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

In [3]:
```
df=pd.read_excel(r"C:\Users\bittu\Desktop\Mini Project files\Book1.xlsx")
```

In [5]:
```
df
```

Out[5]:

	index	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	manufacturing_year	mileage(
0	0	Mercedes-Benz	Comprehensive	Petrol	5	56000	First Owner	Automatic	2017	
1	1	Nissan	Comprehensive	Petrol	5	30615	First Owner	Automatic	2020	
2	2	BMW	Comprehensive	Diesel	5	24000	First Owner	Automatic	2018	
3	3	Kia	Comprehensive	Petrol	5	18378	First Owner	Manual	2019	
4	4	Skoda	Comprehensive	Petrol	5	44900	First Owner	Automatic	2019	
...	...	...	...	...	...	...	...	...	...	
1548	1548	Hyundai	Comprehensive	Diesel	5	35000	First Owner	Automatic	2020	14
1549	1549	Renault	Third Party insurance	Petrol	5	10000	999 cc	2022	Power Steering	9
1550	1550	Honda	Comprehensive	Petrol	5	49000	First Owner	Manual	2017	
1551	1551	Volkswagen	Comprehensive	Petrol	5	40000	Second Owner	Manual	2018	
1552	1552	Maruti	Comprehensive	Petrol	5	34756	First Owner	Manual	2018	

1553 rows × 14 columns

In [7]:
```
df.drop(columns='index',inplace=True)
```

In [9]:
```
df.index=np.arange(1,len(df)+1)
```

In [11]:
```
df
```

Out[11]:

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	manufacturing_year	mileage(kmpl)
1	Mercedes-Benz	Comprehensive	Petrol	5	56000	First Owner	Automatic	2017	7.81
2	Nissan	Comprehensive	Petrol	5	30615	First Owner	Automatic	2020	17.40
3	BMW	Comprehensive	Diesel	5	24000	First Owner	Automatic	2018	20.68
4	Kia	Comprehensive	Petrol	5	18378	First Owner	Manual	2019	16.50
5	Skoda	Comprehensive	Petrol	5	44900	First Owner	Automatic	2019	14.67
...	...	...	...	...	...	...	...	...	...
1549	Hyundai	Comprehensive	Diesel	5	35000	First Owner	Automatic	2020	1493.00
1550	Renault	Third Party insurance	Petrol	5	10000	999 cc	2022	Power Steering	999.00
1551	Honda	Comprehensive	Petrol	5	49000	First Owner	Manual	2017	17.50
1552	Volkswagen	Comprehensive	Petrol	5	40000	Second Owner	Manual	2018	18.78
1553	Maruti	Comprehensive	Petrol	5	34756	First Owner	Manual	2018	20.85

1553 rows × 13 columns

In [13]:
```
df.duplicated().sum()
```

Out[13]:
```
421
```

In [15]:
```
df.drop_duplicates(inplace=True)
```

In [17]:
```
df.shape
```

Out[17]:
```
(1132, 13)
```

In [19]:
```
df.dropna(inplace=True)
```

In [21]:
```
df.isna().sum()
```

Out[21]:
```
car_name              0
insurance_validity    0
fuel_type             0
seats                 0
kms_driven            0
ownsership            0
transmission          0
manufacturing_year    0
mileage(kmpl)         0
engine(cc)            0
max_power(bhp)        0
torque(Nm)            0
price(in lakhs)       0
dtype: int64
```

In [23]:
```
df['car_name']=df['car_name'].replace('Lexus','Nexus')
```

In [25]:
```
df['car_name'].value_counts()
```

Out[25]:
```
car_name
Maruti           254
Hyundai          240
Honda            130
Mercedes-Benz     97
BMW               60
Toyota            46
Audi              40
Tata              38
Mahindra          33
Ford              25
Volkswagen        22
Kia               22
Renault           21
Nissan            17
Land              14
MG                14
Skoda             11
Jeep              11
Volvo             10
Jaguar             6
Datsun             4
Mitsubishi         3
Nexus              3
Isuzu              2
Porsche            2
Mini               2
Fiat               1
Lamborghini        1
Name: count, dtype: int64
```

In [27]:
```
df['insurance_validity']=df['insurance_validity'].replace('Third Party','Third Party insurance')
df['insurance_validity']=df['insurance_validity'].replace('Zero Dep','Third Party insurance')
df['insurance_validity']=df['insurance_validity'].replace('Not Available','')
df['insurance_validity']=df['insurance_validity'].replace('Petrol','')
a=['']
b=df[df['insurance_validity'].isin(a)].index
df.drop(b,inplace=True)
```

In [29]:
```
df['insurance_validity']=df['insurance_validity'].replace('Comprehensive','Yes')
df['insurance_validity']=df['insurance_validity'].replace('Third Party insurance','No')
```

In [31]:
```
dfle=df
```

In [33]:
```
df=df.sort_values(by='price(in lakhs)')
```

In [35]:
```
df
```

Out[35]:

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	manufacturing_year	mileage(kmpl)	
1109	Fiat	No	Diesel	5	80000	Third Owner	Manual	2010	20.30	1.
1300	Porsche	No	Petrol	5	5700	First Owner	Automatic	2019	2995.00	3.
917	Mercedes-Benz	No	Diesel	7	46001	First Owner	Automatic	2021	2925.00	3.
464	Mercedes-Benz	Yes	Petrol	4	40000	First Owner	Automatic	2018	7.81	4.
1464	Mercedes-Benz	Yes	Diesel	7	34987	First Owner	Automatic	2020	2925.00	3.
...	...	...	...	...	...	...	...	...	...	
480	BMW	Yes	Diesel	7	44000	First Owner	Automatic	2020	13.38	2.
221	Mercedes-Benz	Yes	Petrol	4	50000	First Owner	Automatic	2016	7.81	4.
1493	BMW	Yes	Petrol	7	28000	First Owner	Automatic	2019	10.54	2.
174	Ford	Yes	Petrol	5	66717	Second Owner	Manual	2011	1196.00	7.
1278	Maruti	Yes	Petrol	5	44002	Third Owner	Manual	2009	19.70	7.

1127 rows × 13 columns

In [37]:
```
dfle
```

Out[37]:

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	manufacturing_year	mileage(kmpl)
1	Mercedes-Benz	Yes	Petrol	5	56000	First Owner	Automatic	2017	7.81
2	Nissan	Yes	Petrol	5	30615	First Owner	Automatic	2020	17.40
3	BMW	Yes	Diesel	5	24000	First Owner	Automatic	2018	20.68
4	Kia	Yes	Petrol	5	18378	First Owner	Manual	2019	16.50
5	Skoda	Yes	Petrol	5	44900	First Owner	Automatic	2019	14.67
...	...	...	...	...	...	...	...	...	...
1549	Hyundai	Yes	Diesel	5	35000	First Owner	Automatic	2020	1493.00
1550	Renault	No	Petrol	5	10000	999 cc	2022	Power Steering	999.00
1551	Honda	Yes	Petrol	5	49000	First Owner	Manual	2017	17.50
1552	Volkswagen	Yes	Petrol	5	40000	Second Owner	Manual	2018	18.78
1553	Maruti	Yes	Petrol	5	34756	First Owner	Manual	2018	20.85

1127 rows × 13 columns

In [39]:
```
pd.set_option('display.float_format', '{:.2f}'.format)
```

In [41]:
```
df
```

Out[41]:

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	manufacturing_year	mileage(kmpl)
1109	Fiat	No	Diesel	5	80000	Third Owner	Manual	2010	20.30
1300	Porsche	No	Petrol	5	5700	First Owner	Automatic	2019	2995.00
917	Mercedes-Benz	No	Diesel	7	46001	First Owner	Automatic	2021	2925.00
464	Mercedes-Benz	Yes	Petrol	4	40000	First Owner	Automatic	2018	7.81
1464	Mercedes-Benz	Yes	Diesel	7	34987	First Owner	Automatic	2020	2925.00
...	...	...	...	...	...	...	...	...	...
480	BMW	Yes	Diesel	7	44000	First Owner	Automatic	2020	13.38
221	Mercedes-Benz	Yes	Petrol	4	50000	First Owner	Automatic	2016	7.81
1493	BMW	Yes	Petrol	7	28000	First Owner	Automatic	2019	10.54
174	Ford	Yes	Petrol	5	66717	Second Owner	Manual	2011	1196.00
1278	Maruti	Yes	Petrol	5	44002	Third Owner	Manual	2009	19.70

1127 rows × 13 columns

In [43]:
```
df.sort_values(by='engine(cc)')
```

Out[43]:

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	manufacturing_year	mileage(kmpl)
1494	Maruti	No	Petrol	5	79862	First Owner	Manual	2010	1061.00
1275	Hyundai	No	Diesel	5	67000	First Owner	Manual	2015	1120.00
174	Ford	Yes	Petrol	5	66717	Second Owner	Manual	2011	1196.00
1434	Volkswagen	Yes	Petrol	5	62920	First Owner	Manual	2017	1198.00
1021	Mahindra	Yes	Petrol	6	18706	First Owner	Manual	2018	1198.00
...	...	...	...	...	...	...	...	...	...
920	Mercedes-Benz	Yes	Diesel	5	76000	First Owner	Automatic	2015	2987.00
917	Mercedes-Benz	No	Diesel	7	46001	First Owner	Automatic	2021	2925.00
468	Mercedes-Benz	Yes	Diesel	7	19000	Second Owner	Automatic	2022	2925.00
706	Mercedes-Benz	No	Diesel	7	16000	First Owner	Automatic	2023	2925.00
1464	Mercedes-Benz	Yes	Diesel	7	34987	First Owner	Automatic	2020	2925.00

1127 rows × 13 columns

In [45]:
```
remove_extra=df[df['engine(cc)']>8000].index
```

In [47]:
```
df.drop(remove_extra,inplace=True)
```

In [49]:
```
df.describe()
```

Out[49]:

	seats	kms_driven	mileage(kmpl)	engine(cc)	max_power(bhp)	torque(Nm)	price(in lakhs)
count	1072.00	1072.00	1072.00	1072.00	1072.00	1072.00	1072.00
mean	5.18	53857.13	135.94	1627.16	1627.16	13804.22	167.17
std	0.60	43603.46	390.48	1004.20	1004.20	87418.37	3601.99
min	4.00	1000.00	7.81	67.00	67.00	17.00	1.00
25%	5.00	30375.00	16.50	1197.00	1197.00	739.00	4.52
50%	5.00	49796.50	18.90	1384.00	1384.00	1213.00	6.75
75%	5.00	70000.00	21.40	1956.00	1956.00	8873.00	13.90
max	8.00	810000.00	3996.00	7394.00	7394.00	1186600.00	95000.00

In [51]:
```
df['price(in lakhs)']=df['price(in lakhs)'].replace(95000,0.95)
df['price(in lakhs)']=df['price(in lakhs)'].replace(70000,0.70)
```

In [53]:
```
df
```

Out[53]:

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownership	transmission	manufacturing_year	mileage(kmpl)	e
1109	Fiat	No	Diesel	5	80000	Third Owner	Manual	2010	20.30	
1300	Porsche	No	Petrol	5	5700	First Owner	Automatic	2019	2995.00	
464	Mercedes-Benz	Yes	Petrol	4	40000	First Owner	Automatic	2018	7.81	
412	Land	No	Petrol	5	4000	First Owner	Automatic	2019	12.65	
634	Audi	Yes	Petrol	5	7000	First Owner	Automatic	2023	9.80	
...	...	...	...	...	...	...	...	...	...	
480	BMW	Yes	Diesel	7	44000	First Owner	Automatic	2020	13.38	
221	Mercedes-Benz	Yes	Petrol	4	50000	First Owner	Automatic	2016	7.81	
1493	BMW	Yes	Petrol	7	28000	First Owner	Automatic	2019	10.54	
174	Ford	Yes	Petrol	5	66717	Second Owner	Manual	2011	1196.00	
1278	Maruti	Yes	Petrol	5	44002	Third Owner	Manual	2009	19.70	

