CAR PRICE PREDICTION WITH MACHINE LEARNING:

(Task-3)

In [1]:
```
#importing basic libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn import linear_model
```

In [2]:
```
data_set=pd.read_csv("C:\Users\HP\Desktop\Documents\oasis infobyte\car data.csv")
data_set.head(10)
```

Out[2]:

	Car_Name	Year	Selling_Price	Present_Price	Driven_kms	Fuel_Type	Selling_type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0
5	vitara brezza	2018	9.25	9.83	2071	Diesel	Dealer	Manual	0
6	ciaz	2015	6.75	8.12	18796	Petrol	Dealer	Manual	0
7	s cross	2015	6.50	8.61	33429	Diesel	Dealer	Manual	0
8	ciaz	2016	8.75	8.89	20273	Diesel	Dealer	Manual	0
9	ciaz	2015	7.45	8.92	42367	Diesel	Dealer	Manual	0
10	alto 800	2017	2.85	3.60	2135	Petrol	Dealer	Manual	0
11	ciaz	2015	6.85	10.38	51000	Diesel	Dealer	Manual	0
12	ciaz	2015	7.50	9.94	15000	Petrol	Dealer	Automatic	0
13	ertiga	2015	6.10	7.71	26000	Petrol	Dealer	Manual	0
14	dzire	2009	2.25	7.21	77427	Petrol	Dealer	Manual	0
15	ertiga	2016	7.75	10.79	43000	Diesel	Dealer	Manual	0
16	ertiga	2015	7.25	10.79	41678	Diesel	Dealer	Manual	0
17	ertiga	2016	7.75	10.79	43000	Diesel	Dealer	Manual	0
18	wagon r	2015	3.25	5.09	35500	CNG	Dealer	Manual	0
19	sx4	2010	2.65	7.98	41442	Petrol	Dealer	Manual	0
20	alto k10	2016	2.95	3.95	25000	Petrol	Dealer	Manual	0
21	ignis	2017	4.90	5.71	2400	Petrol	Dealer	Manual	0
22	sx4	2011	4.40	9.01	50000	Petrol	Dealer	Automatic	0
23	alto k10	2014	2.50	3.46	45280	Petrol	Dealer	Manual	0
24	wagon r	2013	2.90	4.41	56879	Petrol	Dealer	Manual	0
25	swift	2011	3.00	4.99	20000	Petrol	Dealer	Manual	0
26	swift	2013	4.15	5.87	55138	Petrol	Dealer	Manual	0
27	swift	2017	6.00	6.49	16200	Petrol	Individual	Manual	0
28	alto k10	2010	1.95	3.95	44542	Petrol	Dealer	Manual	0
29	ciaz	2015	7.45	10.38	45080	Diesel	Dealer	Manual	0
30	ritz	2012	3.10	5.98	51439	Diesel	Dealer	Manual	0
31	ritz	2011	2.35	4.89	54200	Petrol	Dealer	Manual	0
32	swift	2014	4.95	7.49	39000	Diesel	Dealer	Manual	0
33	ertiga	2014	6.00	9.95	45000	Diesel	Dealer	Manual	0
34	dzire	2014	5.50	8.06	45000	Diesel	Dealer	Manual	0
35	sx4	2011	2.95	7.74	49000	CNG	Dealer	Manual	0
36	dzire	2016	4.65	7.20	45767	Petrol	Dealer	Manual	0
37	800	2003	0.35	3.28	127000	Petrol	Individual	Manual	0
38	alto k10	2016	3.00	3.76	16874	Petrol	Dealer	Manual	0
39	sx4	2003	2.25	7.98	62000	Petrol	Dealer	Manual	0
40	baleno	2016	5.85	7.87	24524	Petrol	Dealer	Automatic	0
41	alto k10	2014	2.55	3.98	46706	Petrol	Dealer	Manual	0
42	sx4	2008	1.95	7.15	59000	Petrol	Dealer	Manual	0
43	dzire	2014	5.50	8.06	45780	Diesel	Dealer	Manual	0
44	omni	2012	1.25	2.69	50000	Petrol	Dealer	Manual	0
45	ciaz	2014	7.50	12.04	15000	Petrol	Dealer	Automatic	0
46	ritz	2013	2.65	4.89	64532	Petrol	Dealer	Manual	0
47	wagon r	2006	1.05	4.15	65000	Petrol	Dealer	Manual	0
48	ertiga	2019	9.80	7.71	25870	Petrol	Dealer	Manual	0
49	ciaz	2017	7.75	9.29	37000	Petrol	Dealer	Automatic	0

In [3]:
```
data_set.shape
```

Out[3]:
```
(301, 9)
```

In [4]:
```
data_set.isnull() .sum() #checking the null value
```

Out[4]:
```
Car_Name         0
Year             0
Selling_Price    0
Present_Price    0
Driven_kms       0
Fuel_Type        0
Selling_type     0
Transmission     0
Owner            0
dtype: int64
```

In [5]:
```
data_set.info()
```

Out[5]:
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Car_Name       301 non-null    object
 1   Year           301 non-null    int64
 2   Selling_Price  301 non-null    float64
 3   Present_Price  301 non-null    float64
 4   Driven_kms     301 non-null    int64
 5   Fuel_Type      301 non-null    object
 6   Selling_type   301 non-null    object
 7   Transmission   301 non-null    object
 8   Owner          301 non-null    int64
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
```

In [6]:
```
data_set.describe()
```

Out[6]:

	Year	Selling_Price	Present_Price	Driven_kms	Owner
count	301.000000	301.000000	301.000000	301.000000	301.000000
mean	2013.627907	4.661296	7.628472	36947.205980	0.043189
std	2.891554	5.082812	8.642584	38886.883882	0.247915
min	2003.000000	0.100000	0.320000	500.000000	0.000000
25%	2012.000000	0.900000	1.200000	15000.000000	0.000000
50%	2014.000000	3.600000	6.400000	32000.000000	0.000000
75%	2016.000000	6.000000	9.900000	48767.000000	0.000000
max	2018.000000	35.000000	92.600000	500000.000000	3.000000

In [7]:
```
data_set.columns
```

Out[7]:
```
Index(['Car_Name', 'Year', 'Selling_Price', 'Present_Price', 'Driven_kms',
       'Fuel_Type', 'Selling_type', 'Transmission', 'Owner'],
      dtype='object')
```

In [8]:
```

```

In [9]:
```

```

Data Modifications

In [10]:
```
input=data_set.drop(['Car_Name','Owner','Selling_type'],axis='columns')
input
```

Out[10]:

	Year	Selling_Price	Present_Price	Driven_kms	Fuel_Type	Transmission
0	2014	3.35	5.59	27000	Petrol	Manual
1	2013	4.75	9.54	43000	Diesel	Manual
2	2017	7.25	9.85	6900	Petrol	Manual
3	2011	2.85	4.15	5200	Petrol	Manual
4	2014	4.60	6.87	42450	Diesel	Manual
...	...	...	...	...	...	...
296	2016	9.50	11.60	33988	Diesel	Manual
297	2015	4.00	5.90	60000	Petrol	Manual
298	2009	3.35	11.00	87934	Petrol	Manual
299	2017	11.50	12.50	9000	Diesel	Manual
300	2016	5.30	5.90	5464	Petrol	Manual

301 rows × 6 columns

In [11]:
```
target=data_set.Selling_Price
target
```

Out[11]:
```
0      3.35
1      4.75
2      7.25
3      2.85
4      4.60
       ...
296    9.50
297    4.00
298    3.35
299   11.50
300    5.30
Name: Selling_Price, Length: 301, dtype: float64
```

In [12]:
```

```

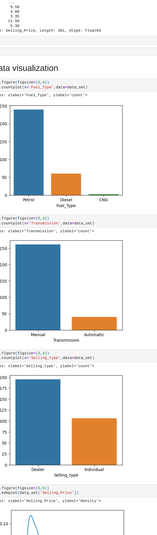
In [13]:
```

```

Data visualization

In [14]:
```
plt.figure(figsize=(5,4))
sns.countplot(x='Fuel_Type',data=data_set)
```

Out[14]:
```
<Axes: xlabel='Fuel_Type', ylabel='count'>
```



In [15]:
```
plt.figure(figsize=(5,4))
sns.countplot(x='Transmission',data=data_set)
```

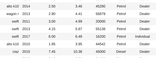
Out[15]:
```
<Axes: xlabel='Transmission', ylabel='count'>
```

In [16]:
```
plt.figure(figsize=(5,4))
sns.countplot(x='Selling_type',data=data_set)
```

Out[16]:
```
<Axes: xlabel='Selling_type', ylabel='count'>
```

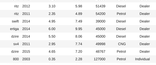

In [17]:
```
plt.figure(figsize=(5,5))
sns.kdeplot(data_set['Selling_Price'])
```

Out[17]:
```
<Axes: xlabel='Selling_Price', ylabel='Density'>
```



In [18]:
```
plt.figure(figsize=(5,5))
sns.kdeplot(data_set['Present_Price'])
```

Out[18]:
```
<Axes: xlabel='Present_Price', ylabel='Density'>
```



In [19]:
```
#data_set.drop(['Car_Name', 'Year', 'Driven_kms',
#       'Fuel_Type', 'Selling_type', 'Transmission', 'Owner'],
#       axis=1)
#
```

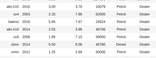
Out[19]:

	Selling_Price	Present_Price
0	3.35	5.59
1	4.75	9.54
2	7.25	9.85
3	2.85	4.15
4	4.60	6.87
...	...	...
296	9.50	11.60
297	4.00	5.90
298	3.35	11.00
299	11.50	12.50
300	5.30	5.90

301 rows × 2 columns

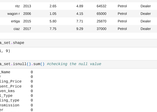
In [20]:
```
sns.kdeplot(z)
```

Out[20]:
```
<Axes: ylabel='Density'>
```



In [21]:
```
sns.heatmap(data_set.corr(),cmap='Blues')
```

Out[21]:
```
C:\Users\HP\AppData\Local\Temp\ipykernel_26408\2519437573.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will defau
lt to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  sns.heatmap(data_set.corr(),cmap='Blues')
<Axes: >
```



In [22]:
```

```

In [23]:
```

```

Training the model

In [24]:
```
#Encoding
from sklearn.preprocessing import LabelEncoder
Number=LabelEncoder()
```

In [25]:
```
input['Fuel_Type']=Number.fit_transform(input['Fuel_Type']) #encoded fuel type CNG-0 , Diesel-1, Petrol-2
input['Transmission']=Number.fit_transform(input['Transmission']) #encoded transmission type  Automatic-0 ,Manual-1
input
```

Out[25]:

	Year	Selling_Price	Present_Price	Driven_kms	Fuel_Type	Transmission
0	2014	3.35	5.59	27000	2	1
1	2013	4.75	9.54	43000	1	1
2	2017	7.25	9.85	6900	2	1
3	2011	2.85	4.15	5200	2	1
4	2014	4.60	6.87	42450	1	1
...	...	...	...	...	...	...
296	2016	9.50	11.60	33988	1	1
297	2015	4.00	5.90	60000	2	1
298	2009	3.35	11.00	87934	2	1
299	2017	11.50	12.50	9000	1	1
300	2016	5.30	5.90	5464	2	1

301 rows × 6 columns

In [26]:
```
model=linear_model.LinearRegression()
input=input.values
```

In [27]:
```
model.fit(input,target)
```

Out[27]:
```
LinearRegression()
```

In [28]:
```

```

In [29]:
```

```

Final result of prediction

In [30]:
```
predict=model.predict ([[2015,150,10,1000,2,0]])  # [Year,Selling_Price,Present_Price,Driven_kms,Fuel_Type,Transmission]
print("Car price predicted value:",predict[0])
                                    #encoded fuel type CNG-0 , Diesel-1, Petrol-2
                                    #encoded transmission type  Automatic-0 ,Manual-1
```

```
Car price predicted value: [1500.]
```

In [31]:
```

```

In [32]:
```

```

In [33]:
```
#Thanking You...
```

Thiruvalluvan G