706	-0.464299
KM	-0.574720	0.512735	1.000000	-0.335285	-0.093825	-0.081248	0.299993	-0.026271
HP	0.309902	-0.157904	-0.335285	1.000000	0.064749	0.013755	0.053758	0.086737
MetColor	0.112041	-0.099659	-0.093825	0.064749	1.000000	-0.013973	0.029189	0.057142
Automatic	0.033081	0.032573	-0.081248	0.013755	-0.013973	1.000000	-0.069321	0.057249
CC	0.165067	-0.120706	0.299993	0.053758	0.029189	-0.069321	1.000000	0.651450
Weight	0.581198	-0.464299	-0.026271	0.086737	0.057142	0.057249	0.651450	1.000000

In [9]:
```
df.describe()
```

Out[9]:

	Price	Age	KM	HP	MetColor	Automatic	CC
count	1436.000000	1336.000000	1421.000000	1430.000000	1286.000000	1436.000000	1436.000000
mean	10730.824513	55.672156	68647.239972	101.478322	0.674961	0.055710	1566.827994
std	3626.964585	18.589804	37333.023589	14.768255	0.468572	0.229441	187.182436
min	4350.000000	1.000000	1.000000	69.000000	0.000000	0.000000	1300.000000
25%	8450.000000	43.000000	43210.000000	90.000000	0.000000	0.000000	1400.000000
50%	9900.000000	60.000000	63634.000000	110.000000	1.000000	0.000000	1600.000000
75%	11950.000000	70.000000	87000.000000	110.000000	1.000000	0.000000	1600.000000
max	32500.000000	80.000000	243000.000000	192.000000	1.000000	1.000000	2000.000000

In [10]:
```
df.isnull().sum()
```

Out[10]:
```
Price          0
Age          100
KM            15
FuelType     100
HP             6
MetColor     150
Automatic      0
CC             0
Doors          0
Weight         0
dtype: int64
```

In [11]:
```
df.head()
```

Out[11]:

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986.0	Diesel	90.0	1.0	0	2000	three	1165
1	13750	23.0	72937.0	Diesel	90.0	1.0	0	2000	3	1165
2	13950	24.0	41711.0	Diesel	90.0	NaN	0	2000	3	1165
3	14950	26.0	48000.0	Diesel	90.0	0.0	0	2000	3	1165
4	13750	30.0	38500.0	Diesel	90.0	0.0	0	2000	3	1170

In [12]:
```
df.count()
```

Out[12]:
```
Price        1436
Age          1336
KM           1421
FuelType     1336
HP           1430
MetColor     1286
Automatic    1436
CC           1436
Doors        1436
Weight       1436
dtype: int64
```

In [13]:
```
df['Price'].value_counts()
```

Out[13]:
```
8950     109
9950      84
7950      63
10950     62
11950     47
         ...
11790      1
4750       1
4350       1
21125      1
10845      1
Name: Price, Length: 236, dtype: int64
```

In [14]:
```
df['Price'].value_counts().sort_values()
```

Out[14]:
```
10845      1
5800       1
5740       1
10295      1
7300       1
         ...
11950     47
10950     62
7950      63
9950      84
8950     109
Name: Price, Length: 236, dtype: int64
```

In [15]:
```
df['Price'].sort_values()
```

Out[15]:
```
191      4350
1048     4400
393      4450
192      4750
402      5150
         ...
112     24950
115     24990
110     31000
111     31275
109     32500
Name: Price, Length: 1436, dtype: int64
```

In [16]:
```
df['Price'].unique()
```

Out[16]:
```
array([13500, 13750, 13950, 14950, 12950, 16900, 18600, 21500, 20950,
       19950, 19600, 22500, 22000, 22750, 17950, 16750, 16950, 15950,
       16250, 17495, 15750, 15500, 14750, 19000, 15800, 21950, 20500,
       13250, 15250, 18950, 15999, 16500, 18750, 22250, 12995, 18450,
       16895, 14900, 17250, 15450, 16650, 17450, 16450, 18900, 18990,
       18500, 19450, 18800, 32500, 31000, 31275, 24950, 22950, 24990,
       17900, 19250, 16350, 21750, 15850, 23000, 19900, 23950, 24500,
       17200, 19500, 16868, 19750, 20750, 17650, 17795, 18245, 23750,
       18700, 21125,  6950,  9500, 11950,  7750,  4350,  4750, 11750,
       11900,  9950, 11495, 11250, 10500, 10450, 11500, 12500, 10950,
       11450, 11790, 12450, 11690, 12750, 11925, 12900, 11650, 10850,
        9940, 13450, 12495, 12000, 11480, 14990, 12850, 11700, 11895,
       13875, 12295, 13995,  9900, 11990, 10750, 11695, 11000, 12400,
       12200, 12695, 14350, 10250,  6500,  6400,  7000,  8900,  8500,
        8950,  9250,  9450,  8250,  4450,  9000,  5150,  7900, 10900,
        9750, 11290, 10895, 10995,  9850,  8695, 10990,  8750,  9930,
        9799,  9700,  9990,  9475, 10000, 10495,  9400,  9650,  9550,
       13000, 11710,  9980, 12250, 11930, 10800, 10600,  7500,  5950,
        6900,  5751,  7950,  6250,  8450,  7350,  9800,  7995,  8600,
        7250,  8000,  8495,  9895,  7999,  8490,  8150,  7450,  9130,
        8990,  8995,  9995, 10400,  8800,  7800,  8100,  8200, 10295,
        9795, 10350,  8400,  8895,  9390,  8745,  8850,  9695,  9245,
        5900,  6000,  5250,  4400,  6750,  6150,  5750,  5800,  5740,
        6550,  6450,  5600,  6650,  7600,  7460,  6800,  8700,  6640,
        8050,  7795,  6490,  6425,  6495,  6990,  7200,  7300,  9200,
        7850,  7495,  7990,  7490,  7145,  7400,  6999,  7499,  7150,
        5845, 10845], dtype=int64)
```

In [17]:
```
df['Price'].nunique()
```

Out[17]:
```
236
```

In [18]:
```
df.sample(7) # randomly provide values
```

Out[18]:

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
521	11950	51.0	50000.0	Petrol	86.0	1.0	0	1300	5	1045
376	11500	39.0	12000.0	Petrol	110.0	1.0	0	1600	5	1075
1012	9950	66.0	36658.0	Petrol	110.0	0.0	0	1600	3	1050
1169	9250	NaN	96000.0	Petrol	110.0	1.0	0	1600	5	1085
883	8250	65.0	61384.0	Petrol	110.0	1.0	0	1600	3	1050
752	8450	65.0	80439.0	Petrol	110.0	1.0	0	1600	4	1035
300	12750	37.0	39757.0	Petrol	110.0	1.0	0	1600	5	1075

Attributes

In [19]:
```
df.columns
```

Out[19]:
```
Index(['Price', 'Age', 'KM', 'FuelType', 'HP', 'MetColor', 'Automatic', 'CC',
       'Doors', 'Weight'],
      dtype='object')
```

In [20]:
```
df.dtypes
```

Out[20]:
```
Price          int64
Age          float64
KM           float64
FuelType      object
HP           float64
MetColor     float64
Automatic      int64
CC             int64
Doors         object
Weight         int64
dtype: object
```

In [21]:
```
df.shape,df.size,type(df)
```

Out[21]:
```
((1436, 10), 14360, pandas.core.frame.DataFrame)
```

In [22]:
```
df.dropna()
```

Out[22]:

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986.0	Diesel	90.0	1.0	0	2000	three	1165
1	13750	23.0	72937.0	Diesel	90.0	1.0	0	2000	3	1165
3	14950	26.0	48000.0	Diesel	90.0	0.0	0	2000	3	1165
4	13750	30.0	38500.0	Diesel	90.0	0.0	0	2000	3	1170
5	12950	32.0	61000.0	Diesel	90.0	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1423	7950	80.0	35821.0	Petrol	86.0	0.0	1	1300	3	1015
1424	7750	73.0	34717.0	Petrol	86.0	0.0	0	1300	3	1015
1429	8950	78.0	24000.0	Petrol	86.0	1.0	1	1300	5	1065
1430	8450	80.0	23000.0	Petrol	86.0	0.0	0	1300	3	1015
1435	6950	76.0	1.0	Petrol	110.0	0.0	0	1600	5	1114

1096 rows × 10 columns

In [23]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1436 entries, 0 to 1435
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Price       1436 non-null   int64  
 1   Age         1336 non-null   float64
 2   KM          1421 non-null   float64
 3   FuelType    1336 non-null   object 
 4   HP          1430 non-null   float64
 5   MetColor    1286 non-null   float64
 6   Automatic   1436 non-null   int64  
 7   CC          1436 non-null   int64  
 8   Doors       1436 non-null   object 
 9   Weight      1436 non-null   int64  
dtypes: float64(4), int64(4), object(2)
memory usage: 123.4+ KB
```

Removing missing values from the dataframe

In [24]:
```
df.dropna(axis = 0, inplace = True)
df
```

Out[24]:

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986.0	Diesel	90.0	1.0	0	2000	three	1165
1	13750	23.0	72937.0	Diesel	90.0	1.0	0	2000	3	1165
3	14950	26.0	48000.0	Diesel	90.0	0.0	0	2000	3	1165
4	13750	30.0	38500.0	Diesel	90.0	0.0	0	2000	3	1170
5	12950	32.0	61000.0	Diesel	90.0	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1423	7950	80.0	35821.0	Petrol	86.0	0.0	1	1300	3	1015
1424	7750	73.0	34717.0	Petrol	86.0	0.0	0	1300	3	1015
1429	8950	78.0	24000.0	Petrol	86.0	1.0	1	1300	5	1065
1430	8450	80.0	23000.0	Petrol	86.0	0.0	0	1300	3	1015
1435	6950	76.0	1.0	Petrol	110.0	0.0	0	1600	5	1114

1096 rows × 10 columns

Frequency tables

pandas.crosstab()

In [25]:
```
pd.crosstab(index = df2['FuelType'] ,columns = 'count' , dropna = True)
```

Out[25]:

FuelType	col_0 count
CNG	15
Diesel	144
Petrol	1177

Two-way table

In [26]:
```
pd.crosstab(index = df2['Automatic'] ,
            columns = df2['FuelType'] ,
            dropna = True)
```

Out[26]:

Automatic	FuelType CNG	Diesel	Petrol
0	15	144	1104
1	0	0	73

Two-way table - Joint Probability

In [27]:
```
pd.crosstab(index = df2['Automatic'] ,
            columns = df2['FuelType'] ,
            normalize = True,
            dropna = True)
```

Out[27]:

Automatic	FuelType CNG	Diesel	Petrol
0	0.011228	0.107784	0.826347
1	0.000000	0.000000	0.054641

Two-way table - Marginal Probability

In [28]:
```
pd.crosstab(index = df2['Automatic'] ,
            columns = df2['FuelType'] ,
            margins = True,
            normalize = True,
            dropna = True)
```

Out[28]:

Automatic	FuelType CNG	Diesel	Petrol	All
0	0.011228	0.107784	0.826347	0.945359
1	0.000000	0.000000	0.054641	0.054641
All	0.011228	0.107784	0.880988	1.000000

Two-way table - Conditional Probability

In [29]:
```
pd.crosstab(index = df2['Automatic'] ,
            columns = df2['FuelType'] ,
            margins = True,
             dropna = True ,
            normalize = 'index')
```

Out[29]:

FuelType	CNG	Diesel	Petrol
Automatic			
0	0.011876	0.114014	0.874109
1	0.000000	0.000000	1.000000
All	0.011228	0.107784	0.880988

In [30]:
```
pd.crosstab(index = df2['Automatic'] ,
            columns = df2['FuelType'] ,
            margins = True,
             dropna = True ,
            normalize = 'columns')
```

Out[30]:

FuelType	CNG	Diesel	Petrol	All
Automatic				
0	1.0	1.0	0.937978	0.945359
1	0.0	0.0	0.062022	0.054641

Correlation

DataFrame.corr(self , method = 'Pearson')

In [31]:
```
df.corr()
```

```
C:\Users\akash\AppData\Local\Temp\ipykernel_10356\1134722465.py:1: FutureWarning: The
default value of numeric_only in DataFrame.corr is deprecated. In a future version, i
t will default to False. Select only valid columns or specify the value of numeric_on
ly to silence this warning.
  df.corr()
```

Out[31]:

	Price	Age	KM	HP	MetColor	Automatic	CC	Weight
Price	1.000000	-0.877706	-0.601944	0.334261	0.117381	0.045111	0.099880	0.532614
Age	-0.877706	1.000000	0.525695	-0.162063	-0.106291	0.026304	-0.084851	-0.442295
KM	-0.601944	0.525695	1.000000	-0.368629	-0.109031	-0.054777	0.319733	-0.029703
HP	0.334261	-0.162063	-0.368629	1.000000	0.065218	0.023112	0.037291	0.084527
MetColor	0.117381	-0.106291	-0.109031	0.065218	1.000000	-0.000476	0.009902	0.055382
Automatic	0.045111	0.026304	-0.054777	0.023112	-0.000476	1.000000	-0.053516	0.069788
CC	0.099880	-0.084851	0.319733	0.037291	0.009902	-0.053516	1.000000	0.623643
Weight	0.532614	-0.442295	-0.029703	0.084527	0.055382	0.069788	0.623643	1.000000

Let's check the no. of variables available under numerical_data

In [32]:
```
numerical_data = df2.select_dtypes(exclude = [object])
print(numerical_data.shape)
```

```
(1436, 8)
```

In [33]:
```
corr_matrix = numerical_data.corr()
corr_matrix
```

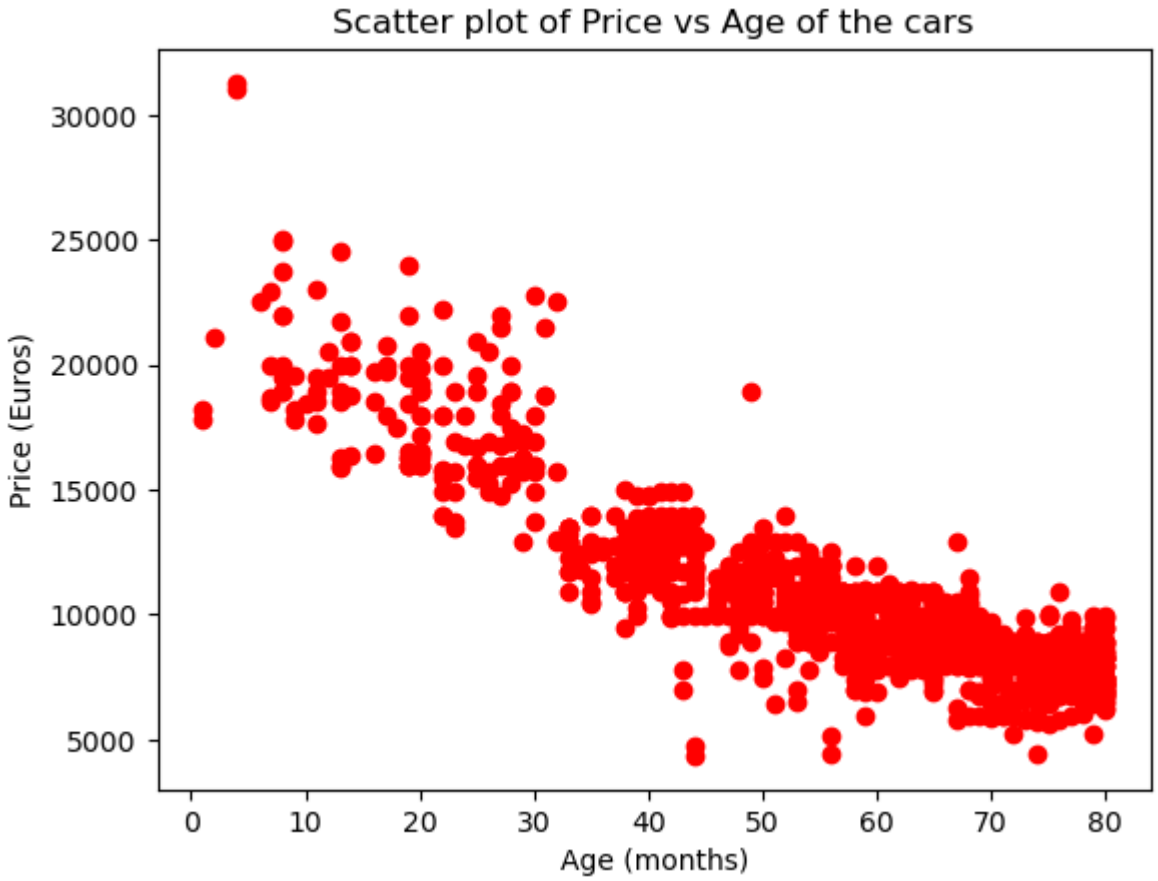
Out[33]:

	Price	Age	KM	HP	MetColor	Automatic	CC	Weight
Price	1.000000	-0.878407	-0.574720	0.309902	0.112041	0.033081	0.165067	0.581198
Age	-0.878407	1.000000	0.512735	-0.157904	-0.099659	0.032573	-0.120706	-0.464299
KM	-0.574720	0.512735	1.000000	-0.335285	-0.093825	-0.081248	0.299993	-0.026271
HP	0.309902	-0.157904	-0.335285	1.000000	0.064749	0.013755	0.053758	0.086737
MetColor	0.112041	-0.099659	-0.093825	0.064749	1.000000	-0.013973	0.029189	0.057142
Automatic	0.033081	0.032573	-0.081248	0.013755	-0.013973	1.000000	-0.069321	0.057249
CC	0.165067	-0.120706	0.299993	0.053758	0.029189	-0.069321	1.000000	0.651450
Weight	0.581198	-0.464299	-0.026271	0.086737	0.057142	0.057249	0.651450	1.000000

Data Visualization

Scatter Plot

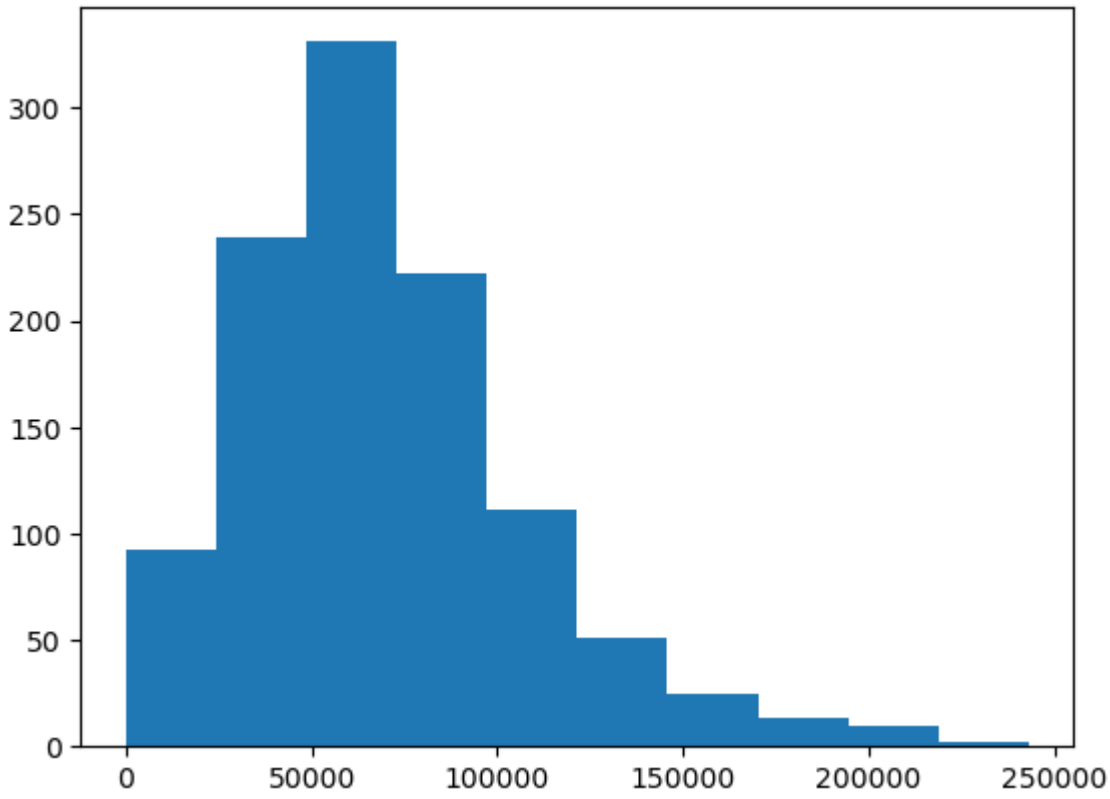
In [34]:
```
plt.scatter(df['Age'], df['Price'], c = 'red')
plt.title("Scatter plot of Price vs Age of the cars")
plt.xlabel('Age (months)')
plt.ylabel('Price (Euros)')
plt.show()
```



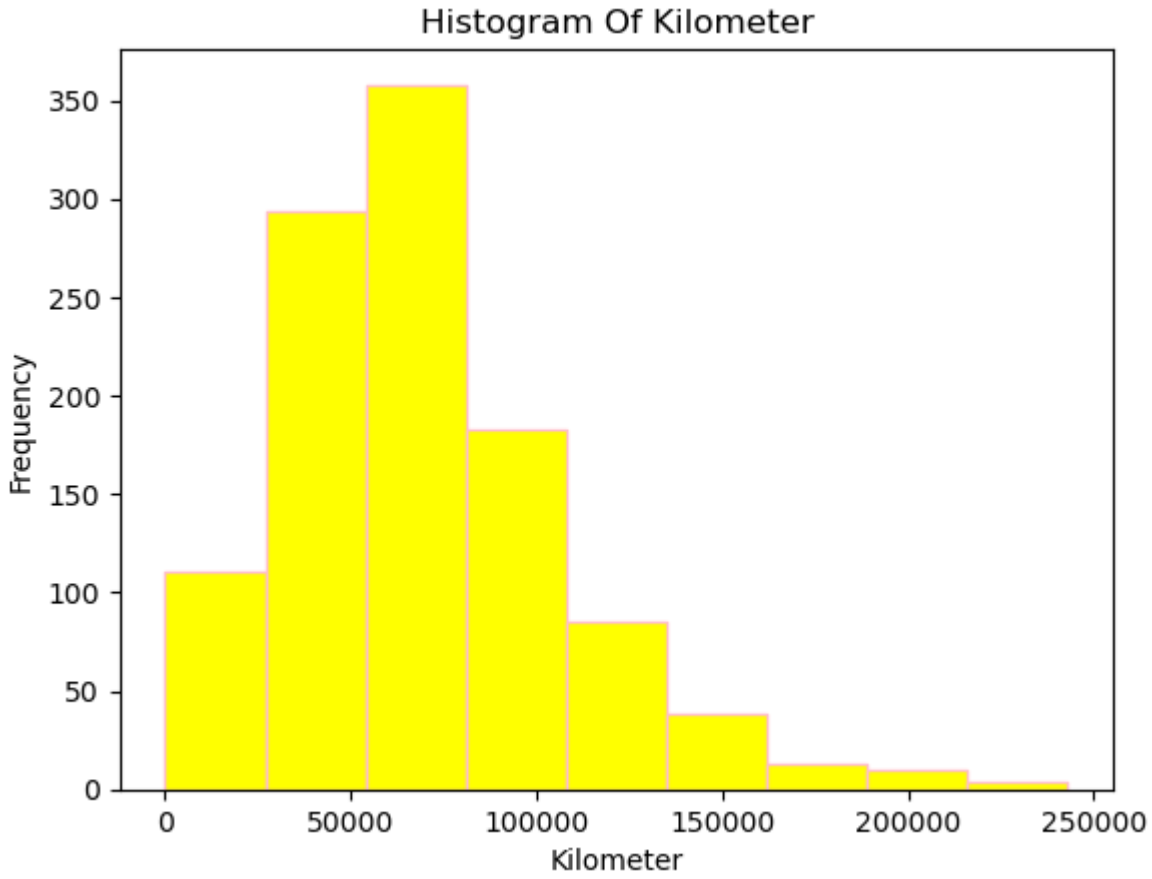
Histogram

In [35]:
```
plt.hist(df['KM'])
plt
```

Out[35]:
```
<module 'matplotlib.pyplot' from 'C:\\ProgramData\\anaconda3\\Lib\\site-packages\\matplotlib\\pyplot.py'>
```

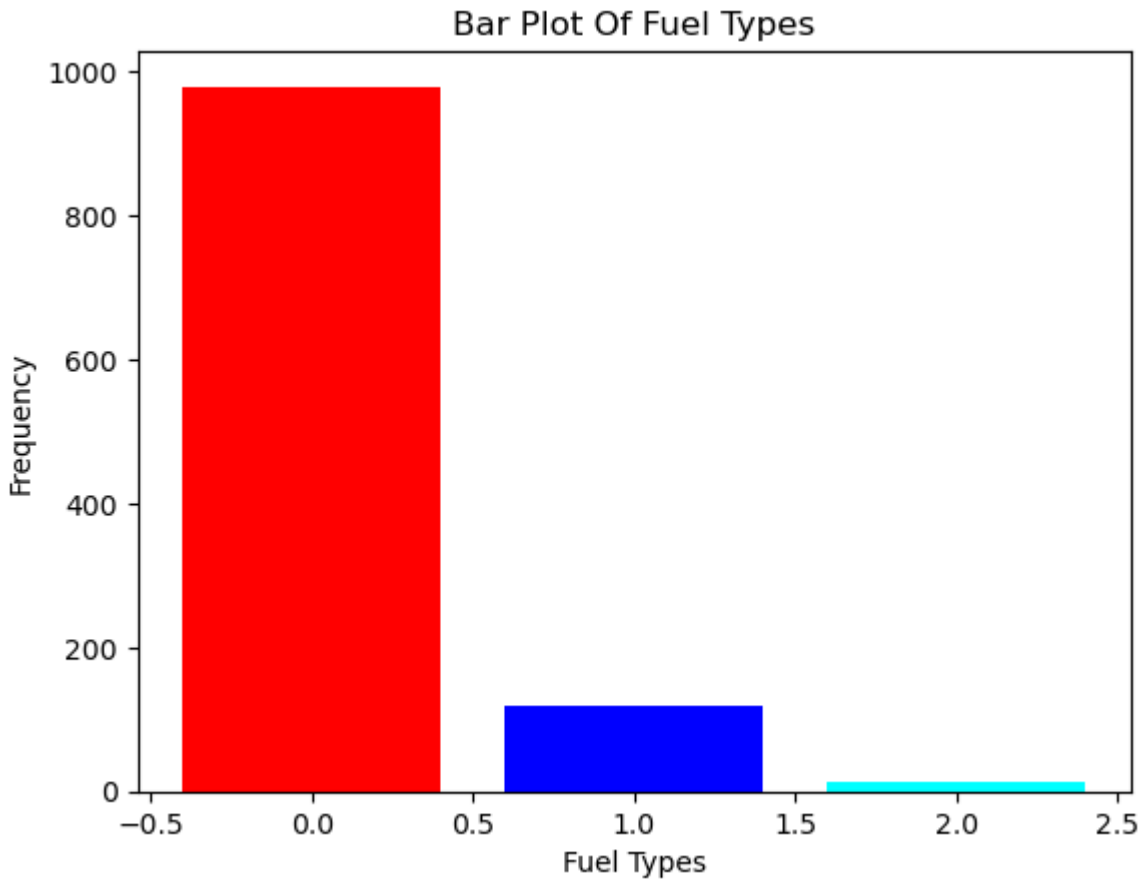


In [36]:
```
plt.hist(df['KM'],
    color = 'yellow' ,
    edgecolor = 'pink' ,
    bins = 9 )
plt.title("Histogram Of Kilometer")
plt.xlabel('Kilometer')
plt.ylabel('Frequency')
plt.show()
```



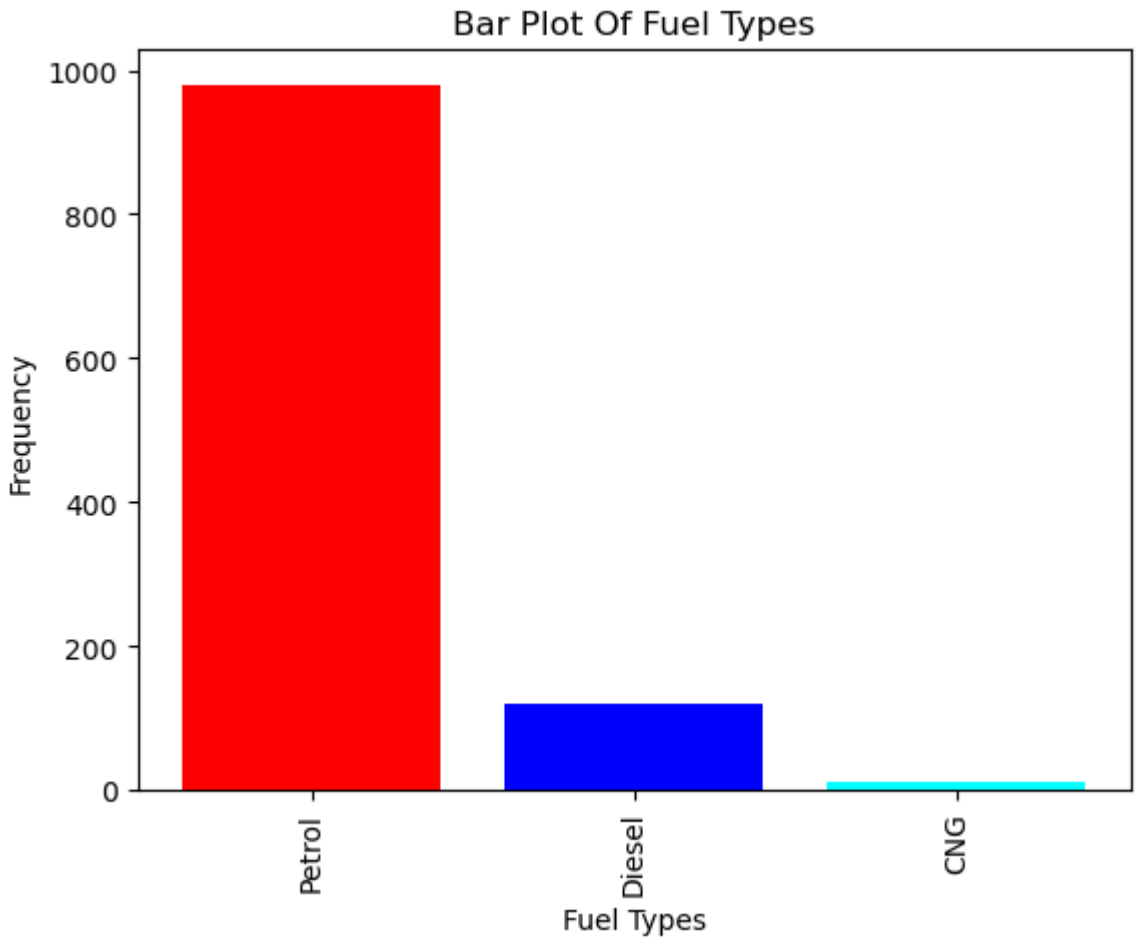
In [37]:
```
counts = [979, 120, 12]
fuelType = ('Petrol', 'Diesel', 'CNG')
index = np.arange(len(fuelType))

plt.bar(index, counts, color=['red', 'blue', 'cyan'])
plt.title('Bar Plot Of Fuel Types')
plt.xlabel('Fuel Types')
plt.ylabel('Frequency')
plt.show()
```



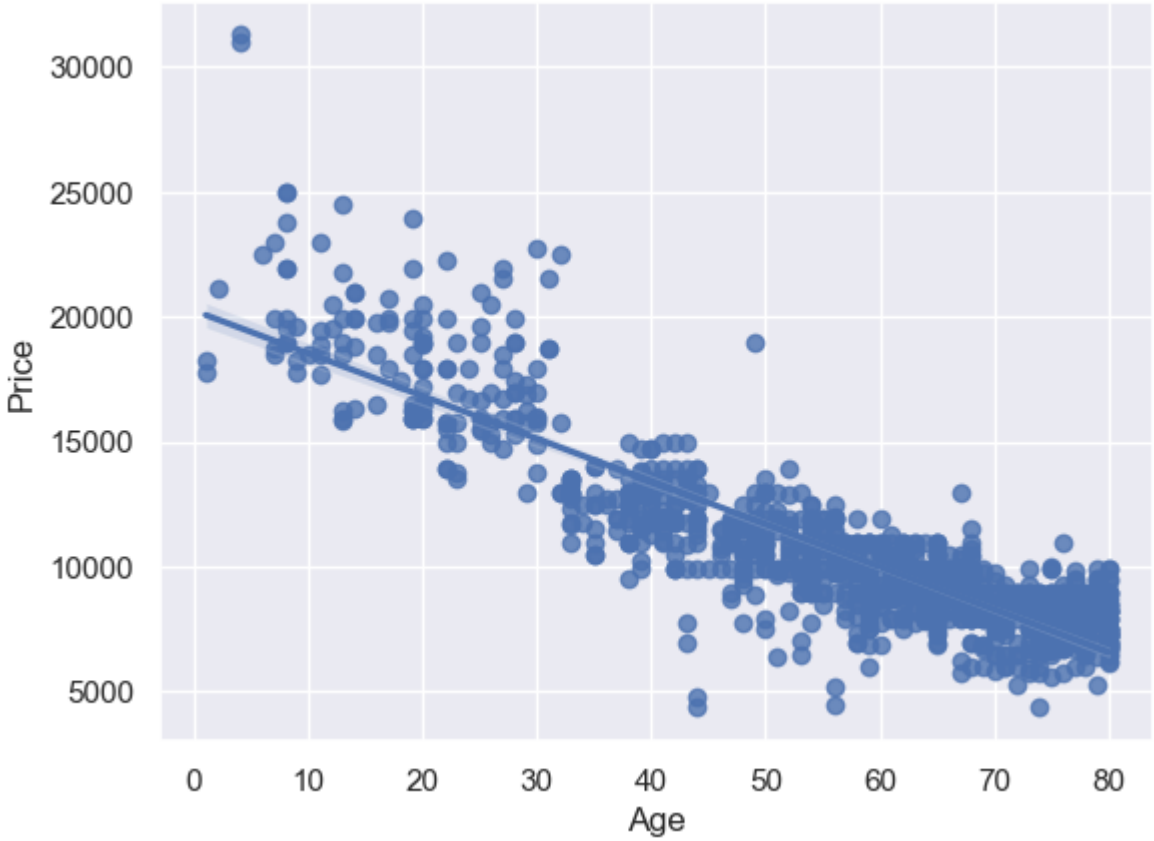
In [38]:
```
counts = [979, 120, 12]
fuelType = ('Petrol', 'Diesel', 'CNG')
index = np.arange(len(fuelType))

plt.bar(index, counts, color=['red', 'blue', 'cyan'])
plt.title('Bar Plot Of Fuel Types')
plt.xlabel('Fuel Types')
plt.ylabel('Frequency')
plt.xticks(index, fuelType,rotation = 90)
plt.show()
```



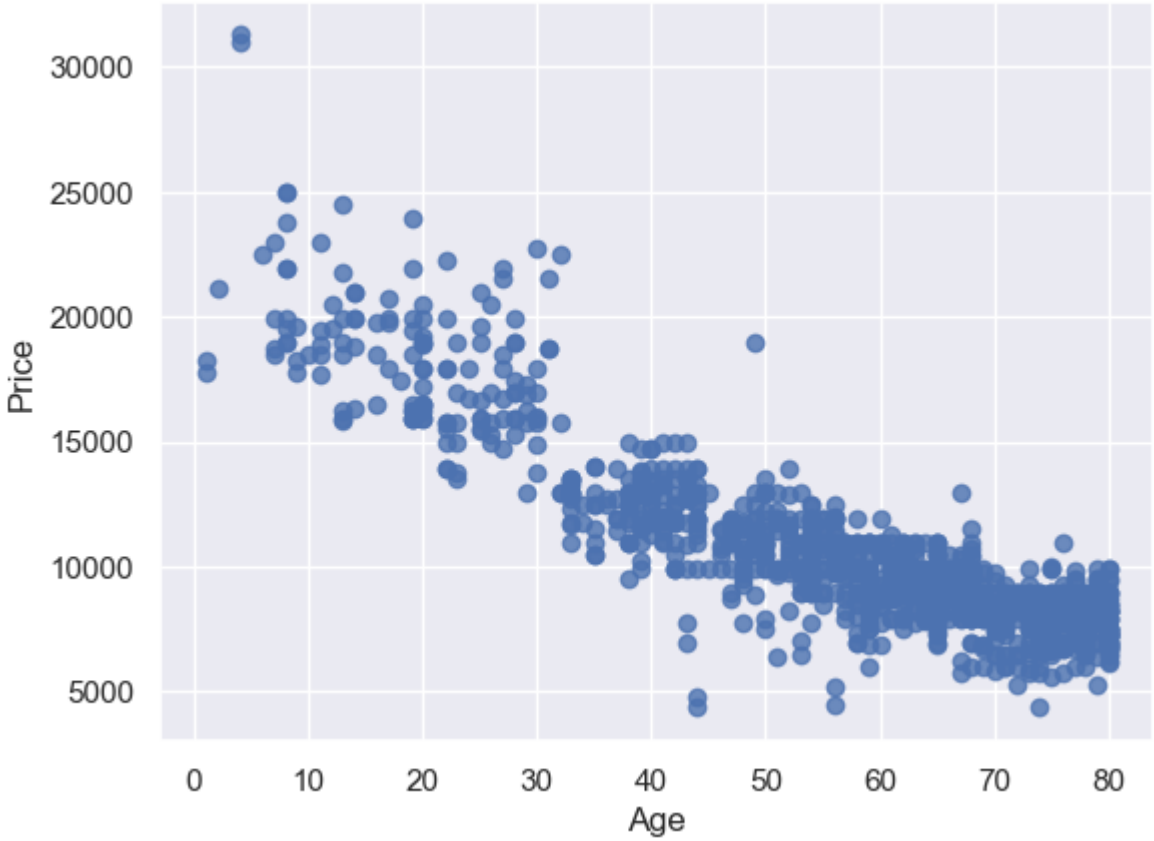
In [39]:
```
sns.set(style = "darkgrid")
sns.regplot( x = df['Age'] , y = df['Price'])
```

Out[39]:
```
<Axes: xlabel='Age', ylabel='Price'>
```



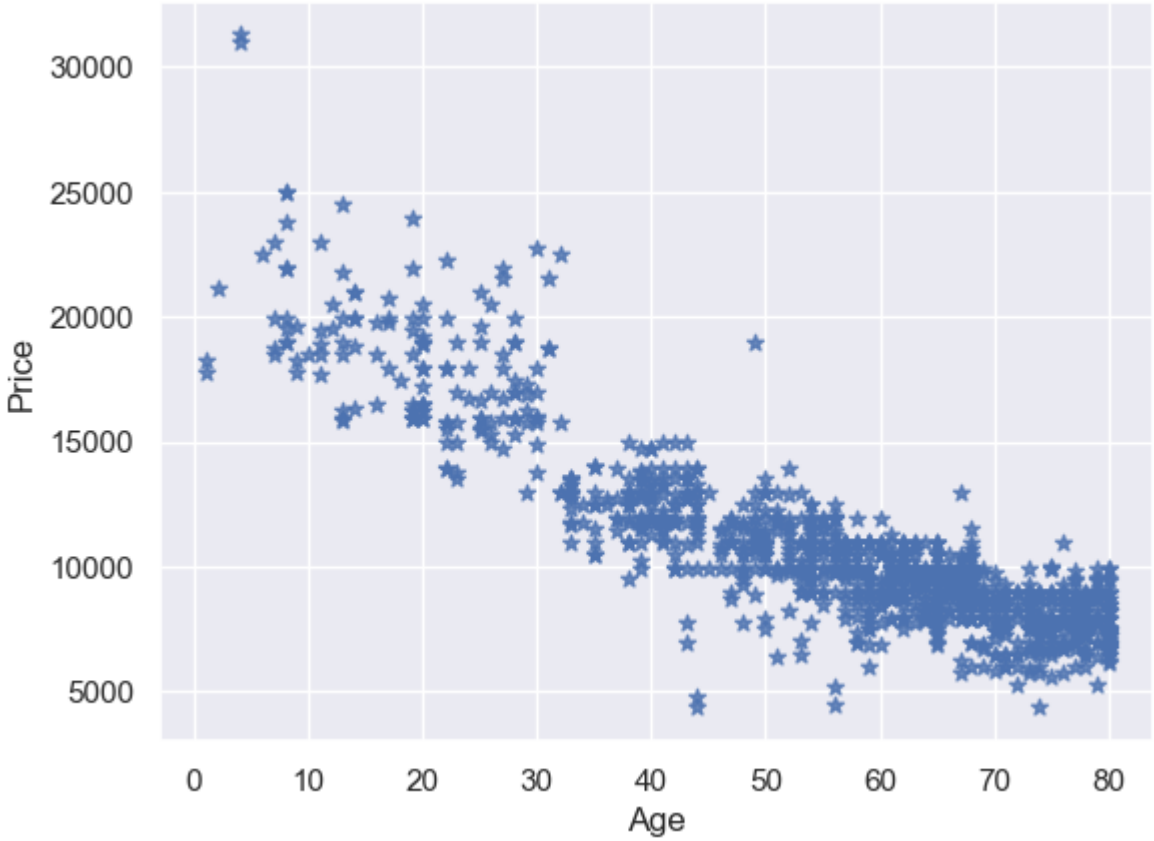
In [40]:
```
sns.set(style = "darkgrid")
sns.regplot( x = df['Age'] , y = df['Price'] ,
            fit_reg = False)
```

Out[40]:
```
<Axes: xlabel='Age', ylabel='Price'>
```



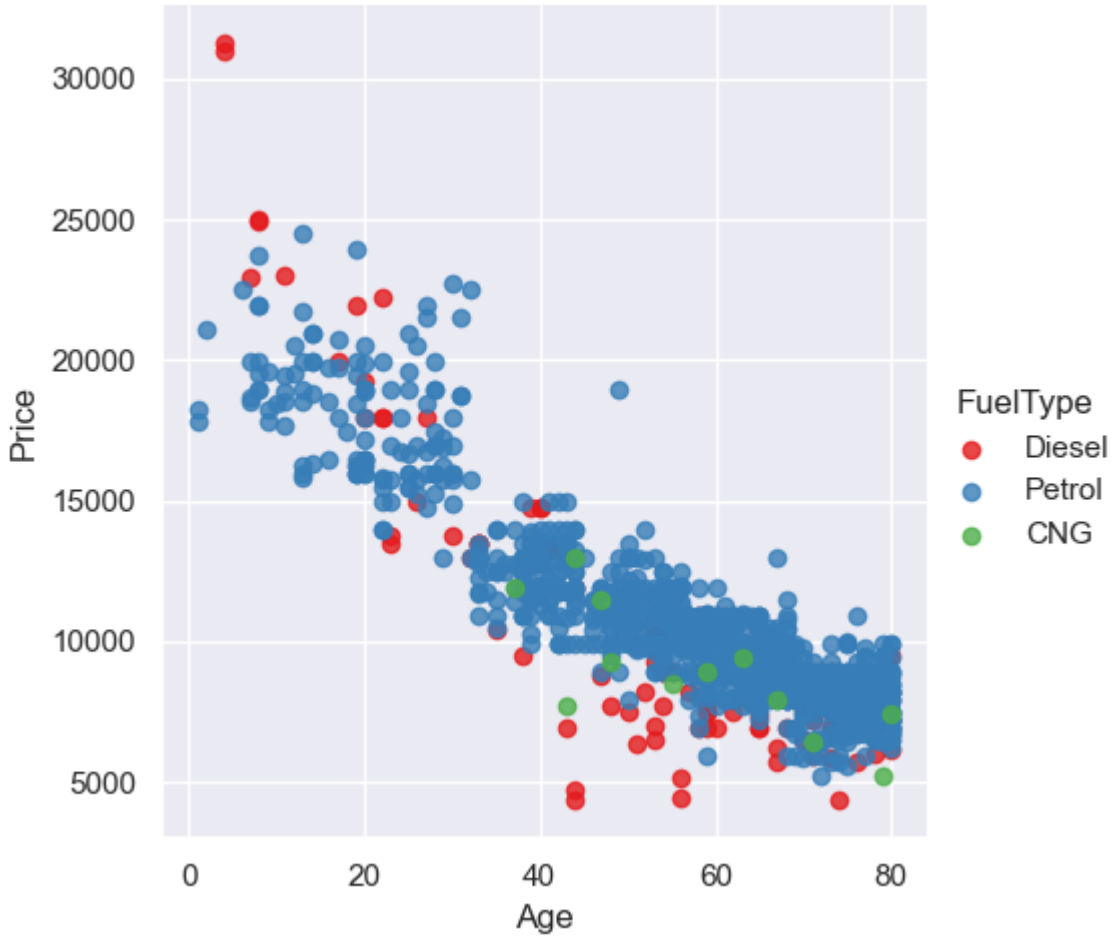
In [41]:
```
sns.set(style = "darkgrid")
sns.regplot( x = df['Age'] , y = df['Price'] ,
            marker = "*" , fit_reg = False)
```

Out[41]:
```
<Axes: xlabel='Age', ylabel='Price'>
```



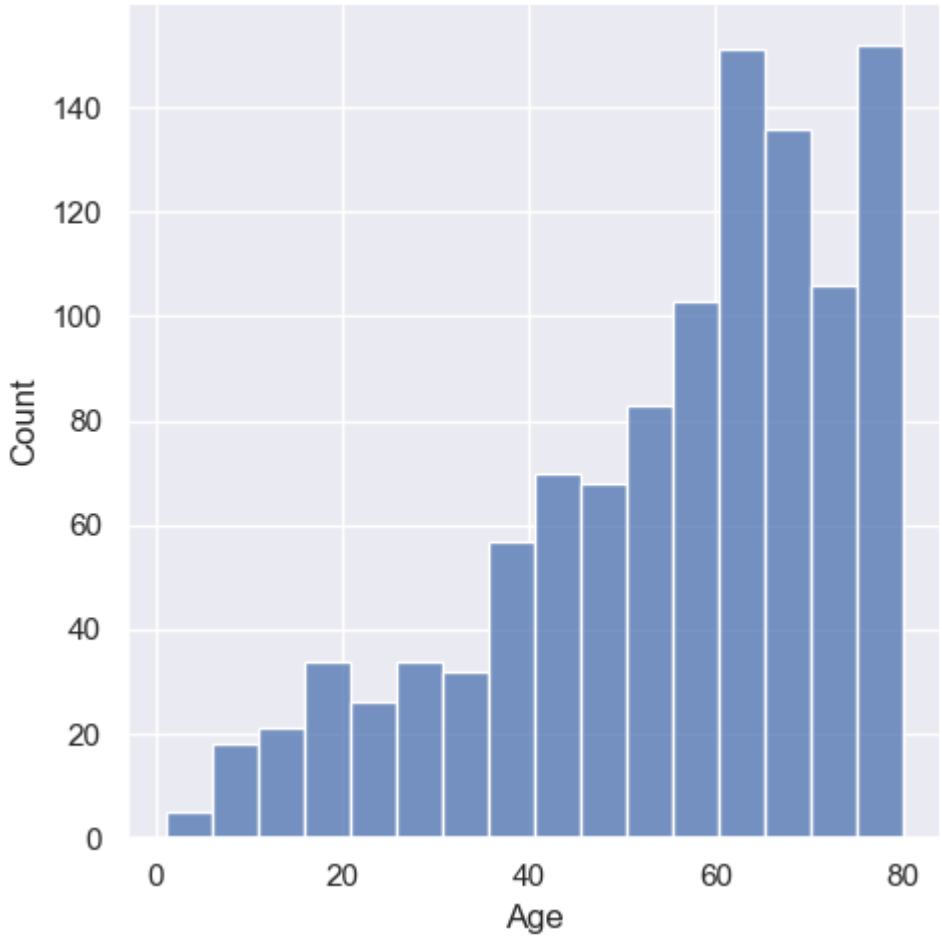
In [42]:
```
sns.lmplot(x='Age', y='Price', data=df,
    fit_reg=False, hue='FuelType',
    legend=True, palette="Set1")
```

Out[42]:
```
<seaborn.axisgrid.FacetGrid at 0x1f3c707b090>
```

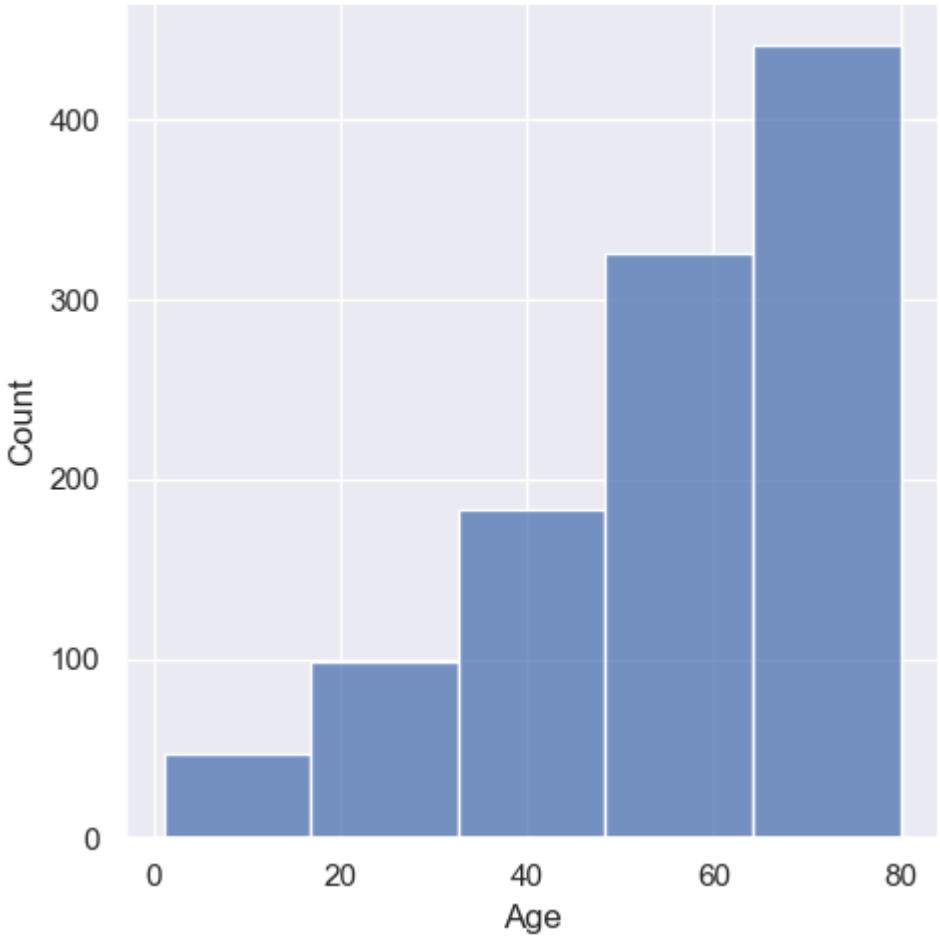


Histogram

In [43]:
```
sns.displot(df['Age'],kde=False)
plt.show()
```



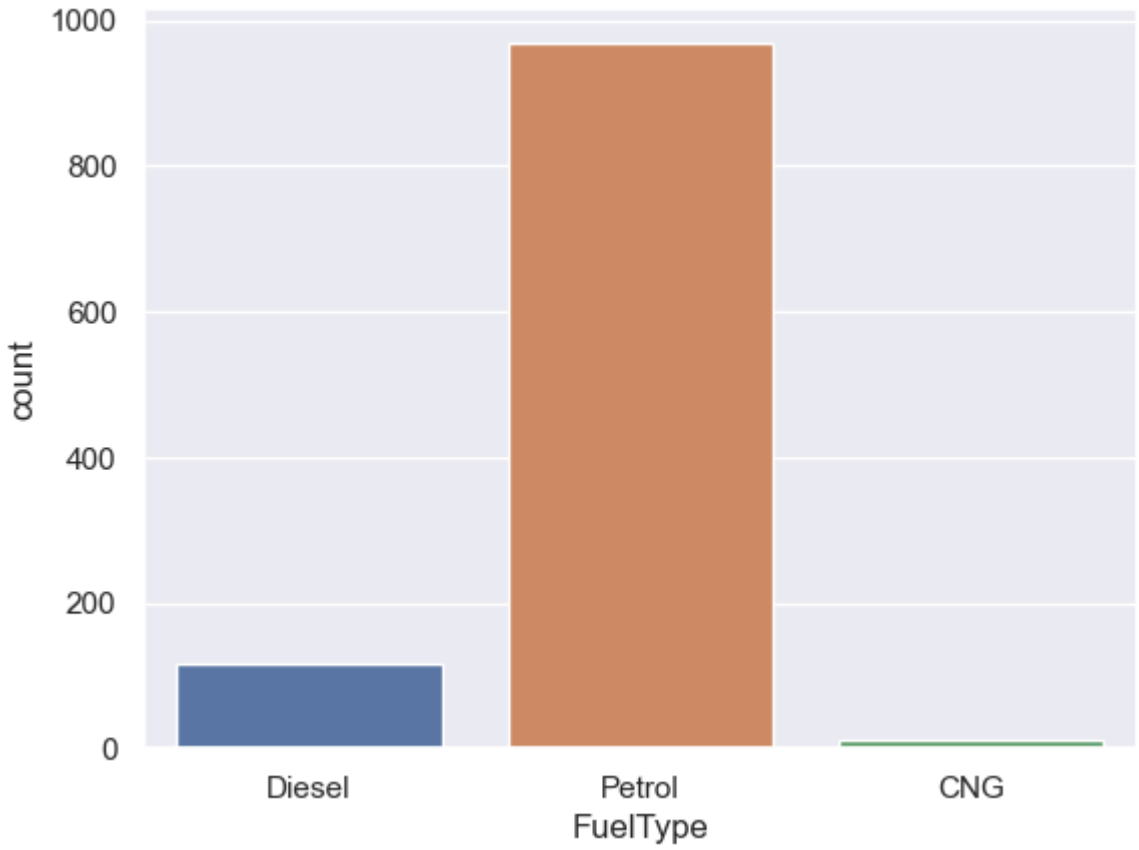
In [44]:
```
sns.displot(df['Age'],kde=False , bins = 5)
plt.show()
```



Bar Plot

In [45]:
```
sns.countplot(x = "FuelType" , data = df)
```

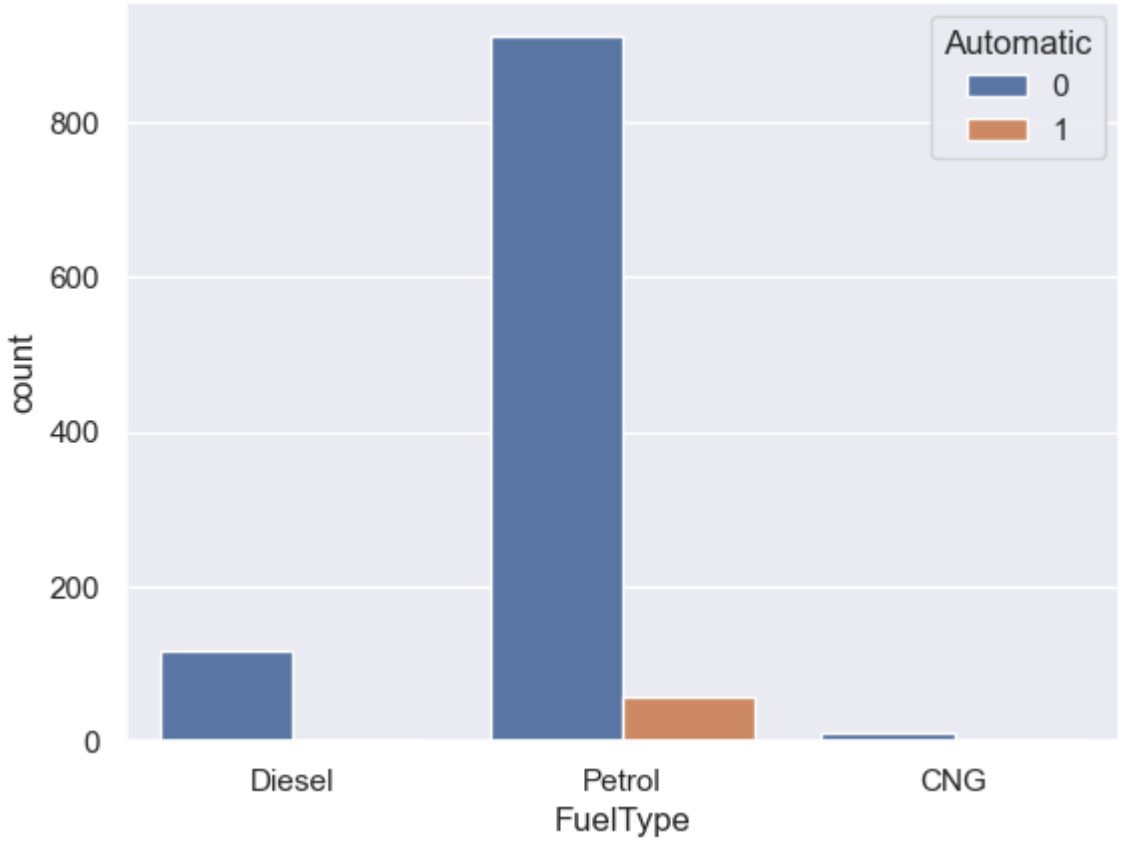
Out[45]:
```
<Axes: xlabel='FuelType', ylabel='count'>
```



Grouped Bar Plot

In [46]:
```
sns.countplot(x = "FuelType" , data = df , hue = "Automatic")
```

Out[46]:
```
<Axes: xlabel='FuelType', ylabel='count'>
```



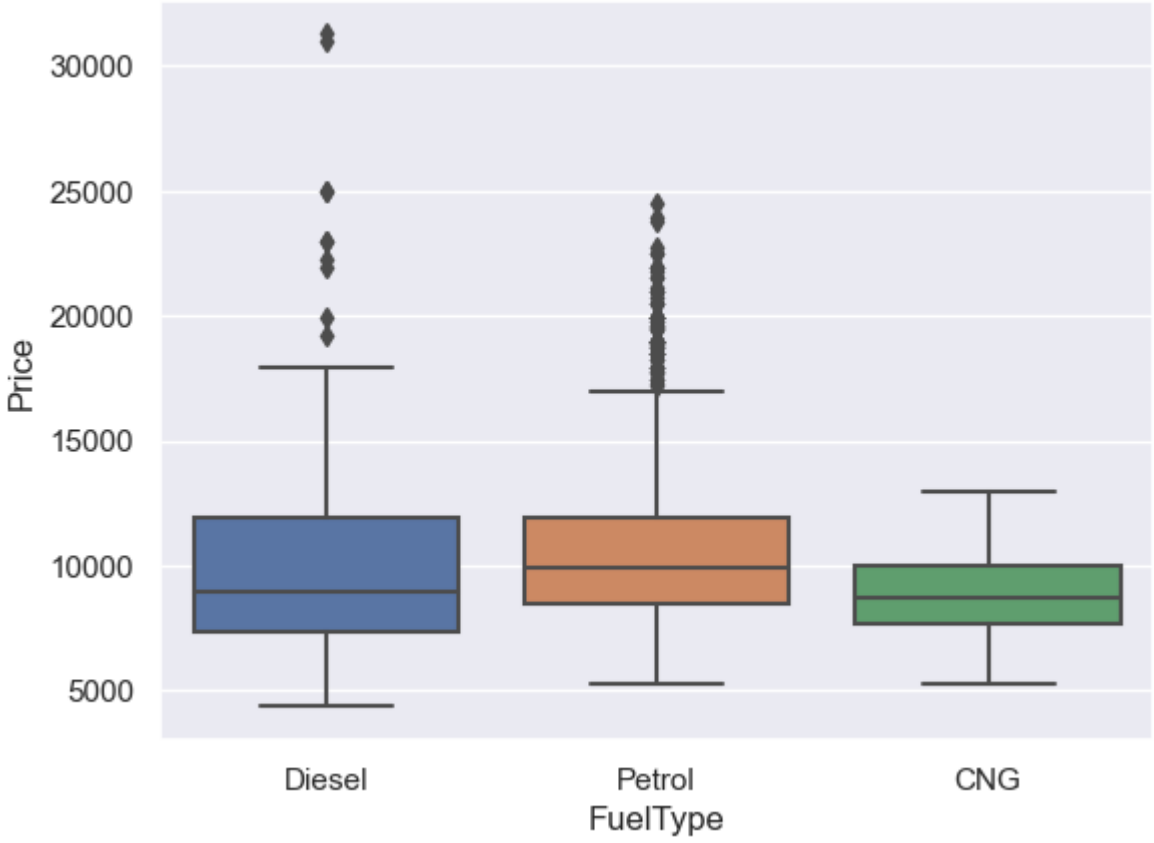
Box and Whiskers Plot - Numerical Variable

In [47]:
```
sns.boxplot(y = df["Price"])
```

Out[47]:
```
<Axes: ylabel='Price'>
```

In [48]:
```
sns.boxplot(x = df["FuelType"], y = df["Price"])
```

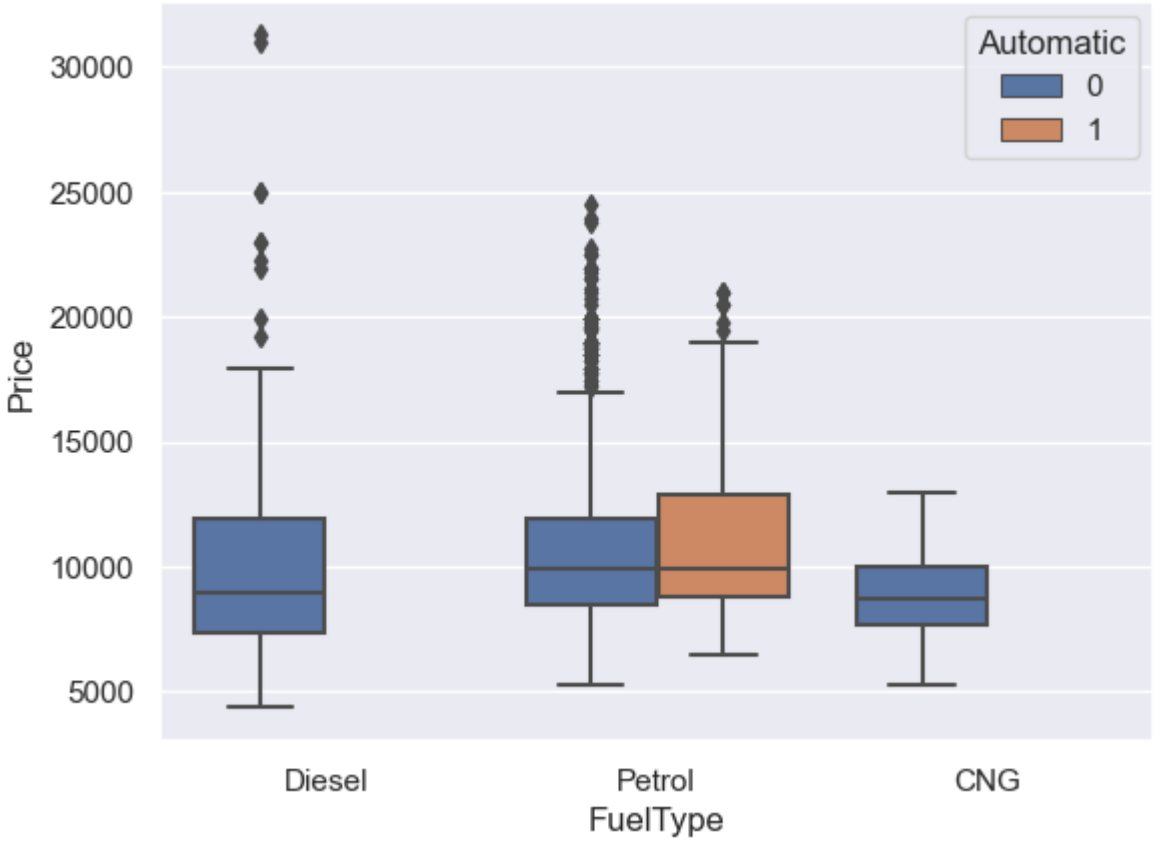
Out[48]:
```
<Axes: xlabel='FuelType', ylabel='Price'>
```



Grouped Box and Whiskers Plot

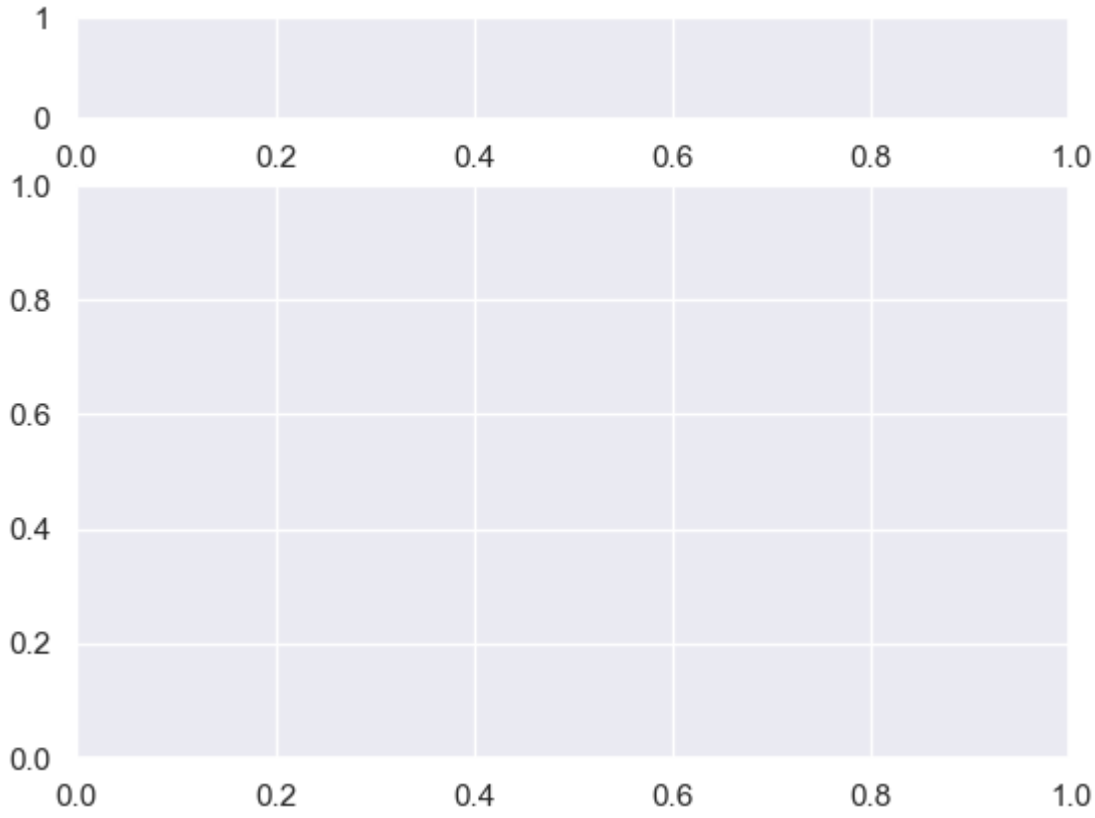
In [49]:
```
sns.boxplot(x = "FuelType", y = df["Price"],
            hue = "Automatic" , data = df)
```

Out[49]:
```
<Axes: xlabel='FuelType', ylabel='Price'>
```



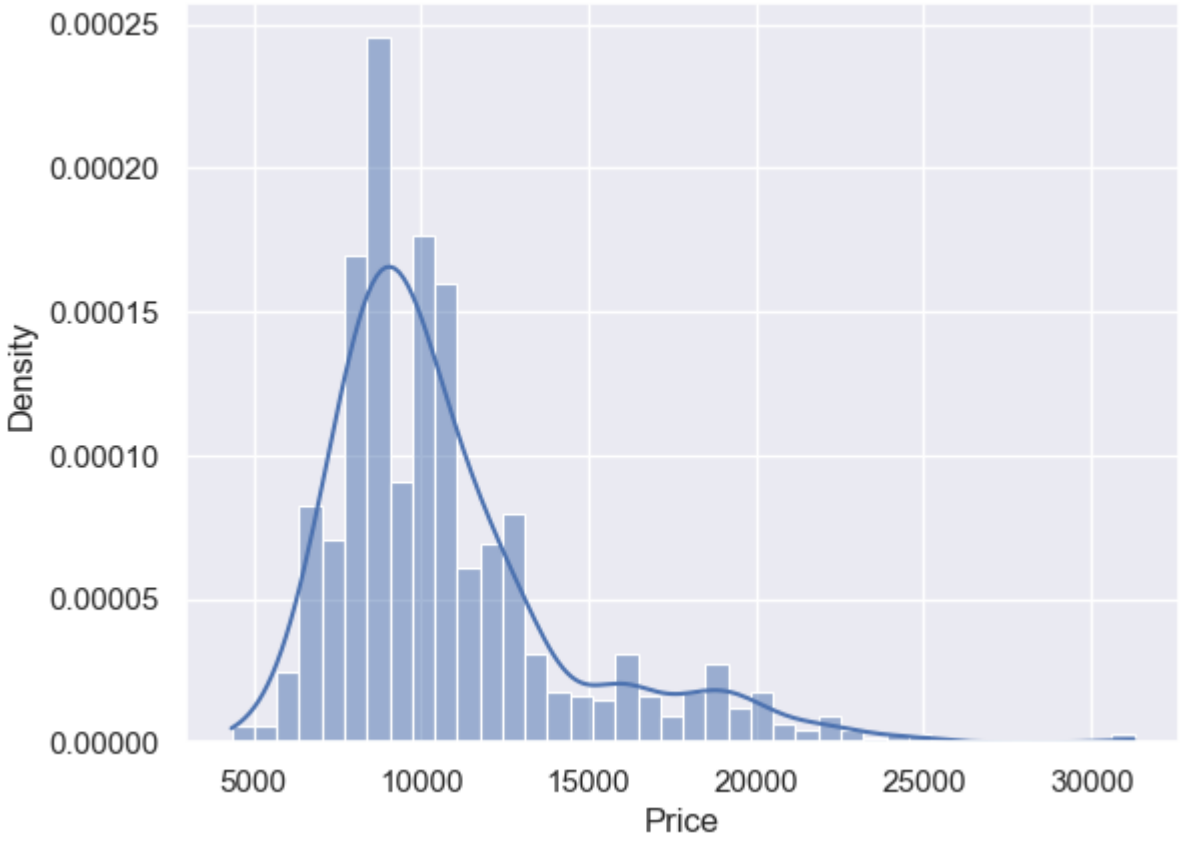
Box-Whiskers Plot and Histogram

In [50]:
```
f,(ax_box , ax_hist) = plt.subplots(2, gridspec_kw = {"height_ratios" : (.15 , .85)})
```

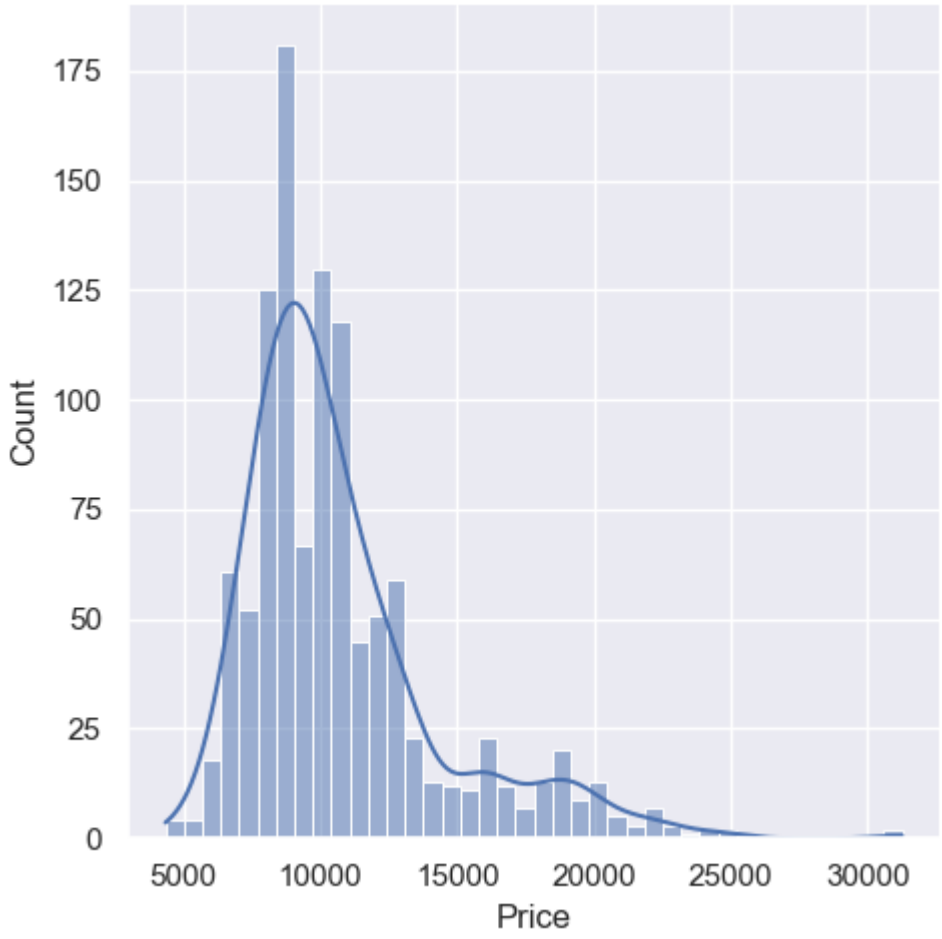


In [51]:
```
sns.histplot(df["Price"], kde=True, stat="density")
```

Out[51]:
```
<Axes: xlabel='Price', ylabel='Density'>
```

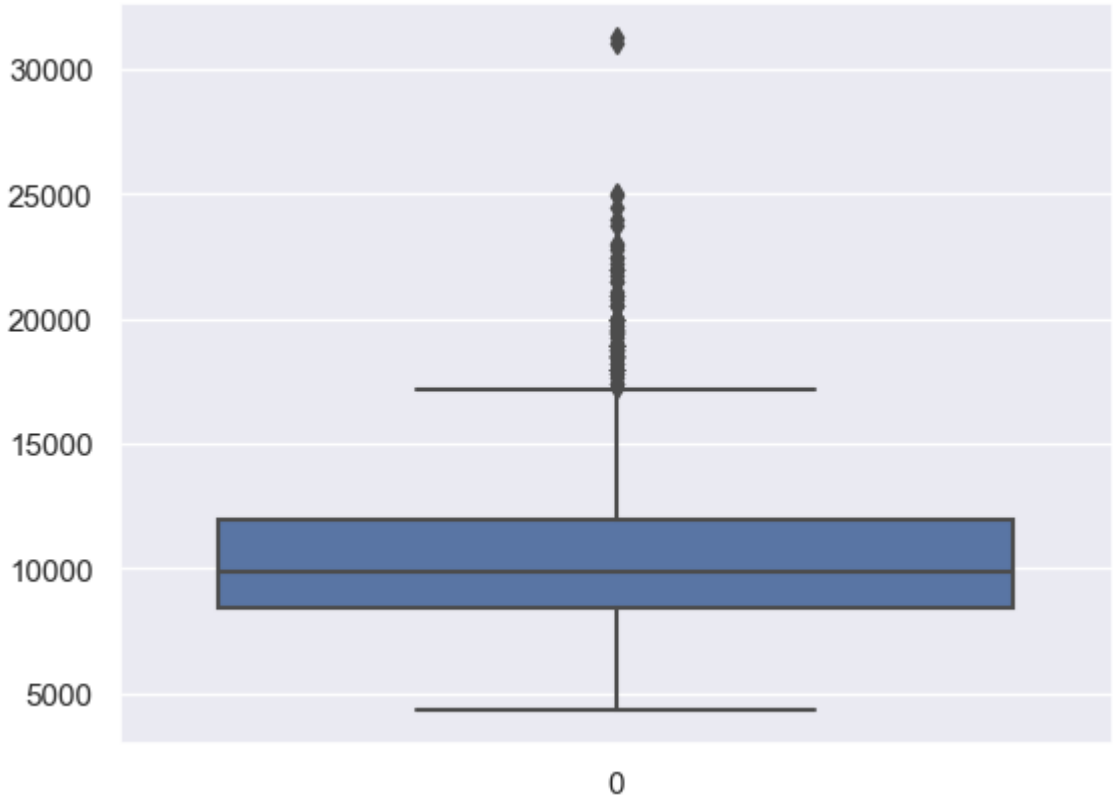


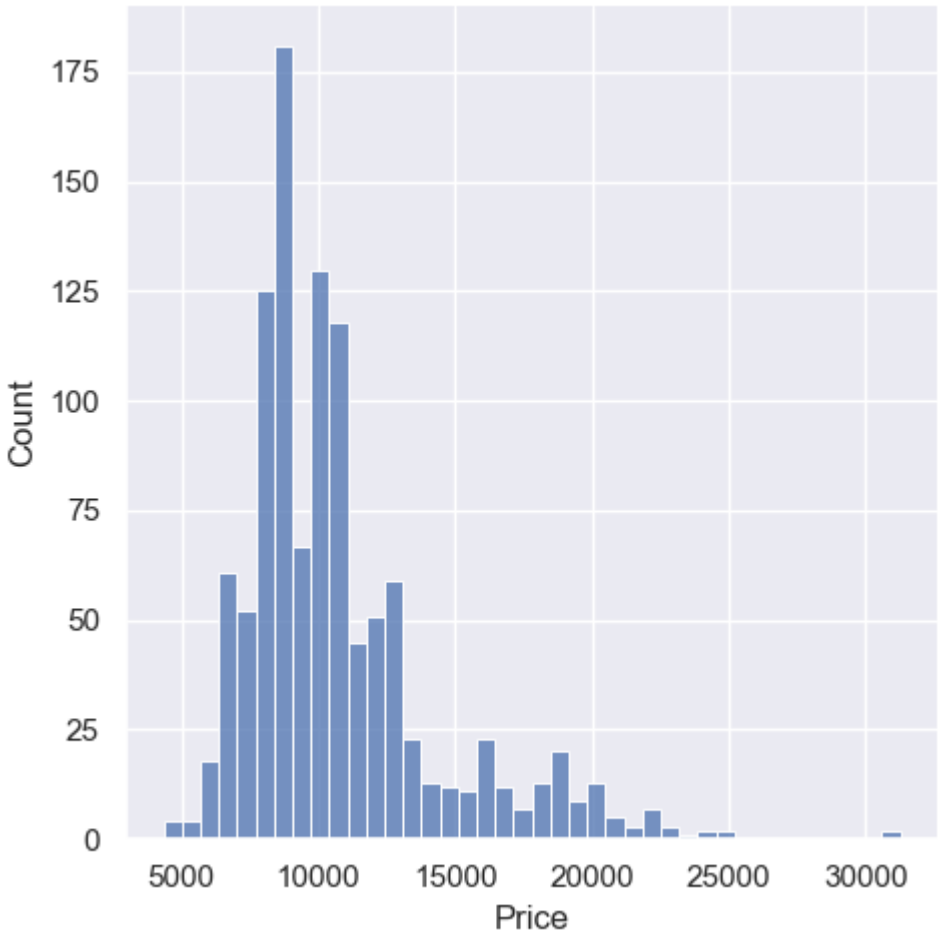
In [52]:
```
sns.displot(df["Price"], kde=True)
plt.show()
```



Now , add create two plots

In [53]:
```
sns.boxplot(df["Price"])
sns.displot(df["Price"], kde = False)
plt.show()
```


30000
25000
20000
15000
10000
5000
0


175
150
125
100
75
50
25
0
Count
5000
10000
15000
20000
25000
30000
Price

Pairwise Plots

In [54]:
```
sns.pairplot(df, kind = "scatter" , hue = "FuelType")
plt.show()
```

Dealing with Missing Data

In [55]:
```
df.isna().sum()
```

Out[55]:
```
Price        0
Age          0
KM           0
FuelType     0
HP           0
MetColor     0
Automatic    0
CC           0
Doors        0
Weight       0
dtype: int64
```

In [56]:
```
df.isnull().sum()
```

Out[56]:
```
Price        0
Age          0
KM           0
FuelType     0
HP           0
MetColor     0
Automatic    0
CC           0
Doors        0
Weight       0
dtype: int64
```

In [57]:
```
df.describe()
```

Out[57]:

	Price	Age	KM	HP	MetColor	Automatic	CC
count	1096.000000	1096.000000	1096.000000	1096.000000	1096.000000	1096.000000	1096.000000
mean	10735.937044	55.661496	69268.826642	101.806569	0.673358	0.052920	1568.863139
std	3636.716945	18.699777	38070.667467	15.034116	0.469199	0.223975	184.386960
min	4350.000000	1.000000	1.000000	69.000000	0.000000	0.000000	1300.000000
25%	8450.000000	43.000000	43590.500000	90.000000	0.000000	0.000000	1400.000000
50%	9900.000000	60.000000	63393.500000	110.000000	1.000000	0.000000	1600.000000
75%	11950.000000	70.000000	88031.750000	110.000000	1.000000	0.000000	1600.000000
max	31275.000000	80.000000	243000.000000	192.000000	1.000000	1.000000	2000.000000

In [58]:
```
df["Age"].mean()
```

Out[58]:
```
55.66149635036496
```

In [59]:
```
df['Age'].fillna(df['Age'].mean() ,
                 inplace = True)
df
```

Out[59]:

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986.0	Diesel	90.0	1.0	0	2000	three	1165
1	13750	23.0	72937.0	Diesel	90.0	1.0	0	2000	3	1165
3	14950	26.0	48000.0	Diesel	90.0	0.0	0	2000	3	1165
4	13750	30.0	38500.0	Diesel	90.0	0.0	0	2000	3	1170
5	12950	32.0	61000.0	Diesel	90.0	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1423	7950	80.0	35821.0	Petrol	86.0	0.0	1	1300	3	1015
1424	7750	73.0	34717.0	Petrol	86.0	0.0	0	1300	3	1015
1429	8950	78.0	24000.0	Petrol	86.0	1.0	1	1300	5	1065
1430	8450	80.0	23000.0	Petrol	86.0	0.0	0	1300	3	1015
1435	6950	76.0	1.0	Petrol	110.0	0.0	0	1600	5	1114

1096 rows × 10 columns

In [60]:
```
df["KM"].median()
```

Out[60]:
```
63393.5
```

In [61]:
```
df["HP"].mean()
```

Out[61]:
```
101.80656934306569
```

In [62]:
```
df['HP'].fillna(df['HP'].mean() ,
                inplace = True)
df
```

Out[62]:

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986.0	Diesel	90.0	1.0	0	2000	three	1165
1	13750	23.0	72937.0	Diesel	90.0	1.0	0	2000	3	1165
3	14950	26.0	48000.0	Diesel	90.0	0.0	0	2000	3	1165
4	13750	30.0	38500.0	Diesel	90.0	0.0	0	2000	3	1170
5	12950	32.0	61000.0	Diesel	90.0	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1423	7950	80.0	35821.0	Petrol	86.0	0.0	1	1300	3	1015
1424	7750	73.0	34717.0	Petrol	86.0	0.0	0	1300	3	1015
1429	8950	78.0	24000.0	Petrol	86.0	1.0	1	1300	5	1065
1430	8450	80.0	23000.0	Petrol	86.0	0.0	0	1300	3	1015
1435	6950	76.0	1.0	Petrol	110.0	0.0	0	1600	5	1114

1096 rows × 10 columns

In [63]:
```
df.isnull().sum()
```

Out[63]:
```
Price        0
Age          0
KM           0
FuelType     0
HP           0
MetColor     0
Automatic    0
CC           0
Doors        0
Weight       0
dtype: int64
```

AKASH KUMAR

TRUBA COLLGE OF SCIENCE & TECHNOLOGY , BHOPAL M.P.