1072 rows × 13 columns

In [55]:
```
df.drop(columns='max_power(bhp)',inplace=True)
```

In [57]:
```
df
```

Out[57]:

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownership	transmission	manufacturing_year	mileage(kmpl)	e
1109	Fiat	No	Diesel	5	80000	Third Owner	Manual	2010	20.30	
1300	Porsche	No	Petrol	5	5700	First Owner	Automatic	2019	2995.00	
464	Mercedes-Benz	Yes	Petrol	4	40000	First Owner	Automatic	2018	7.81	
412	Land	No	Petrol	5	4000	First Owner	Automatic	2019	12.65	
634	Audi	Yes	Petrol	5	7000	First Owner	Automatic	2023	9.80	
...	...	...	...	...	...	...	...	...	...	
480	BMW	Yes	Diesel	7	44000	First Owner	Automatic	2020	13.38	
221	Mercedes-Benz	Yes	Petrol	4	50000	First Owner	Automatic	2016	7.81	
1493	BMW	Yes	Petrol	7	28000	First Owner	Automatic	2019	10.54	
174	Ford	Yes	Petrol	5	66717	Second Owner	Manual	2011	1196.00	
1278	Maruti	Yes	Petrol	5	44002	Third Owner	Manual	2009	19.70	

1072 rows × 12 columns

In [59]:
```
df['manufacturing_year'].value_counts()
remove_cat=['Power Steering','Power Windows Front']
```

In [61]:
```
ac=df[df['manufacturing_year'].isin(remove_cat)].index
```

In [63]:
```
df.drop(ac,inplace=True)
```

In [65]:
```
df
```

Out[65]:

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	manufacturing_year	mileage(kmpl)	e
1109	Fiat	No	Diesel	5	80000	Third Owner	Manual	2010	20.30	
1300	Porsche	No	Petrol	5	5700	First Owner	Automatic	2019	2995.00	
464	Mercedes-Benz	Yes	Petrol	4	40000	First Owner	Automatic	2018	7.81	
412	Land	No	Petrol	5	4000	First Owner	Automatic	2019	12.65	
634	Audi	Yes	Petrol	5	7000	First Owner	Automatic	2023	9.80	
...	...	...	...	...	...	...	...	...	...	
480	BMW	Yes	Diesel	7	44000	First Owner	Automatic	2020	13.38	
221	Mercedes-Benz	Yes	Petrol	4	50000	First Owner	Automatic	2016	7.81	
1493	BMW	Yes	Petrol	7	28000	First Owner	Automatic	2019	10.54	
174	Ford	Yes	Petrol	5	66717	Second Owner	Manual	2011	1196.00	
1278	Maruti	Yes	Petrol	5	44002	Third Owner	Manual	2009	19.70	

1055 rows × 12 columns

In [67]:
```
df['current_year']=2024
```

In [69]:
```
df['car_age']=df['current_year']-df['manufacturing_year']
```

In [71]:
```
df.drop(columns=['current_year','manufacturing_year'],inplace=True)
```

In [73]:
```
df['ownsership'].value_counts()
```

Out[73]:
```
ownsership
First Owner     876
Second Owner    161
Third Owner      18
Name: count, dtype: int64
```

In [75]:
```
df
```

Out[75]:

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	mileage(kmpl)	engine(cc)	torque(Nm)
1109	Fiat	No	Diesel	5	80000	Third Owner	Manual	20.30	1248.00	75.00
1300	Porsche	No	Petrol	5	5700	First Owner	Automatic	2995.00	340.00	450.00
464	Mercedes-Benz	Yes	Petrol	4	40000	First Owner	Automatic	7.81	4663.00	459.00
412	Land	No	Petrol	5	4000	First Owner	Automatic	12.65	1997.00	29636.00
634	Audi	Yes	Petrol	5	7000	First Owner	Automatic	9.80	2995.00	340.00
...	...	...	...	...	...	...	...	...	...	...
480	BMW	Yes	Diesel	7	44000	First Owner	Automatic	13.38	2993.00	26150.00
221	Mercedes-Benz	Yes	Petrol	4	50000	First Owner	Automatic	7.81	4663.00	459.00
1493	BMW	Yes	Petrol	7	28000	First Owner	Automatic	10.54	2998.00	33525.00
174	Ford	Yes	Petrol	5	66717	Second Owner	Manual	1196.00	70.00	102.00
1278	Maruti	Yes	Petrol	5	44002	Third Owner	Manual	19.70	796.00	463.00

1055 rows × 12 columns

In [77]:
```
df.describe()
```

Out[77]:

	seats	kms_driven	mileage(kmpl)	engine(cc)	torque(Nm)	price(in lakhs)
count	1055.00	1055.00	1055.00	1055.00	1055.00	1055.00
mean	5.18	53968.92	114.03	1634.13	14022.70	13.26
std	0.61	43778.72	347.04	974.43	88103.44	16.42
min	4.00	1000.00	7.81	67.00	19.00	0.70
25%	5.00	30483.00	16.47	1197.00	789.00	4.50
50%	5.00	50000.00	18.80	1396.00	1262.00	6.70
75%	5.00	70000.00	21.40	1956.00	8876.00	13.54
max	8.00	810000.00	3996.00	7394.00	1186600.00	99.00

In [79]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1055 entries, 1109 to 1278
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   car_name            1055 non-null   object 
 1   insurance_validity  1055 non-null   object 
 2   fuel_type           1055 non-null   object 
 3   seats               1055 non-null   int64  
 4   kms_driven          1055 non-null   int64  
 5   ownsership          1055 non-null   object 
 6   transmission        1055 non-null   object 
 7   mileage(kmpl)       1055 non-null   float64
 8   engine(cc)          1055 non-null   float64
 9   torque(Nm)          1055 non-null   float64
 10  price(in lakhs)     1055 non-null   float64
 11  car_age             1055 non-null   object 
dtypes: float64(4), int64(2), object(6)
memory usage: 103.0+ KB
```

In [81]:
```
df['car_age']=df['car_age'].astype(int)
```

In [83]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1055 entries, 1109 to 1278
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   car_name            1055 non-null   object 
 1   insurance_validity  1055 non-null   object 
 2   fuel_type           1055 non-null   object 
 3   seats               1055 non-null   int64  
 4   kms_driven          1055 non-null   int64  
 5   ownsership          1055 non-null   object 
 6   transmission        1055 non-null   object 
 7   mileage(kmpl)       1055 non-null   float64
 8   engine(cc)          1055 non-null   float64
 9   torque(Nm)          1055 non-null   float64
 10  price(in lakhs)     1055 non-null   float64
 11  car_age             1055 non-null   int32  
dtypes: float64(4), int32(1), int64(2), object(5)
memory usage: 98.9+ KB
```

In [85]:
```
df.describe()
```

Out[85]:

	seats	kms_driven	mileage(kmpl)	engine(cc)	torque(Nm)	price(in lakhs)	car_age
count	1055.00	1055.00	1055.00	1055.00	1055.00	1055.00	1055.00
mean	5.18	53968.92	114.03	1634.13	14022.70	13.26	6.77
std	0.61	43778.72	347.04	974.43	88103.44	16.42	2.95
min	4.00	1000.00	7.81	67.00	19.00	0.70	1.00
25%	5.00	30483.00	16.47	1197.00	789.00	4.50	5.00
50%	5.00	50000.00	18.80	1396.00	1262.00	6.70	6.00
75%	5.00	70000.00	21.40	1956.00	8876.00	13.54	9.00
max	8.00	810000.00	3996.00	7394.00	1186600.00	99.00	17.00

In [87]:
```
df.drop(columns='torque(Nm)',inplace=True)
```

In [89]:
```
df
```

Out[89]:

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	mileage(kmpl)	engine(cc)	price(in lakhs)	c
1109	Fiat	No	Diesel	5	80000	Third Owner	Manual	20.30	1248.00	1.00	
1300	Porsche	No	Petrol	5	5700	First Owner	Automatic	2995.00	340.00	1.05	
464	Mercedes-Benz	Yes	Petrol	4	40000	First Owner	Automatic	7.81	4663.00	1.09	
412	Land	No	Petrol	5	4000	First Owner	Automatic	12.65	1997.00	1.10	
634	Audi	Yes	Petrol	5	7000	First Owner	Automatic	9.80	2995.00	1.12	
...	...	...	...	...	...	...	...	...	...	...	
480	BMW	Yes	Diesel	7	44000	First Owner	Automatic	13.38	2993.00	98.50	
221	Mercedes-Benz	Yes	Petrol	4	50000	First Owner	Automatic	7.81	4663.00	98.50	
1493	BMW	Yes	Petrol	7	28000	First Owner	Automatic	10.54	2998.00	99.00	
174	Ford	Yes	Petrol	5	66717	Second Owner	Manual	1196.00	70.00	0.70	
1278	Maruti	Yes	Petrol	5	44002	Third Owner	Manual	19.70	796.00	0.95	

1055 rows × 11 columns

In [91]:
```
df.describe()
```

Out[91]:

	seats	kms_driven	mileage(kmpl)	engine(cc)	price(in lakhs)	car_age
count	1055.00	1055.00	1055.00	1055.00	1055.00	1055.00
mean	5.18	53968.92	114.03	1634.13	13.26	6.77
std	0.61	43778.72	347.04	974.43	16.42	2.95
min	4.00	1000.00	7.81	67.00	0.70	1.00
25%	5.00	30483.00	16.47	1197.00	4.50	5.00
50%	5.00	50000.00	18.80	1396.00	6.70	6.00
75%	5.00	70000.00	21.40	1956.00	13.54	9.00
max	8.00	810000.00	3996.00	7394.00	99.00	17.00

In [93]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1055 entries, 1109 to 1278
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   car_name            1055 non-null   object 
 1   insurance_validity  1055 non-null   object 
 2   fuel_type           1055 non-null   object 
 3   seats               1055 non-null   int64  
 4   kms_driven          1055 non-null   int64  
 5   ownsership          1055 non-null   object 
 6   transmission        1055 non-null   object 
 7   mileage(kmpl)       1055 non-null   float64
 8   engine(cc)          1055 non-null   float64
 9   price(in lakhs)     1055 non-null   float64
 10  car_age             1055 non-null   int32  
dtypes: float64(3), int32(1), int64(2), object(5)
memory usage: 90.7+ KB
```

In [95]:
```
remove_mileage=df[df['mileage(kmpl)']>70].index
```

In [97]:
```
df.drop(remove_mileage,inplace=True)
```

In [99]:
```
df.index=np.arange(1,len(df)+1)
```

In [101…
```
df
```

Out[101…

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	mileage(kmpl)	engine(cc)	price(in lakhs)	ca
1	Fiat	No	Diesel	5	80000	Third Owner	Manual	20.30	1248.00	1.00	
2	Mercedes-Benz	Yes	Petrol	4	40000	First Owner	Automatic	7.81	4663.00	1.09	
3	Land	No	Petrol	5	4000	First Owner	Automatic	12.65	1997.00	1.10	
4	Audi	Yes	Petrol	5	7000	First Owner	Automatic	9.80	2995.00	1.12	
5	BMW	No	Petrol	7	7000	First Owner	Automatic	10.54	2998.00	1.15	
...	...	...	...	...	...	...	...	...	...	...	
965	Mercedes-Benz	No	Diesel	5	18346	First Owner	Automatic	13.50	2925.00	97.00	
966	BMW	Yes	Diesel	7	44000	First Owner	Automatic	13.38	2993.00	98.50	
967	Mercedes-Benz	Yes	Petrol	4	50000	First Owner	Automatic	7.81	4663.00	98.50	
968	BMW	Yes	Petrol	7	28000	First Owner	Automatic	10.54	2998.00	99.00	
969	Maruti	Yes	Petrol	5	44002	Third Owner	Manual	19.70	796.00	0.95	

969 rows × 11 columns

In [103…
```
df.describe()
```

Out[103…

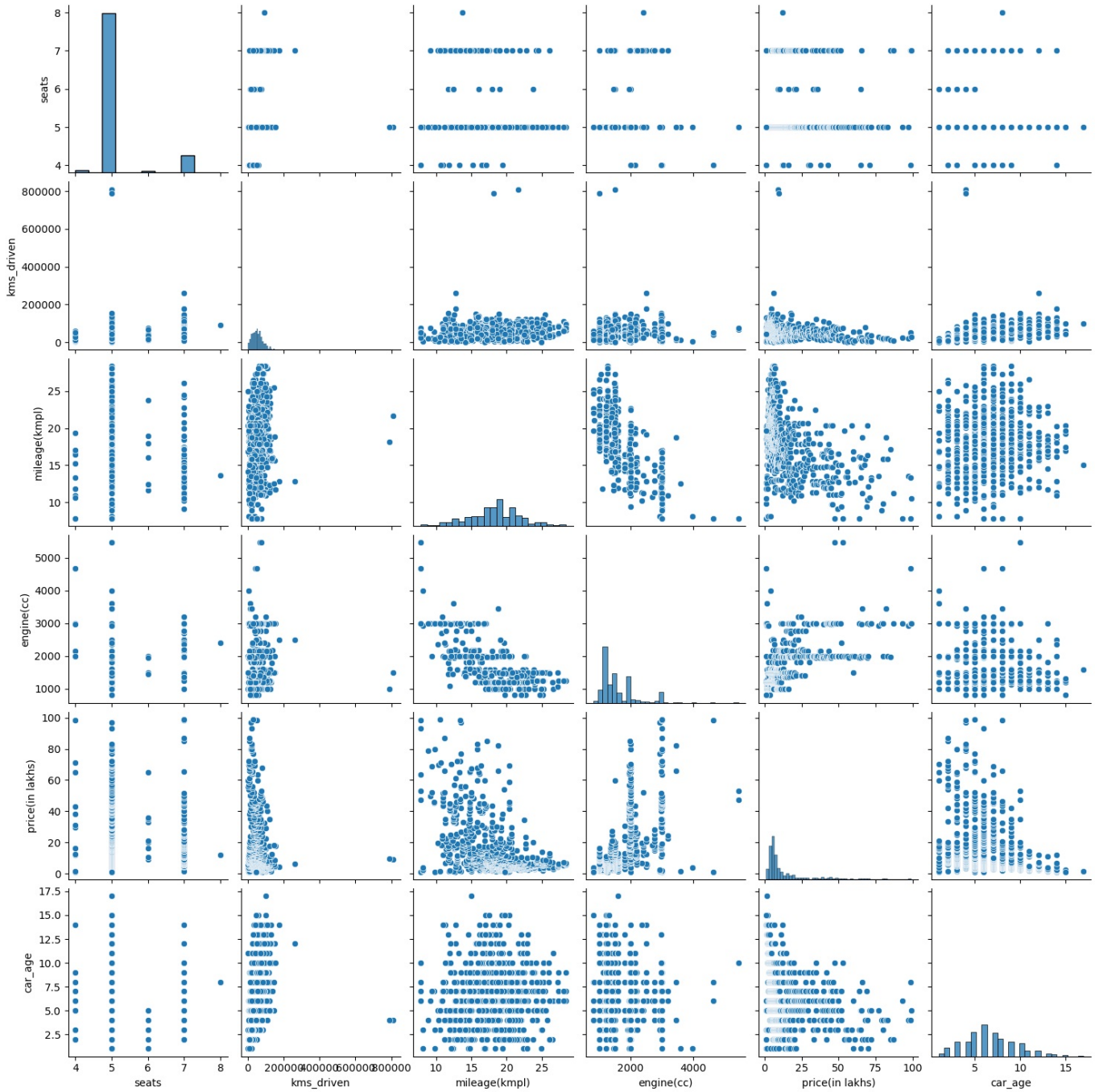
	seats	kms_driven	mileage(kmpl)	engine(cc)	price(in lakhs)	car_age
count	969.00	969.00	969.00	969.00	969.00	969.00
mean	5.19	54438.11	18.28	1588.95	13.90	6.62
std	0.62	45063.50	3.75	583.87	16.77	2.85
min	4.00	1000.00	7.81	796.00	0.95	1.00
25%	5.00	31000.00	16.10	1197.00	4.90	5.00
50%	5.00	50000.00	18.53	1461.00	6.95	6.00
75%	5.00	70000.00	20.70	1956.00	14.75	8.00
max	8.00	810000.00	28.40	5461.00	99.00	17.00

In [105…
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [107…
```
df_num=df[['seats','kms_driven','mileage(kmpl)','engine(cc)','price(in lakhs)','car_age']]
```

In [109…
```
sns.pairplot(df_num)
```

Out[109… <seaborn.axisgrid.PairGrid at 0x215f16af1f0>

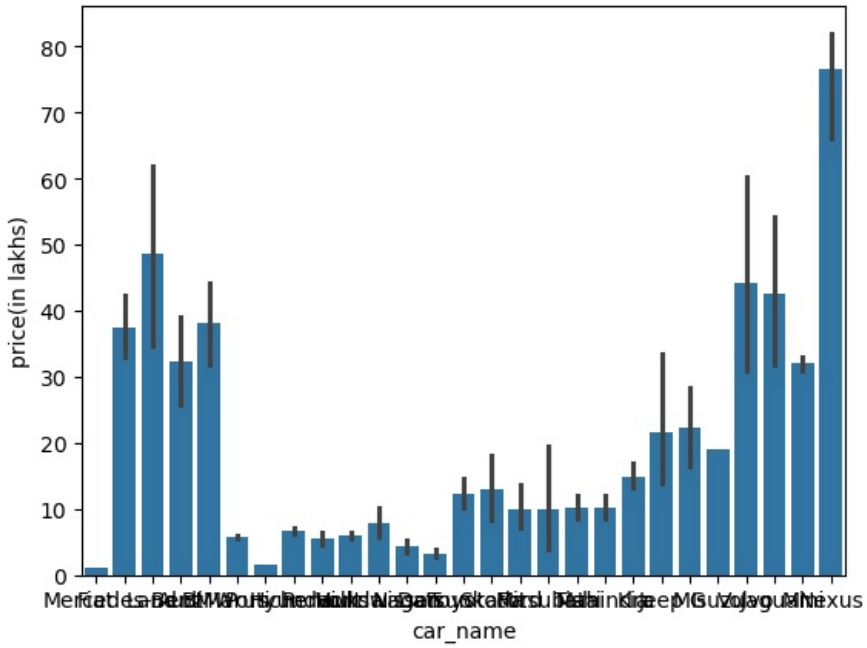


In [111…

```
sns.barplot(x=df['car_name'],y=df['price(in lakhs)'])
```

```
plt.figure(figsize=(200,20))
```

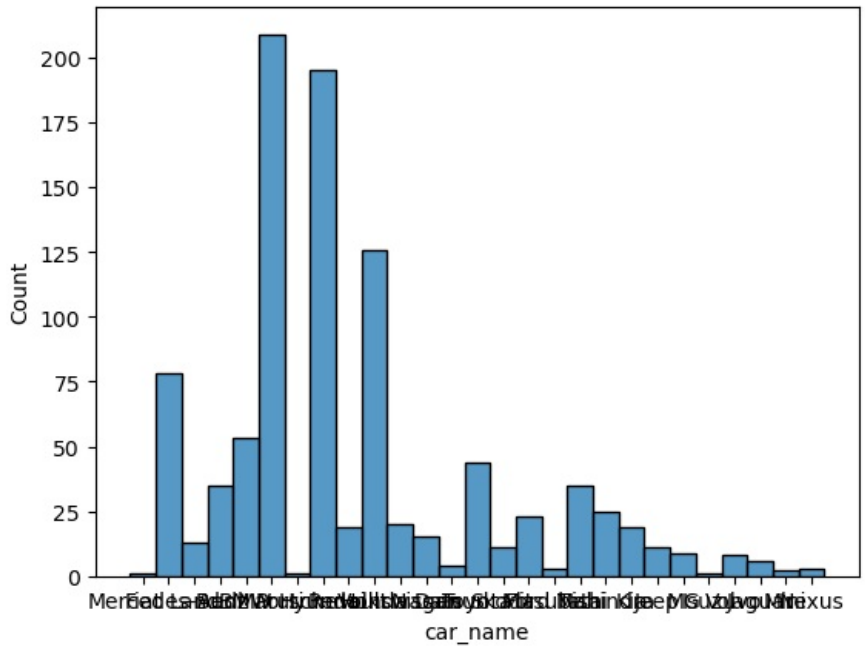
Out[111… <Figure size 20000x2000 with 0 Axes>



<Figure size 20000x2000 with 0 Axes>

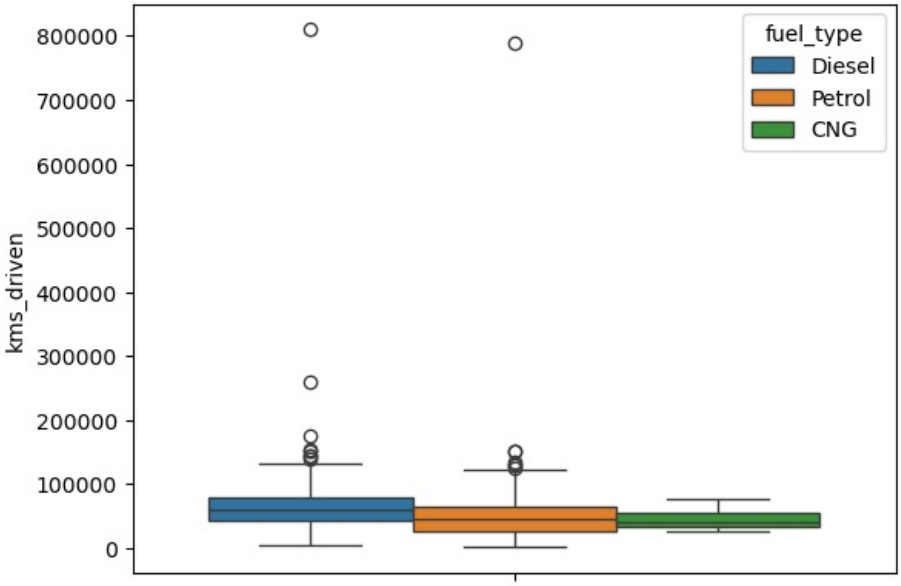
In [113…
```
sns.histplot(df.car_name)
```

Out[113… <Axes: xlabel='car_name', ylabel='Count'>



In [115…
```
sns.boxplot(y=df['kms_driven'],hue=df['fuel_type'])
```

Out[115… <Axes: ylabel='kms_driven'>



In [117…
```
remove_kms_outlier=df[df['kms_driven']>400000].index
```

In [119…
```
df.drop(remove_kms_outlier,inplace=True)
```

In [121…
```
from sklearn.preprocessing import LabelEncoder
```

In [123…
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 967 entries, 1 to 969
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   car_name            967 non-null    object 
 1   insurance_validity  967 non-null    object 
 2   fuel_type           967 non-null    object 
 3   seats               967 non-null    int64  
 4   kms_driven          967 non-null    int64  
 5   ownership           967 non-null    object 
 6   transmission        967 non-null    object 
 7   mileage(kmpl)       967 non-null    float64
 8   engine(cc)          967 non-null    float64
 9   price(in lakhs)     967 non-null    float64
 10  car_age             967 non-null    int32  
dtypes: float64(3), int32(1), int64(2), object(5)
memory usage: 83.1+ KB
```

In [125…
```
dfcopy1=df
```

In [127…
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

In [129…
```
le.fit_transform(df['car_name'])
for car_name,label in zip(le.classes_,le.transform(le.classes_)):
    print(f" Car Name : {car_name} , label : {label} ")
```

```
Car Name : Audi , label : 0
Car Name : BMW , label : 1
Car Name : Datsun , label : 2
Car Name : Fiat , label : 3
Car Name : Ford , label : 4
Car Name : Honda , label : 5
Car Name : Hyundai , label : 6
Car Name : Isuzu , label : 7
Car Name : Jaguar , label : 8
Car Name : Jeep , label : 9
Car Name : Kia , label : 10
Car Name : Land , label : 11
Car Name : MG , label : 12
Car Name : Mahindra , label : 13
Car Name : Maruti , label : 14
Car Name : Mercedes-Benz , label : 15
Car Name : Mini , label : 16
Car Name : Mitsubishi , label : 17
Car Name : Nexus , label : 18
Car Name : Nissan , label : 19
Car Name : Porsche , label : 20
Car Name : Renault , label : 21
Car Name : Skoda , label : 22
Car Name : Tata , label : 23
Car Name : Toyota , label : 24
Car Name : Volkswagen , label : 25
Car Name : Volvo , label : 26
```

In [131…
```
df
```

Out[131…

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	mileage(kmpl)	engine(cc)	price(in lakhs)	ca
1	Fiat	No	Diesel	5	80000	Third Owner	Manual	20.30	1248.00	1.00	
2	Mercedes-Benz	Yes	Petrol	4	40000	First Owner	Automatic	7.81	4663.00	1.09	
3	Land	No	Petrol	5	4000	First Owner	Automatic	12.65	1997.00	1.10	
4	Audi	Yes	Petrol	5	7000	First Owner	Automatic	9.80	2995.00	1.12	
5	BMW	No	Petrol	7	7000	First Owner	Automatic	10.54	2998.00	1.15	
...	...	...	...	...	...	...	...	...	...	...	
965	Mercedes-Benz	No	Diesel	5	18346	First Owner	Automatic	13.50	2925.00	97.00	
966	BMW	Yes	Diesel	7	44000	First Owner	Automatic	13.38	2993.00	98.50	
967	Mercedes-Benz	Yes	Petrol	4	50000	First Owner	Automatic	7.81	4663.00	98.50	
968	BMW	Yes	Petrol	7	28000	First Owner	Automatic	10.54	2998.00	99.00	
969	Maruti	Yes	Petrol	5	44002	Third Owner	Manual	19.70	796.00	0.95	

967 rows × 11 columns

In [133…
```
df['car_name']=le.fit_transform(df['car_name'])
```

In [135…
```
for car_name,label in zip(le.classes_,le.transform(le.classes_)):
    print(f" Car Name : {car_name} , label : {label} ")
```

```
Car Name : Audi , label : 0
Car Name : BMW , label : 1
Car Name : Datsun , label : 2
Car Name : Fiat , label : 3
Car Name : Ford , label : 4
Car Name : Honda , label : 5
Car Name : Hyundai , label : 6
Car Name : Isuzu , label : 7
Car Name : Jaguar , label : 8
Car Name : Jeep , label : 9
Car Name : Kia , label : 10
Car Name : Land , label : 11
Car Name : MG , label : 12
Car Name : Mahindra , label : 13
Car Name : Maruti , label : 14
Car Name : Mercedes-Benz , label : 15
Car Name : Mini , label : 16
Car Name : Mitsubishi , label : 17
Car Name : Nexus , label : 18
Car Name : Nissan , label : 19
Car Name : Porsche , label : 20
Car Name : Renault , label : 21
Car Name : Skoda , label : 22
Car Name : Tata , label : 23
Car Name : Toyota , label : 24
Car Name : Volkswagen , label : 25
Car Name : Volvo , label : 26
```

In [137…
```
df
```

Out[137…

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	mileage(kmpl)	engine(cc)	price(in lakhs)	ca
1	3	No	Diesel	5	80000	Third Owner	Manual	20.30	1248.00	1.00	
2	15	Yes	Petrol	4	40000	First Owner	Automatic	7.81	4663.00	1.09	
3	11	No	Petrol	5	4000	First Owner	Automatic	12.65	1997.00	1.10	
4	0	Yes	Petrol	5	7000	First Owner	Automatic	9.80	2995.00	1.12	
5	1	No	Petrol	7	7000	First Owner	Automatic	10.54	2998.00	1.15	
...	...	...	...	...	...	...	...	...	...	...	
965	15	No	Diesel	5	18346	First Owner	Automatic	13.50	2925.00	97.00	
966	1	Yes	Diesel	7	44000	First Owner	Automatic	13.38	2993.00	98.50	
967	15	Yes	Petrol	4	50000	First Owner	Automatic	7.81	4663.00	98.50	
968	1	Yes	Petrol	7	28000	First Owner	Automatic	10.54	2998.00	99.00	
969	14	Yes	Petrol	5	44002	Third Owner	Manual	19.70	796.00	0.95	

967 rows × 11 columns

In [139…
```
df['insurance_validity']=le.fit_transform(df['insurance_validity'])
```

In [141…
```
for insurance_validity,label in zip(le.classes_,le.transform(le.classes_)):
    print(f" Insurance : {insurance_validity} , label : {label} ")
```

```
Insurance : No , label : 0
Insurance : Yes , label : 1
```

In [143…
```
df['fuel_type']=le.fit_transform(df['fuel_type'])
```

In [145…
```
for fuel_type,label in zip(le.classes_,le.transform(le.classes_)):
    print(f" Fuel Type: {fuel_type},Label:{label}")
```

```
Fuel Type: CNG,Label:0
Fuel Type: Diesel,Label:1
Fuel Type: Petrol,Label:2
```

In [147…
```
df['transmission']=le.fit_transform(df['transmission'])
```

In [149…
```
for transmission,label in zip(le.classes_,le.transform(le.classes_)):
    print(f" Transmission :{transmission}, Label : {label}")
```

```
Transmission :Automatic, Label : 0
Transmission :Manual, Label : 1
```

In [151…
```
df['ownsership'].unique()
```

Out[151…
```
array(['Third Owner', 'First Owner', 'Second Owner'], dtype=object)
```

In [153…
```
df['ownsership']=df['ownsership'].replace('First Owner',1)
df['ownsership']=df['ownsership'].replace('Second Owner',2)
df['ownsership']=df['ownsership'].replace('Third Owner',3)
```

```
C:\Users\bittu\AppData\Local\Temp\ipykernel_22268\3074003180.py:3: FutureWarning: Downcasting behavior in `repla
ce` is deprecated and will be removed in a future version. To retain the old behavior, explicitly call `result.i
nfer_objects(copy=False)`. To opt-in to the future behavior, set `pd.set_option('future.no_silent_downcasting',
True)`
  df['ownsership']=df['ownsership'].replace('Third Owner',3)
```

In [155…
```
df
```

Out[155…

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	mileage(kmpl)	engine(cc)	price(in lakhs)	ca
1	3	0	1	5	80000	3	1	20.30	1248.00	1.00	
2	15	1	2	4	40000	1	0	7.81	4663.00	1.09	
3	11	0	2	5	4000	1	0	12.65	1997.00	1.10	
4	0	1	2	5	7000	1	0	9.80	2995.00	1.12	
5	1	0	2	7	7000	1	0	10.54	2998.00	1.15	
...	...	...	...	...	...	...	...	...	...	...	
965	15	0	1	5	18346	1	0	13.50	2925.00	97.00	
966	1	1	1	7	44000	1	0	13.38	2993.00	98.50	
967	15	1	2	4	50000	1	0	7.81	4663.00	98.50	
968	1	1	2	7	28000	1	0	10.54	2998.00	99.00	
969	14	1	2	5	44002	3	1	19.70	796.00	0.95	

967 rows × 11 columns

In [157…
```
df.corr()
```

Out[157…

	car_name	insurance_validity	fuel_type	seats	kms_driven	ownsership	transmission	mileage(kmpl)	engine(cc)
car_name	1.00	0.03	-0.03	0.11	-0.01	-0.06	0.05	0.00	0.01
insurance_validity	0.03	1.00	0.06	-0.06	-0.06	-0.14	0.06	0.05	-0.07
fuel_type	-0.03	0.06	1.00	-0.32	-0.24	-0.10	0.15	-0.13	-0.37
seats	0.11	-0.06	-0.32	1.00	0.12	0.04	-0.08	-0.27	0.33
kms_driven	-0.01	-0.06	-0.24	0.12	1.00	0.15	0.15	0.07	0.03
ownsership	-0.06	-0.14	-0.10	0.04	0.15	1.00	-0.05	-0.06	0.09
transmission	0.05	0.06	0.15	-0.08	0.15	-0.05	1.00	0.49	-0.53
mileage(kmpl)	0.00	0.05	-0.13	-0.27	0.07	-0.06	0.49	1.00	-0.64
engine(cc)	0.01	-0.07	-0.37	0.33	0.03	0.09	-0.53	-0.64	1.00
price(in lakhs)	-0.06	-0.07	-0.18	0.16	-0.25	0.01	-0.54	-0.48	0.64
car_age	-0.11	-0.02	-0.03	-0.05	0.54	0.25	0.24	0.05	-0.07

In [159…
```
X=df.drop(columns='price(in lakhs)')
```

In [161…
```
y=df['price(in lakhs)']
```

In [165…
```
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
```

In [175…
```
X['kms_driven']=scaler.fit_transform(X[['kms_driven']])
```

In [177…
```
X['mileage(kmpl)']=scaler.fit_transform(X[['mileage(kmpl)']])
X['engine(cc)']=scaler.fit_transform(X[['engine(cc)']])
```

In [179…
```
from sklearn.model_selection import train_test_split
```

In [181…
```
X_train,X_test,y_train,y_test=train_test_split(X,y,train_size=0.8)
```

In [183…
```
from sklearn.ensemble import RandomForestRegressor
```

In [185…
```
model=RandomForestRegressor()
```

In [187…
```
model.fit(X_train,y_train)
```

Out[187…
```
RandomForestRegressor
RandomForestRegressor()
```

In [189…
```
y_pred=model.predict(X_test)
```

In [191…
```
from sklearn.metrics import r2_score
```

In [193…
```
r2_score(y_test,y_pred)
```

Out[193… 0.5437430295938439

In [199…
```
from sklearn.model_selection import RandomizedSearchCV

param_distributions = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10]
}
```

In [251…
```
random_search = RandomizedSearchCV(
    estimator=model,
    param_distributions=param_distributions,
    n_iter=50,
    cv=10,
    scoring='neg_mean_squared_error',
    random_state=42
)
```

In [253…
```
random_search.fit(X_train, y_train)
```

```
C:\Users\bittu\anaconda3\lib\site-packages\sklearn\model_selection\_search.py:318: UserWarning: The total space of parameters 36 is smaller than n_iter=50. Running 36 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
```

Out[253…
```
RandomizedSearchCV
estimator: RandomForestRegressor
RandomForestRegressor
```

In [254…
```
y_pred_opt=random_search.predict(X_test)
```

In [255…
```
r2_score(y_test,y_pred_opt)
```

Out[255… 0.5690330728614846

In [259…
```
import joblib
```

In [261…
```
joblib.dump(model,'carspred.pkl')
```

Out[261… ['carspred.pkl']

In [ ]: