BMW Car Task

1.Setup

Import

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\sujit\tanishka\lib\site-package
s (2.2.3)Note: you may need to restart the kernel to use updated packages.

Requirement already satisfied: numpy>=1.26.0 in c:\users\sujit\tanishka\lib\site-
packages (from pandas) (2.1.3)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\sujit\tanishka
\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\sujit\tanishka\lib\site-p
ackages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\sujit\tanishka\lib\site
-packages (from pandas) (2025.2)
Requirement already satisfied: six>=1.5 in c:\users\sujit\tanishka\lib\site-packa
ges (from python-dateutil>=2.8.2->pandas) (1.17.0)
```

In [3]:
```
pip install openpyxl
```

```
Requirement already satisfied: openpyxl in c:\users\sujit\tanishka\lib\site-packa
ges (3.1.5)
Requirement already satisfied: et-xmlfile in c:\users\sujit\tanishka\lib\site-pac
kages (from openpyxl) (1.1.0)
Note: you may need to restart the kernel to use updated packages.
```

In [4]:
```
pip install numpy
```

```
Requirement already satisfied: numpy in c:\users\sujit\tanishka\lib\site-packages
(2.1.3)
Note: you may need to restart the kernel to use updated packages.
```

2. Loading Different Data Formats Into a Pandas Data frame

Reading csv file

In [5]:
```
df = pd.read_csv('C:\\Users\\Sujit\\OneDrive\\Desktop\\pandas2\\Bmw car\\bmw_car
```

In [6]:
```
df.head()
```

Out[6]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Power_BHP	T
0	1	X5	xDrive	2019	Electric	Manual	1496	395.18	
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	312.78	
2	3	Z4	M Sport	2012	Electric	Semi-Automatic	4395	604.45	
3	4	X7	M Sport	2013	Hybrid	Semi-Automatic	4395	383.04	
4	5	X7	xDrive	2010	Electric	Manual	2993	315.78	

In [7]:

```
df.tail()
```

Out[7]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Pow
99995	99996	Z4	Sport Line	2014	Hybrid	Semi-Automatic	1998	
99996	99997	Z4	Sport Line	2021	Electric	Automatic	1496	
99997	99998	M4	Standard	2012	Hybrid	Automatic	2993	
99998	99999	X7	Luxury Line	2015	Electric	Manual	4395	
99999	100000	520d	Luxury Line	2020	Electric	Semi-Automatic	1496	

Read csv file from URL

In [8]:

```
url="C:\\Users\\Sujit\\OneDrive\\Desktop\\pandas2\\Bmw car\\bmw_car_data.csv"
df_url = pd.read_csv(url)
df_url
```

Out[8]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Pow
0	1	X5	xDrive	2019	Electric	Manual	1496	
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	
2	3	Z4	M Sport	2012	Electric	Semi-Automatic	4395	
3	4	X7	M Sport	2013	Hybrid	Semi-Automatic	4395	
4	5	X7	xDrive	2010	Electric	Manual	2993	
...	...	...	...	...	...	...	...	
99995	99996	Z4	Sport Line	2014	Hybrid	Semi-Automatic	1998	
99996	99997	Z4	Sport Line	2021	Electric	Automatic	1496	
99997	99998	M4	Standard	2012	Hybrid	Automatic	2993	
99998	99999	X7	Luxury Line	2015	Electric	Manual	4395	
99999	100000	520d	Luxury Line	2020	Electric	Semi-Automatic	1496	

100000 rows × 15 columns

3.Data Preprocessing

3.1 Data Exploring

Retriving rows from data frame

In [9]:

```
df.head(10)
```

Out[9]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Power_BHP
0	1	X5	xDrive	2019	Electric	Manual	1496	395.18
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	312.78
2	3	Z4	M Sport	2012	Electric	Semi-Automatic	4395	604.45
3	4	X7	M Sport	2013	Hybrid	Semi-Automatic	4395	383.04
4	5	X7	xDrive	2010	Electric	Manual	2993	315.78
5	6	X1	M Sport	2012	Petrol	Manual	2993	193.38
6	7	M3	Standard	2014	Hybrid	Semi-Automatic	4395	480.93
7	8	320d	Sport Line	2015	Electric	Semi-Automatic	1496	203.37
8	9	X3	Sport Line	2019	Electric	Manual	1998	609.33
9	10	i8	Luxury Line	2015	Petrol	Manual	1998	386.49

In [10]:

```
df.tail(10)
```

Out[10]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Pow
99990	99991	X5	xDrive	2013	Petrol	Automatic	1998	
99991	99992	X1	Luxury Line	2023	Hybrid	Automatic	1496	
99992	99993	Z4	M Sport	2021	Hybrid	Manual	1496	
99993	99994	X3	M Sport	2011	Hybrid	Manual	4395	
99994	99995	X1	xDrive	2015	Hybrid	Manual	1998	
99995	99996	Z4	Sport Line	2014	Hybrid	Semi-Automatic	1998	
99996	99997	Z4	Sport Line	2021	Electric	Automatic	1496	
99997	99998	M4	Standard	2012	Hybrid	Automatic	2993	
99998	99999	X7	Luxury Line	2015	Electric	Manual	4395	
99999	100000	520d	Luxury Line	2020	Electric	Semi-Automatic	1496	

In [11]:

```
df.sample(4)
```

Out[11]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Powe
45541	45542	X3	M Sport	2012	Petrol	Semi-Automatic	1496	
26816	26817	X3	Luxury Line	2018	Diesel	Automatic	2993	
41233	41234	M3	xDrive	2022	Diesel	Automatic	4395	
41588	41589	X7	Standard	2020	Hybrid	Semi-Automatic	1496	

Retriving information about dataframe

In [12]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 15 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   ID                   100000 non-null  int64  
 1   Model_Name           100000 non-null  object 
 2   Variant              100000 non-null  object 
 3   Year                 100000 non-null  int64  
 4   Fuel_Type            100000 non-null  object 
 5   Transmission         100000 non-null  object 
 6   Engine_CC            100000 non-null  int64  
 7   Power_BHP            100000 non-null  float64
 8   Torque_Nm            100000 non-null  float64
 9   Mileage_kmpl         100000 non-null  float64
 10  Price_Ex_Showroom    100000 non-null  float64
 11  Owner_Type           100000 non-null  object 
 12  Insurance_Valid_Till 100000 non-null  int64  
 13  Location             100000 non-null  object 
 14  Registration_State   100000 non-null  object 
dtypes: float64(4), int64(4), object(7)
memory usage: 11.4+ MB
```

display no. of rows and column

In [13]:

```
df.shape
```

Out[13]:

```
(100000, 15)
```

In [14]:

```
df.columns
```

Out[14]:

```
Index(['ID', 'Model_Name', 'Variant', 'Year', 'Fuel_Type', 'Transmission',
       'Engine_CC', 'Power_BHP', 'Torque_Nm', 'Mileage_kmpl',
       'Price_Ex_Showroom', 'Owner_Type', 'Insurance_Valid_Till', 'Location',
       'Registration_State'],
      dtype='object')
```

In [15]:

```
df['Model_Name'].head(2)
```

Out[15]:

```
0    X5
1    X5
Name: Model_Name, dtype: object
```

In [16]:

```
df[['Model_Name','ID','Year']].head(4)
```

Out[16]:

	Model_Name	ID	Year
0	X5	1	2019
1	X5	2	2016
2	Z4	3	2012
3	X7	4	2013

In [17]:

```
df[['Model_Name','ID','Year']].tail(10)
```

Out[17]:

	Model_Name	ID	Year
99990	X5	99991	2013
99991	X1	99992	2023
99992	Z4	99993	2021
99993	X3	99994	2011
99994	X1	99995	2015
99995	Z4	99996	2014
99996	Z4	99997	2021
99997	M4	99998	2012
99998	X7	99999	2015
99999	520d	100000	2020

Retreving a range of rows

In [18]:

```
df[6:14]
```

Out[18]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Power_BHP
6	7	M3	Standard	2014	Hybrid	Semi-Automatic	4395	480.93
7	8	320d	Sport Line	2015	Electric	Semi-Automatic	1496	203.37
8	9	X3	Sport Line	2019	Electric	Manual	1998	609.33
9	10	i8	Luxury Line	2015	Petrol	Manual	1998	386.49
10	11	i8	Sport Line	2024	Diesel	Manual	4395	292.13
11	12	X7	M Sport	2019	Hybrid	Automatic	1998	174.84
12	13	X1	Standard	2015	Petrol	Manual	2993	555.12
13	14	320d	Standard	2018	Petrol	Manual	2993	477.10

In [19]:

```
df:[5]
```

3.2 handling missing values

Display missing values

In [20]:

```
import pandas as pd
import numpy as np
```

In [21]:

```
df = pd.read_csv('C:\\Users\\Sujit\\OneDrive\\Desktop\\pandas2\\Bmw car\\bmw_car
df.head(10)
```

Out[21]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Power_BHP
0	1	X5	xDrive	2019	Electric	Manual	1496	395.18
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	312.78
2	3	Z4	M Sport	2012	Electric	Semi-Automatic	4395	604.45
3	4	X7	M Sport	2013	Hybrid	Semi-Automatic	4395	383.04
4	5	X7	xDrive	2010	Electric	Manual	2993	315.78
5	6	X1	M Sport	2012	Petrol	Manual	2993	193.38
6	7	M3	Standard	2014	Hybrid	Semi-Automatic	4395	480.93
7	8	320d	Sport Line	2015	Electric	Semi-Automatic	1496	203.37
8	9	X3	Sport Line	2019	Electric	Manual	1998	609.33
9	10	i8	Luxury Line	2015	Petrol	Manual	1998	386.49

In [22]:

```
df.isna()
```

Out[22]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Power_I
0	False	False	False	False	False	False	False	F
1	False	False	False	False	False	False	False	F
2	False	False	False	False	False	False	False	F
3	False	False	False	False	False	False	False	F
4	False	False	False	False	False	False	False	F
...	...	...	...	...	...	...	...	
99995	False	False	False	False	False	False	False	F
99996	False	False	False	False	False	False	False	F
99997	False	False	False	False	False	False	False	F
99998	False	False	False	False	False	False	False	F
99999	False	False	False	False	False	False	False	F

100000 rows × 15 columns

In [23]:

```
df.isna().sum()
```

Out[23]:

```
ID                    0
Model_Name            0
Variant               0
Year                  0
Fuel_Type             0
Transmission          0
Engine_CC             0
Power_BHP             0
Torque_Nm             0
Mileage_kmpl          0
Price_Ex_Showroom     0
Owner_Type            0
Insurance_Valid_Till  0
Location              0
Registration_State    0
dtype: int64
```

In [24]:

```
df
```

Out[24]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Pow
0	1	X5	xDrive	2019	Electric	Manual	1496	
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	
2	3	Z4	M Sport	2012	Electric	Semi-Automatic	4395	
3	4	X7	M Sport	2013	Hybrid	Semi-Automatic	4395	
4	5	X7	xDrive	2010	Electric	Manual	2993	
...	...	...	...	...	...	...	...	
99995	99996	Z4	Sport Line	2014	Hybrid	Semi-Automatic	1998	
99996	99997	Z4	Sport Line	2021	Electric	Automatic	1496	
99997	99998	M4	Standard	2012	Hybrid	Automatic	2993	
99998	99999	X7	Luxury Line	2015	Electric	Manual	4395	
99999	100000	520d	Luxury Line	2020	Electric	Semi-Automatic	1496	

100000 rows × 15 columns

4 Filter cars manufacture after 2020

In [25]:

```
df_filtered_sorted = df[(df['Year']>2020)].sort_values(by= 'Year',ascending=True
df_filtered_sorted
```

Out[25]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Powe
99840	99841	X1	xDrive	2021	Petrol	Manual	1496	
99850	99851	Z4	Luxury Line	2021	Hybrid	Automatic	1496	
99878	99879	X7	Luxury Line	2021	Electric	Manual	4395	
99893	99894	X1	xDrive	2021	Hybrid	Automatic	4395	
99	100	i8	xDrive	2021	Hybrid	Automatic	2993	
...	...	...	...	...	...	...	...	
50196	50197	X7	Luxury Line	2024	Diesel	Automatic	1998	
60	61	X5	xDrive	2024	Hybrid	Semi-Automatic	1998	
50200	50201	X5	Standard	2024	Hybrid	Semi-Automatic	1998	
99939	99940	X1	xDrive	2024	Diesel	Manual	2993	
50141	50142	Z4	Sport Line	2024	Electric	Manual	2993	

26720 rows × 15 columns

5 list unique car models and fule types

In [26]:

```
df['Fuel_Type'].unique()
```

Out[26]:

```
array(['Electric', 'Hybrid', 'Petrol', 'Diesel'], dtype=object)
```

In [27]:

```
df['Model_Name'].unique()
```

Out[27]:

```
array(['X5', 'Z4', 'X7', 'X1', 'M3', '320d', 'X3', 'i8', 'M4', '520d'],
      dtype=object)
```

6. count how many cars per fules type

In [28]:

```
df['Fuel_Type'].value_counts()
```

Out[28]:

```
Fuel_Type
Petrol      25125
Diesel      25089
Hybrid      24934
Electric    24852
Name: count, dtype: int64
```

In [29]:

```
df['Price_Ex_Showroom'].sort_values(ascending=False)
```

Out[29]:

```
71910    135.00
62468    135.00
86274    135.00
97582    135.00
57442    135.00
          ...  
39430     35.01
30975     35.01
29372     35.00
31867     35.00
99357     35.00
Name: Price_Ex_Showroom, Length: 100000, dtype: float64
```

In [30]:

```
df
```

Out[30]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Pow
0	1	X5	xDrive	2019	Electric	Manual	1496	
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	
2	3	Z4	M Sport	2012	Electric	Semi-Automatic	4395	
3	4	X7	M Sport	2013	Hybrid	Semi-Automatic	4395	
4	5	X7	xDrive	2010	Electric	Manual	2993	
...	...	...	...	...	...	...	...	
99995	99996	Z4	Sport Line	2014	Hybrid	Semi-Automatic	1998	
99996	99997	Z4	Sport Line	2021	Electric	Automatic	1496	
99997	99998	M4	Standard	2012	Hybrid	Automatic	2993	
99998	99999	X7	Luxury Line	2015	Electric	Manual	4395	
99999	100000	520d	Luxury Line	2020	Electric	Semi-Automatic	1496	

100000 rows × 15 columns

7.Filter only Automatic transmission BMWs

In [31]:

```
df_filtered_sorted = df[(df['Transmission']=='Automatic')].sort_values(by= 'Tran
df_filtered_sorted
```

Out[31]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Powe
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	
66467	66468	X5	xDrive	2018	Diesel	Automatic	1998	
66464	66465	X7	M Sport	2013	Petrol	Automatic	4395	
66462	66463	320d	Standard	2017	Hybrid	Automatic	2993	
66461	66462	i8	Sport Line	2020	Hybrid	Automatic	4395	
...	...	...	...	...	...	...	...	
33470	33471	X1	Luxury Line	2024	Diesel	Automatic	1998	
33469	33470	Z4	Luxury Line	2013	Diesel	Automatic	1998	
33466	33467	520d	M Sport	2017	Petrol	Automatic	1998	
33462	33463	M4	xDrive	2019	Hybrid	Automatic	2993	
99997	99998	M4	Standard	2012	Hybrid	Automatic	2993	

33665 rows × 15 columns

8. Replace missing values

In [32]:

```
df
```

Out[32]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Pow
0	1	X5	xDrive	2019	Electric	Manual	1496	
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	
2	3	Z4	M Sport	2012	Electric	Semi-Automatic	4395	
3	4	X7	M Sport	2013	Hybrid	Semi-Automatic	4395	
4	5	X7	xDrive	2010	Electric	Manual	2993	
...	...	...	...	...	...	...	...	
99995	99996	Z4	Sport Line	2014	Hybrid	Semi-Automatic	1998	
99996	99997	Z4	Sport Line	2021	Electric	Automatic	1496	
99997	99998	M4	Standard	2012	Hybrid	Automatic	2993	
99998	99999	X7	Luxury Line	2015	Electric	Manual	4395	
99999	100000	520d	Luxury Line	2020	Electric	Semi-Automatic	1496	

100000 rows × 15 columns

In [33]:

```
mean_Mileage_kmpl = df['Mileage_kmpl'].mean()
df['Mileage_kmpl'].fillna(mean_Mileage_kmpl, inplace=False)
```

Out[33]:

```
0        20.48
1         8.12
2        16.24
3         9.80
4        23.47
         ...  
99995     9.83
99996    13.45
99997    14.44
99998    14.93
99999    13.97
Name: Mileage_kmpl, Length: 100000, dtype: float64
```

In [34]:

```
print(df['Mileage_kmpl'].isnull().sum())  # Should now be 0
```

```
0
```

9. delete Rows

In [35]:

```
df_new = df.copy()
```

In [36]:

```
df_new.dropna(subset=['Engine_CC'],inplace=True)
df_new
```

Out[36]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Pow
0	1	X5	xDrive	2019	Electric	Manual	1496	
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	
2	3	Z4	M Sport	2012	Electric	Semi-Automatic	4395	
3	4	X7	M Sport	2013	Hybrid	Semi-Automatic	4395	
4	5	X7	xDrive	2010	Electric	Manual	2993	
...	...	...	...	...	...	...	...	
99995	99996	Z4	Sport Line	2014	Hybrid	Semi-Automatic	1998	
99996	99997	Z4	Sport Line	2021	Electric	Automatic	1496	
99997	99998	M4	Standard	2012	Hybrid	Automatic	2993	
99998	99999	X7	Luxury Line	2015	Electric	Manual	4395	
99999	100000	520d	Luxury Line	2020	Electric	Semi-Automatic	1496	

100000 rows × 15 columns

10. Grouping

In [37]:

```
df[['Price_Ex_Showroom']].groupby(df['Model_Name']).agg(['max'])
```

Out[37]:

	Price_Ex_Showroom
	max
Model_Name	
320d	135.00
520d	135.00
M3	135.00
M4	134.99
X1	135.00
X3	134.98
X5	134.98
X7	135.00
Z4	134.97
i8	134.98

11. Adding new column

In [38]:

```
print(df.columns)
```

```
Index(['ID', 'Model_Name', 'Variant', 'Year', 'Fuel_Type', 'Transmission',
       'Engine_CC', 'Power_BHP', 'Torque_Nm', 'Mileage_kmpl',
       'Price_Ex_Showroom', 'Owner_Type', 'Insurance_Valid_Till', 'Location',
       'Registration_State'],
      dtype='object')
```

In [39]:

```
df_col=df.copy()
df_col
```

Out[39]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Pow
0	1	X5	xDrive	2019	Electric	Manual	1496	
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	
2	3	Z4	M Sport	2012	Electric	Semi-Automatic	4395	
3	4	X7	M Sport	2013	Hybrid	Semi-Automatic	4395	
4	5	X7	xDrive	2010	Electric	Manual	2993	
...	...	...	...	...	...	...	...	
99995	99996	Z4	Sport Line	2014	Hybrid	Semi-Automatic	1998	
99996	99997	Z4	Sport Line	2021	Electric	Automatic	1496	
99997	99998	M4	Standard	2012	Hybrid	Automatic	2993	
99998	99999	X7	Luxury Line	2015	Electric	Manual	4395	
99999	100000	520d	Luxury Line	2020	Electric	Semi-Automatic	1496	

100000 rows × 15 columns

In [40]:

```
df['Car_Age'] = 2025- df['Year']
df[['Year','Car_Age']].head()
```

Out[40]:

	Year	Car_Age
0	2019	6
1	2016	9
2	2012	13
3	2013	12
4	2010	15

In [41]:

```
df
```

Out[41]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Pow
0	1	X5	xDrive	2019	Electric	Manual	1496	
1	2	X5	Luxury Line	2016	Electric	Automatic	1496	
2	3	Z4	M Sport	2012	Electric	Semi-Automatic	4395	
3	4	X7	M Sport	2013	Hybrid	Semi-Automatic	4395	
4	5	X7	xDrive	2010	Electric	Manual	2993	
...	...	...	...	...	...	...	...	
99995	99996	Z4	Sport Line	2014	Hybrid	Semi-Automatic	1998	
99996	99997	Z4	Sport Line	2021	Electric	Automatic	1496	
99997	99998	M4	Standard	2012	Hybrid	Automatic	2993	
99998	99999	X7	Luxury Line	2015	Electric	Manual	4395	
99999	100000	520d	Luxury Line	2020	Electric	Semi-Automatic	1496	

100000 rows × 16 columns

Filter Cars

In [42]:

```
df_filtered_sorted = df[(df['Price_Ex_Showroom']>60)].sort_values(by= 'Price_Ex_
df_filtered_sorted
```

Out[42]:

	ID	Model_Name	Variant	Year	Fuel_Type	Transmission	Engine_CC	Powe
56374	56375	M4	Sport Line	2015	Petrol	Semi-Automatic	1998	
70354	70355	X3	Standard	2017	Electric	Semi-Automatic	4395	
28962	28963	i8	xDrive	2019	Hybrid	Automatic	4395	
51241	51242	M4	Luxury Line	2023	Hybrid	Semi-Automatic	4395	
14031	14032	M3	M Sport	2018	Petrol	Automatic	1998	
...	...	...	...	...	...	...	...	
57442	57443	M3	Standard	2013	Hybrid	Semi-Automatic	1496	
62468	62469	520d	M Sport	2016	Hybrid	Semi-Automatic	1496	
97582	97583	X7	M Sport	2012	Petrol	Manual	1496	
94872	94873	320d	Luxury Line	2018	Petrol	Manual	4395	
71910	71911	X1	Luxury Line	2016	Diesel	Semi-Automatic	1998	

75155 rows × 16 columns

12.Data Analysis

In [43]:

```
df['Location'].nunique()
```

Out[43]:

```
8
```

In [44]:

```
df['Location'].value_counts().sort_values(ascending = False).head(5)
```

Out[44]:

```
Location
Ahmedabad    12668
Chennai      12549
Mumbai       12495
Bangalore    12492
Delhi        12478
Name: count, dtype: int64
```

13. Histogram

In [45]:

```
import numpy as np
```

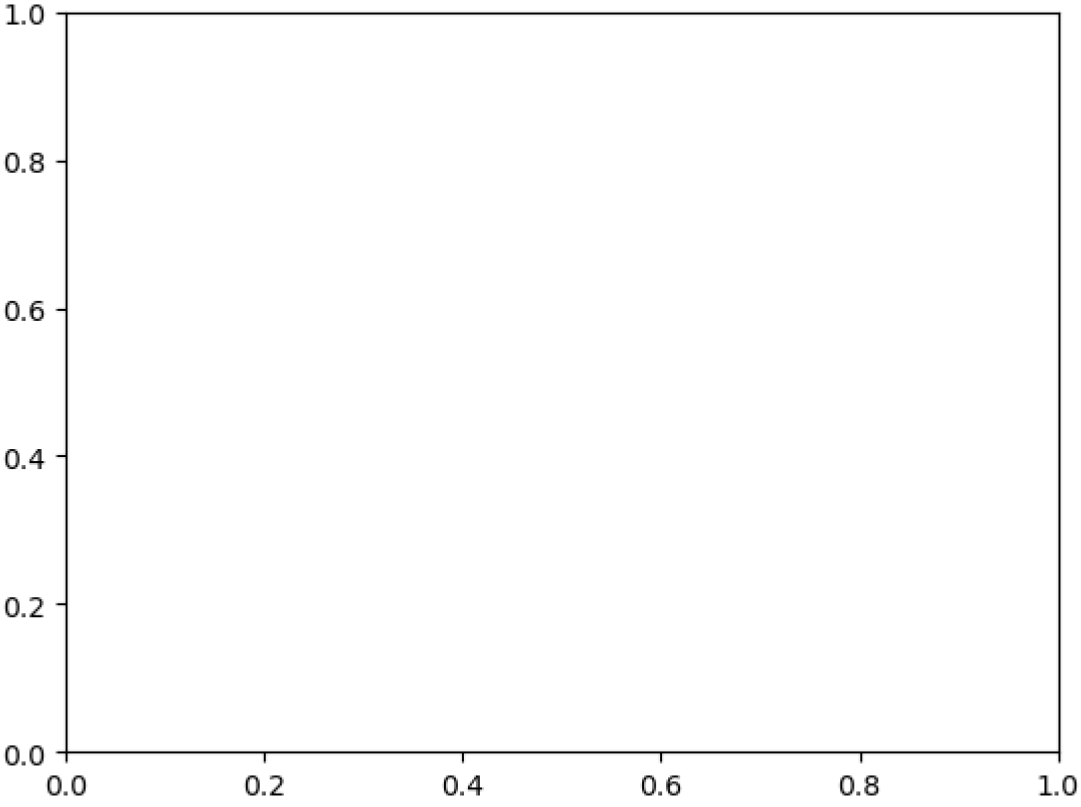
In [46]:

```
df.plot(
    y='Mileage_kmpl',
    xline=(0,100),
    kind='kde'
);
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[46], line 1
----> 1 df.plot(
      2     y='Mileage_kmpl',
      3     xline=(0,100),
      4     kind='kde'
      5 )

File ~\Tanishka\Lib\site-packages\pandas\plotting\_core.py:1030, in PlotAccessor.
__call__(self, *args, **kwargs)
   1027             label_name = label_kw or data.columns
   1028             data.columns = label_name
-> 1030 return plot_backend.plot(data, kind=kind, **kwargs)

File ~\Tanishka\Lib\site-packages\pandas\plotting\_matplotlib\__init__.py:71, in
plot(data, kind, **kwargs)
     69         kwargs["ax"] = getattr(ax, "left_ax", ax)
     70 plot_obj = PLOT_CLASSES[kind](data, **kwargs)
---> 71 plot_obj.generate()
     72 plot_obj.draw()
     73 return plot_obj.result

File ~\Tanishka\Lib\site-packages\pandas\plotting\_matplotlib\core.py:501, in MPL
Plot.generate(self)
    499 self._compute_plot_data()
    500 fig = self.fig
--> 501 self._make_plot(fig)
    502 self._add_table()
    503 self._make_legend()

File ~\Tanishka\Lib\site-packages\pandas\plotting\_matplotlib\hist.py:168, in His
tPlot._make_plot(self, fig)
    164     kwds["weights"] = type(self)._get_column_weights(self.weights, i, y)
    166 y = reformat_hist_y_given_by(y, self.by)
--> 168 artists = self._plot(ax, y, column_num=i, stacking_id=stacking_id, **kwd
s)
    170 # when by is applied, show title for subplots to know which group it is
    171 if self.by is not None:

File ~\Tanishka\Lib\site-packages\pandas\plotting\_matplotlib\hist.py:282, in Kde
Plot._plot(cls, ax, y, style, bw_method, ind, column_num, stacking_id, **kwds)
    279 gkde = gaussian_kde(y, bw_method=bw_method)
    281 y = gkde.evaluate(ind)
--> 282 lines = MPLPlot._plot(ax, ind, y, style=style, **kwds)
    283 return lines

File ~\Tanishka\Lib\site-packages\pandas\plotting\_matplotlib\converter.py:95, in
register_pandas_matplotlib_converters.<locals>.wrapper(*args, **kwargs)
     92 @functools.wraps(func)
     93 def wrapper(*args, **kwargs):
     94     with pandas_converters():
---> 95         return func(*args, **kwargs)

File ~\Tanishka\Lib\site-packages\pandas\plotting\_matplotlib\core.py:981, in MPL
Plot._plot(cls, ax, x, y, style, is_errorbar, **kwds)
    978 else:
    979     # prevent style kwarg from going to errorbar, where it is unsupported
    980     args = (x, y, style) if style is not None else (x, y)
--> 981     return ax.plot(*args, **kwds)
```

```
File ~\Tanishka\Lib\site-packages\matplotlib\axes\_axes.py:1777, in Axes.plot(sel
f, scalex, scaley, data, *args, **kwargs)
   1534 """
   1535 Plot y versus x as lines and/or markers.
   1536
   (...)
   1774 (``'green'``) or hex strings (``'#008000'``).
   1775 """
   1776 kwargs = cbook.normalize_kwargs(kwargs, mlines.Line2D)
-> 1777 lines = [*self._get_lines(self, *args, data=data, **kwargs)]
   1778 for line in lines:
   1779     self.add_line(line)

File ~\Tanishka\Lib\site-packages\matplotlib\axes\_base.py:297, in _process_plot_
var_args.__call__(self, axes, data, return_kwargs, *args, **kwargs)
    295     this += args[0],
    296     args = args[1:]
--> 297 yield from self._plot_args(
    298     axes, this, kwargs, ambiguous_fmt_datakey=ambiguous_fmt_datakey,
    299     return_kwargs=return_kwargs
    300 )

File ~\Tanishka\Lib\site-packages\matplotlib\axes\_base.py:546, in _process_plot_
var_args._plot_args(self, axes, tup, kwargs, return_kwargs, ambiguous_fmt_datake
y)
    544     return list(result)
    545 else:
--> 546     return [l[0] for l in result]

File ~\Tanishka\Lib\site-packages\matplotlib\axes\_base.py:539, in <genexpr>(.0)
    534 else:
    535     raise ValueError(
    536         f"label must be scalar or have the same length as the input "
    537         f"data, but found {len(label)} for {n_datasets} datasets.")
--> 539 result = (make_artist(axes, x[:, j % ncx], y[:, j % ncy], kw,
    540                       {**kwargs, 'label': label})
    541           for j, label in enumerate(labels))
    543 if return_kwargs:
    544     return list(result)

File ~\Tanishka\Lib\site-packages\matplotlib\axes\_base.py:338, in _process_plot_
var_args._make_line(self, axes, x, y, kw, kwargs)
    336 kw = {**kw, **kwargs}  # Don't modify the original kw.
    337 self._setdefaults(self._getdefaults(kw), kw)
--> 338 seg = mlines.Line2D(x, y, **kw)
    339 return seg, kw

File ~\Tanishka\Lib\site-packages\matplotlib\lines.py:407, in Line2D.__init__(sel
f, xdata, ydata, linewidth, linestyle, color, gapcolor, marker, markersize, marke
redgewidth, markeredgecolor, markerfacecolor, markerfacecoloralt, fillstyle, anti
aliased, dash_capstyle, solid_capstyle, dash_joinstyle, solid_joinstyle, pickradi
us, drawstyle, markevery, **kwargs)
    403 self.set_markeredgewidth(markeredgewidth)
    405 # update kwargs before updating data to give the caller a
    406 # chance to init axes (and hence unit support)
--> 407 self._internal_update(kwargs)
    408 self.pickradius = pickradius
    409 self.ind_offset = 0
```

```
File ~\Tanishka\Lib\site-packages\matplotlib\artist.py:1233, in Artist._internal_
update(self, kwargs)
   1226 def _internal_update(self, kwargs):
   1227     """
   1228     Update artist properties without prenormalizing them, but generating
   1229     errors as if calling `set`.
   1230 
   1231     The lack of prenormalization is to maintain backcompatibility.
   1232     """
-> 1233     return self._update_props(
   1234         kwargs, "{cls.__name__}.set() got an unexpected keyword argument
"
   1235         "{prop_name!r}")

File ~\Tanishka\Lib\site-packages\matplotlib\artist.py:1206, in Artist._update_pr
ops(self, props, errfmt)
   1204             func = getattr(self, f"set_{k}", None)
   1205             if not callable(func):
-> 1206                 raise AttributeError(
   1207                     errfmt.format(cls=type(self), prop_name=k),
   1208                     name=k)
   1209             ret.append(func(v))
   1210 if ret:

AttributeError: Line2D.set() got an unexpected keyword argument 'xline'
```



Bar chart

In [ ]:

```
import pandas as pd
```

In [ ]:

```
Fuel_Type = df.groupby('Fuel_Type')['Power_BHP'].mean()
```

```
In [ ]: Fuel_Type.plot(kind='bar')
```

```
In [ ]: df.loc[df['Fuel_Type'] == 'Fuel_Type'] = 'Electric'
        df
```

Convert

```
In [ ]: df['Price_Ex_Showroom'] = df['Price_Ex_Showroom'] * 100000
        df
```

Find how many cars are older than 10 years

```
In [ ]: import pandas as pd
```

```
In [ ]: df['Car_Age'] = 2025 - df['Year']
        df
```

```
In [ ]: older_Cars = df[df['Car_Age']>10]
```

```
In [ ]: count_older_Cars = older_cars.shape[0]
```

```
In [ ]: print("Number of cars older than 10 years:",count_older_Cars)
        df
```

```
In [ ]: count_by_Owner_Type = df.groupby(['Transmission','Owner_Type']).size().reset_ind
        count_by_Owner_Type
```

Find maximum BHP in diesel cars

```
In [ ]: max_power_bhp_diesel = df[df['Fuel_Type'] == 'Diesel']['Power_BHP'].max()
```

```
In [ ]: max_power_bhp_diesel
```

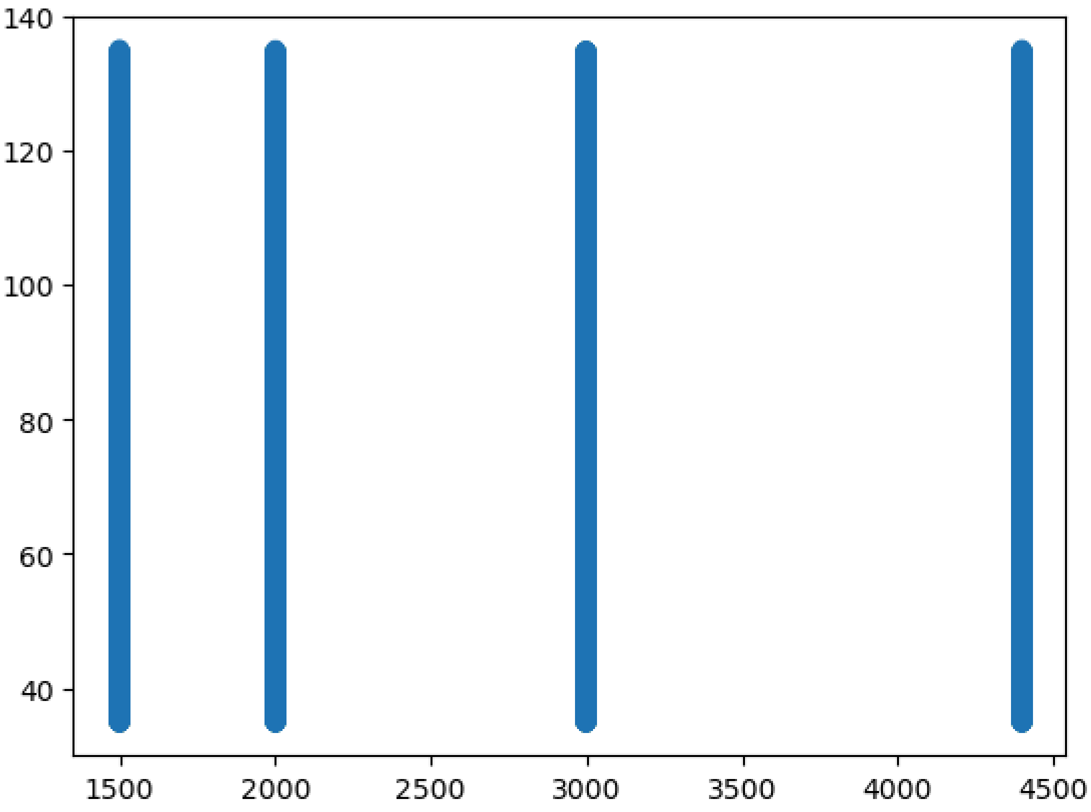
Plot scatter plot of Engine_CC vs Price.

```
In [47]: import matplotlib.pyplot as plt

         # Scatter plot
         plt.figure(figsize=(8,5))
```

```
Out[47]: <Figure size 800x500 with 0 Axes>
         <Figure size 800x500 with 0 Axes>
```

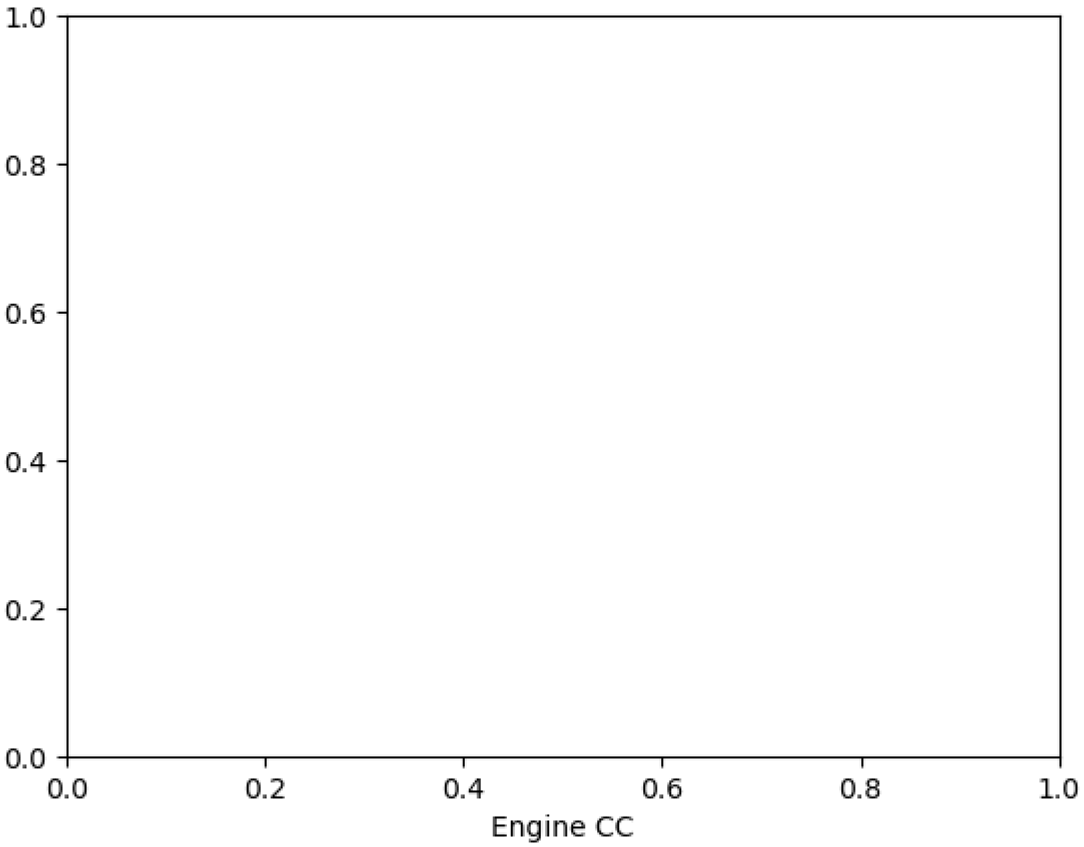
```
In [48]: plt.scatter(df['Engine_CC'], df['Price_Ex_Showroom'], alpha=0.6)
```

```
Out[48]: <matplotlib.collections.PathCollection at 0x2345602d010>
```



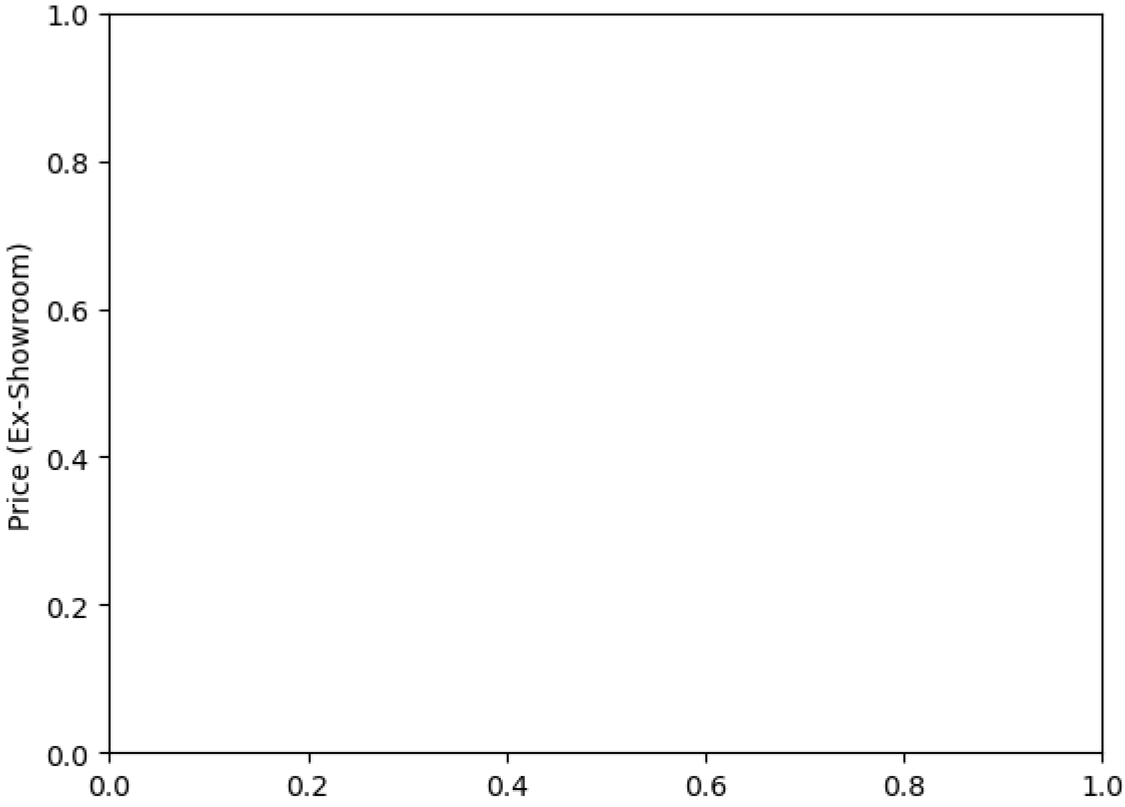
```
In [49]: plt.xlabel('Engine CC')
```

```
Out[49]: Text(0.5, 0, 'Engine CC')
```



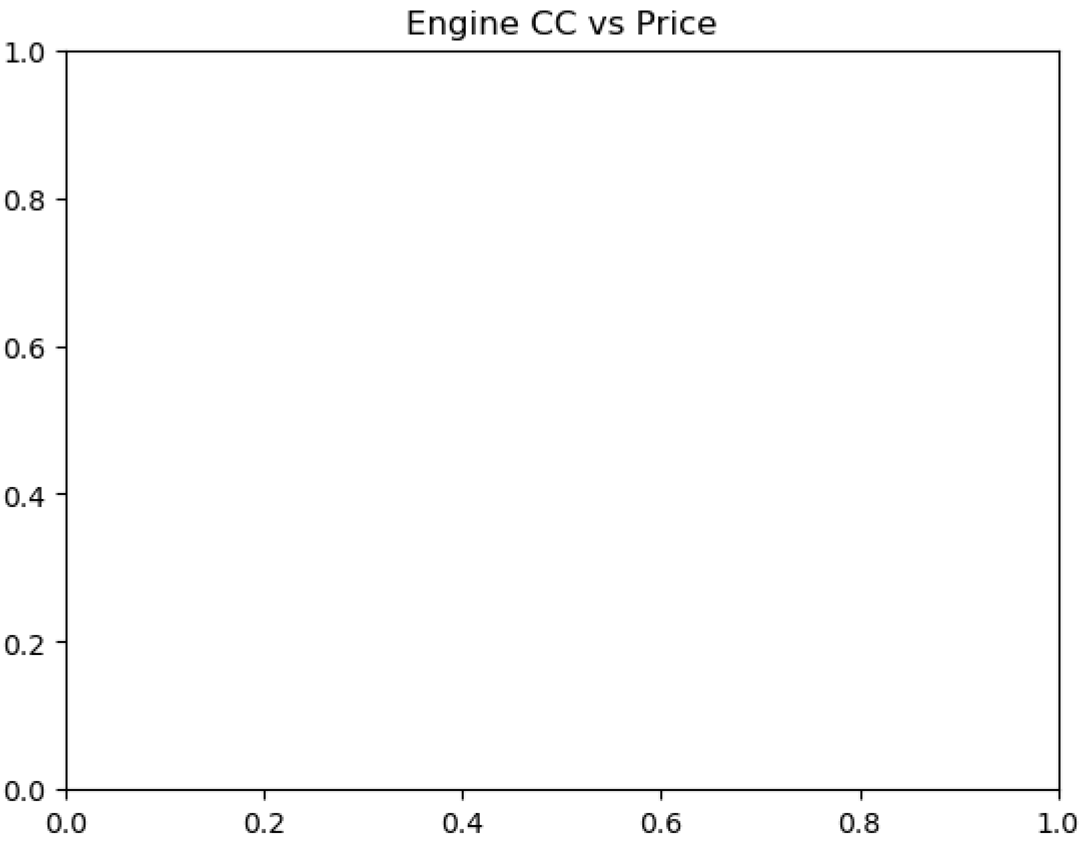
```
In [50]: plt.ylabel('Price (Ex-Showroom)')
```

```
Out[50]: Text(0, 0.5, 'Price (Ex-Showroom)')
```

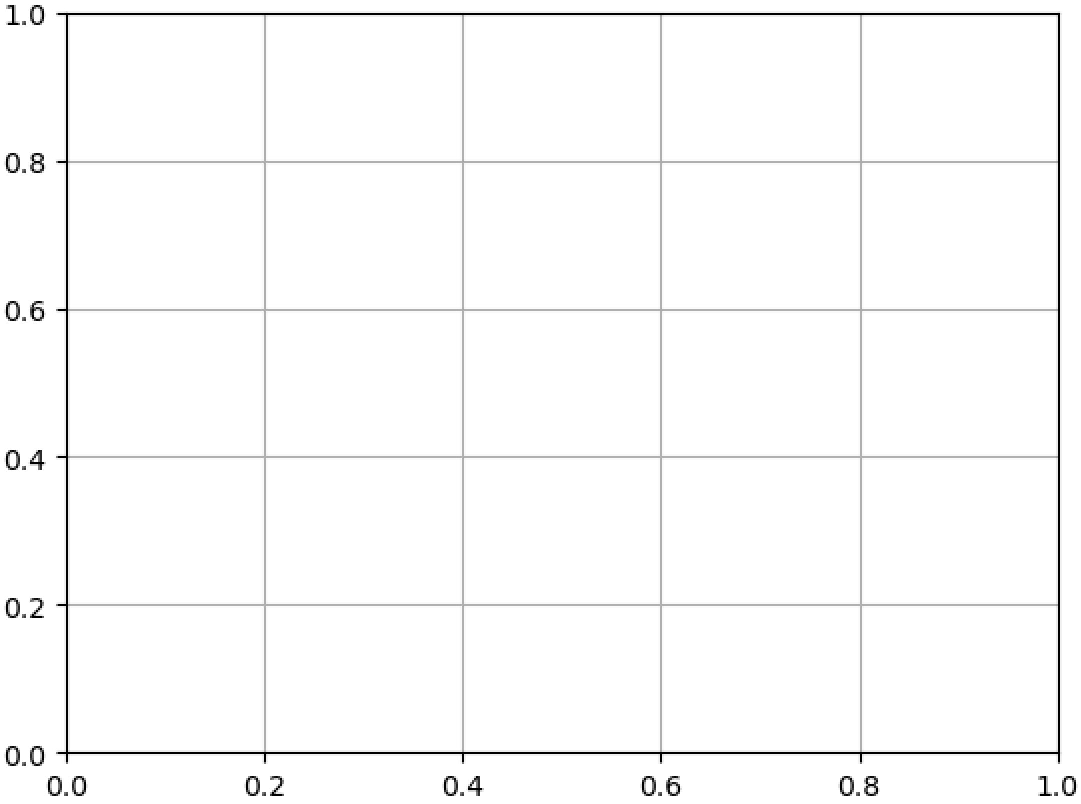


```
In [51]: plt.title('Engine CC vs Price')
```

```
Out[51]: Text(0.5, 1.0, 'Engine CC vs Price')
```



```
In [52]: plt.grid(True)
```



14. Count number of variants per model.

In [54]:
```
variants_per_model = df.groupby('Model_Name')['Variant'].nunique().reset_index(n
```

In [55]:
```
variants_per_model
```

Out[55]:

	Model_Name	Variant_Count
0	320d	5
1	520d	5
2	M3	5
3	M4	5
4	X1	5
5	X3	5
6	X5	5
7	X7	5
8	Z4	5
9	i8	5

15. Find car with max and min mileage.

In [56]:
```
max_mileage_kmpl_car = df.loc[df['Mileage_kmpl'].idxmax()]
```

In [57]:
```
max_mileage_kmpl_car
```

Out[57]:
```
ID                         7542
Model_Name                   M3
Variant                  xDrive
Year                       2013
Fuel_Type                Hybrid
Transmission             Manual
Engine_CC                  1496
Power_BHP                474.04
Torque_Nm                739.46
Mileage_kmpl               25.0
Price_Ex_Showroom        131.67
Owner_Type                First
Insurance_Valid_Till       2023
Location                Kolkata
Registration_State           TN
Car_Age                      12
Name: 7541, dtype: object
```

In [58]:
```
min_mileage_kmpl_car = df.loc[df['Mileage_kmpl'].idxmin()]
min_mileage_kmpl_car
```

Out[58]:
```
ID                                4144
Model_Name                        320d
Variant                        M Sport
Year                              2014
Fuel_Type                       Petrol
Transmission                    Manual
Engine_CC                         1998
Power_BHP                        197.3
Torque_Nm                       735.21
Mileage_kmpl                       8.0
Price_Ex_Showroom               101.52
Owner_Type              Fourth & Above
Insurance_Valid_Till              2023
Location                       Kolkata
Registration_State                  WB
Car_Age                             11
Name: 4143, dtype: object
```

Calculate median price per year.

In [59]:
```
median_price_per_year = df.groupby('Year')['Price_Ex_Showroom'].median().reset_i
```

In [60]:
```
median_price_per_year
```

Out[60]:

	Year	Median_Price
0	2010	85.990
1	2011	85.585
2	2012	84.855
3	2013	84.930
4	2014	84.840
5	2015	85.570
6	2016	85.460
7	2017	85.020
8	2018	85.090
9	2019	84.850
10	2020	85.150
11	2021	85.890
12	2022	85.265
13	2023	84.850
14	2024	85.240

16. Pivot table with Location vs Fuel_Type.

In [61]:

```
pivot_table = pd.pivot_table(df,
                             index='Location',
                             columns='Fuel_Type',
                             values='Model_Name',  # can be any column, we use c
                             aggfunc='count',
                             fill_value=0)

pivot_table
```

Out[61]:

Fuel_Type	Diesel	Electric	Hybrid	Petrol
Location				
Ahmedabad	3210	3220	3095	3143
Bangalore	3074	3155	3183	3080
Chennai	3090	3171	3125	3163
Delhi	3201	3014	3078	3185
Hyderabad	3149	3069	3054	3168
Kolkata	3137	3093	3050	3135
Mumbai	3139	3080	3165	3111
Pune	3089	3050	3184	3140

17.Create a column Is_Luxury (if Price > ₹70L = Yes).

```
In [63]: df['Is_Luxury'] = df['Price_Ex_Showroom'].apply(lambda x: 'Yes' if x > 7000000 e
```

```
In [65]: df['Is_Luxury']
```

```
Out[65]: 0        No
         1        No
         2        No
         3        No
         4        No
                  ..
         99995    No
         99996    No
         99997    No
         99998    No
         99999    No
         Name: Is_Luxury, Length: 100000, dtype: object
```

18.Remove duplicate rows if any.

```
In [66]: df = df.drop_duplicates()
```

```
In [67]: df=df.reset_index
```

```
In [68]: df
```

Out[68]:

```
<bound method DataFrame.reset_index of            ID Model_Name      Variant  Y
ear Fuel_Type    Transmission  \
0           1         X5       xDrive  2019  Electric          Manual
1           2         X5  Luxury Line  2016  Electric       Automatic
2           3         Z4      M Sport  2012  Electric  Semi-Automatic
3           4         X7      M Sport  2013    Hybrid  Semi-Automatic
4           5         X7       xDrive  2010  Electric          Manual
...       ...        ...          ...   ...       ...             ...
99995   99996         Z4   Sport Line  2014    Hybrid  Semi-Automatic
99996   99997         Z4   Sport Line  2021  Electric       Automatic
99997   99998         M4     Standard  2012    Hybrid       Automatic
99998   99999         X7  Luxury Line  2015  Electric          Manual
99999  100000       520d  Luxury Line  2020  Electric  Semi-Automatic

       Engine_CC  Power_BHP  Torque_Nm  Mileage_kmpl  Price_Ex_Showroom  \
0           1496     395.18     492.04         20.48              68.33
1           1496     312.78     708.05          8.12             118.58
2           4395     604.45     550.74         16.24             109.90
3           4395     383.04     413.78          9.80              41.05
4           2993     315.78     632.13         23.47             111.08
...          ...        ...        ...           ...                ...
99995       1998     208.30     385.44          9.83              99.15
99996       1496     562.64     730.43         13.45              70.56
99997       2993     140.86     386.50         14.44              44.46
99998       4395     434.00     548.41         14.93              84.82
99999       1496     392.05     307.86         13.97              77.14

           Owner_Type  Insurance_Valid_Till   Location Registration_State  \
0              Second                  2022  Hyderabad                 KA
1              Second                  2020    Chennai                 DL
2              Second                  2022  Hyderabad                 DL
3      Fourth & Above                  2023  Hyderabad                 DL
4               First                  2024    Chennai                 TS
...               ...                   ...        ...                ...
99995           First                  2022    Chennai                 TN
99996           Third                  2021  Bangalore                 TS
99997  Fourth & Above                  2024    Kolkata                 DL
99998          Second                  2022    Chennai                 TS
99999          Second                  2022       Pune                 DL

       Car_Age Is_Luxury
0            6        No
1            9        No
2           13        No
3           12        No
4           15        No
...        ...       ...
99995       11        No
99996        4        No
99997       13        No
99998       10        No
99999        5        No

[100000 rows x 17 columns]>
```

19. Apply filter: Diesel + Automatic + Above 200 BHP

In [69]:

```
df
```

Out[69]:

```
<bound method DataFrame.reset_index of            ID Model_Name      Variant  Y
ear Fuel_Type    Transmission  \
0           1         X5       xDrive  2019  Electric          Manual
1           2         X5  Luxury Line  2016  Electric       Automatic
2           3         Z4      M Sport  2012  Electric  Semi-Automatic
3           4         X7      M Sport  2013    Hybrid  Semi-Automatic
4           5         X7       xDrive  2010  Electric          Manual
...       ...        ...          ...   ...       ...             ...
99995   99996         Z4   Sport Line  2014    Hybrid  Semi-Automatic
99996   99997         Z4   Sport Line  2021  Electric       Automatic
99997   99998         M4     Standard  2012    Hybrid       Automatic
99998   99999         X7  Luxury Line  2015  Electric          Manual
99999  100000       520d  Luxury Line  2020  Electric  Semi-Automatic

       Engine_CC  Power_BHP  Torque_Nm  Mileage_kmpl  Price_Ex_Showroom  \
0           1496     395.18     492.04         20.48              68.33
1           1496     312.78     708.05          8.12             118.58
2           4395     604.45     550.74         16.24             109.90
3           4395     383.04     413.78          9.80              41.05
4           2993     315.78     632.13         23.47             111.08
...          ...        ...        ...           ...                ...
99995       1998     208.30     385.44          9.83              99.15
99996       1496     562.64     730.43         13.45              70.56
99997       2993     140.86     386.50         14.44              44.46
99998       4395     434.00     548.41         14.93              84.82
99999       1496     392.05     307.86         13.97              77.14

           Owner_Type  Insurance_Valid_Till   Location Registration_State  \
0              Second                  2022  Hyderabad                 KA
1              Second                  2020    Chennai                 DL
2              Second                  2022  Hyderabad                 DL
3      Fourth & Above                  2023  Hyderabad                 DL
4               First                  2024    Chennai                 TS
...               ...                   ...        ...                ...
99995           First                  2022    Chennai                 TN
99996           Third                  2021  Bangalore                 TS
99997  Fourth & Above                  2024    Kolkata                 DL
99998          Second                  2022    Chennai                 TS
99999          Second                  2022       Pune                 DL

       Car_Age Is_Luxury
0            6        No
1            9        No
2           13        No
3           12        No
4           15        No
...        ...       ...
99995       11        No
99996        4        No
99997       13        No
99998       10        No
99999        5        No

[100000 rows x 17 columns]>
```

In [74]:

```
import pandas as pd
```

In [75]:

```
df = pd.read_csv('C:\\Users\\Sujit\\OneDrive\\Desktop\\pandas2\\Bmw car\\bmw_car
```

In [76]:

```
filtered_cars = df[
    (df['Fuel_Type'] == 'Diesel') &
    (df['Transmission'] == 'Automatic') &
    (df['Power_BHP'] > 200)
]

print(filtered_cars)
```

```
          ID Model_Name       Variant  Year Fuel_Type Transmission  Engine_CC  \
45        46         X3       xDrive  2016    Diesel    Automatic       1998
77        78         Z4  Luxury Line  2019    Diesel    Automatic       1998
118      119       520d      M Sport  2011    Diesel    Automatic       1998
127      128       520d      M Sport  2016    Diesel    Automatic       1496
138      139         Z4  Luxury Line  2024    Diesel    Automatic       4395
...      ...        ...          ...   ...       ...          ...        ...
99903  99904         Z4       xDrive  2012    Diesel    Automatic       4395
99919  99920       320d   Sport Line  2021    Diesel    Automatic       4395
99926  99927       520d     Standard  2021    Diesel    Automatic       1998
99934  99935         M3  Luxury Line  2018    Diesel    Automatic       4395
99947  99948         X7   Sport Line  2012    Diesel    Automatic       1998

       Power_BHP  Torque_Nm  Mileage_kmpl  Price_Ex_Showroom      Owner_Type  \
45        487.85     309.18         18.18              85.91  Fourth & Above
77        257.02     435.53         24.27              64.34          Second
118       349.39     441.50         14.33              75.27           Third
127       574.09     364.75         22.80             131.70           First
138       442.82     550.26         16.37              60.09           Third
...          ...        ...           ...                ...             ...
99903     450.75     335.93         17.63             112.61          Second
99919     269.10     565.04         19.72              57.51          Second
99926     426.14     377.34         21.33              99.41  Fourth & Above
99934     536.44     313.94         14.60              85.84           Third
99947     287.72     688.12         15.90              66.47          Second

       Insurance_Valid_Till   Location Registration_State
45                     2025      Delhi                 TN
77                     2022  Ahmedabad                 DL
118                    2024  Bangalore                 TS
127                    2024      Delhi                 TN
138                    2020       Pune                 MH
...                     ...        ...                ...
99903                  2021      Delhi                 GJ
99919                  2025    Chennai                 DL
99926                  2025  Ahmedabad                 GJ
99934                  2021      Delhi                 GJ
99947                  2025    Kolkata                 DL

[7473 rows x 15 columns]
```

In [ ]: