In [58]:

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

In [59]:

```python
dataset=pd.read_csv('car data.csv')
```

In [60]:

```python
dataset.head(2)
```

Out[60]:

	Car_Name	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owne
0	ritz	2014	5.59	27000	Petrol	Dealer	Manual	(
1	sx4	2013	9.54	43000	Diesel	Dealer	Manual	(

In [61]:

```python
dataset.isnull().sum()
```

Out[61]:

```
Car_Name         0
Year             0
Present_Price    0
Kms_Driven       0
Fuel_Type        0
Seller_Type      0
Transmission     0
Owner            0
Selling_Price    0
dtype: int64
```

In [62]:

```python
dataset.info()
```

```
<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   Present_Price  301 non-null    float64
 3   Kms_Driven     301 non-null    int64  
 4   Fuel_Type      301 non-null    object 
 5   Seller_Type    301 non-null    object 
 6   Transmission   301 non-null    object 
 7   Owner          301 non-null    int64  
 8   Selling_Price  301 non-null    float64
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
```

Car_Name

In [63]:

```python
from sklearn.preprocessing import LabelEncoder
```

In [64]:
```
LE= LabelEncoder()
```

In [65]:
```
LE
```

Out[65]:
```
LabelEncoder()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [66]:
```
dataset['Car_Name']=LE.fit_transform(dataset['Car_Name'])
```

In [67]:
```
dataset.head()
```

Out[67]:

	Car_Name	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owne
0	90	2014	5.59	27000	Petrol	Dealer	Manual	(
1	93	2013	9.54	43000	Diesel	Dealer	Manual	(
2	68	2017	9.85	6900	Petrol	Dealer	Manual	(
3	96	2011	4.15	5200	Petrol	Dealer	Manual	(
4	92	2014	6.87	42450	Diesel	Dealer	Manual	(

Fuel_Type

In [68]:
```
dataset['Fuel_Type'].unique()
```

Out[68]:
```
array(['Petrol', 'Diesel', 'CNG'], dtype=object)
```

In [ ]:
```

```

In [69]:
```
dataset['Fuel_Type']=LE.fit_transform(dataset['Fuel_Type'])
```

In [70]:
```
dataset.head(1)
```

Out[70]:

	Car_Name	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owne
0	90	2014	5.59	27000	2	Dealer	Manual	(

Seller_Type

In [71]:
```
dataset['Seller_Type']=LE.fit_transform(dataset['Seller_Type'])
```

In [72]:
```
dataset.head(1)
```

Out[72]:

	Car_Name	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owne
0	90	2014	5.59	27000	2	0	Manual	(

Transmission

In [73]:
```
dataset['Transmission']=LE.fit_transform(dataset['Transmission'])
```

In [74]:
```
dataset.head(1)
```

Out[74]:

	Car_Name	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owne
0	90	2014	5.59	27000	2	0	1	(

In [75]:
```
x=dataset.iloc[:,:-1]
```

In [76]:
```
x
```

Out[76]:

	Car_Name	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Ow
0	90	2014	5.59	27000	2	0	1	
1	93	2013	9.54	43000	1	0	1	
2	68	2017	9.85	6900	2	0	1	
3	96	2011	4.15	5200	2	0	1	
4	92	2014	6.87	42450	1	0	1	
...	...	...	...	...	...	...	...	
296	69	2016	11.60	33988	1	0	1	
297	66	2015	5.90	60000	2	0	1	
298	69	2009	11.00	87934	2	0	1	
299	69	2017	12.50	9000	1	0	1	
300	66	2016	5.90	5464	2	0	1	

301 rows × 8 columns

In [77]:
```
y=dataset['Selling_Price']
```

In [78]:
```
y
```

Out[78]:
```
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 [79]:
```
from sklearn.preprocessing import StandardScaler
```

In [80]:
```
SS=StandardScaler()
```

In [81]:
```
SS
```

Out[81]:
```
StandardScaler()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [82]:
```
x=pd.DataFrame(SS.fit_transform(x), columns=x.columns)
```

In [83]:
```
x.head()
```

Out[83]:

	Car_Name	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	
0	1.074323	0.128897	-0.236215	-0.256224	0.500183	-0.737285	0.39148	-
1	1.191828	-0.217514	0.221505	0.155911	-1.852241	-0.737285	0.39148	-
2	0.212627	1.168129	0.257427	-0.773969	0.500183	-0.737285	0.39148	-
3	1.309332	-0.910335	-0.403079	-0.817758	0.500183	-0.737285	0.39148	-
4	1.152659	0.128897	-0.087890	0.141743	-1.852241	-0.737285	0.39148	-

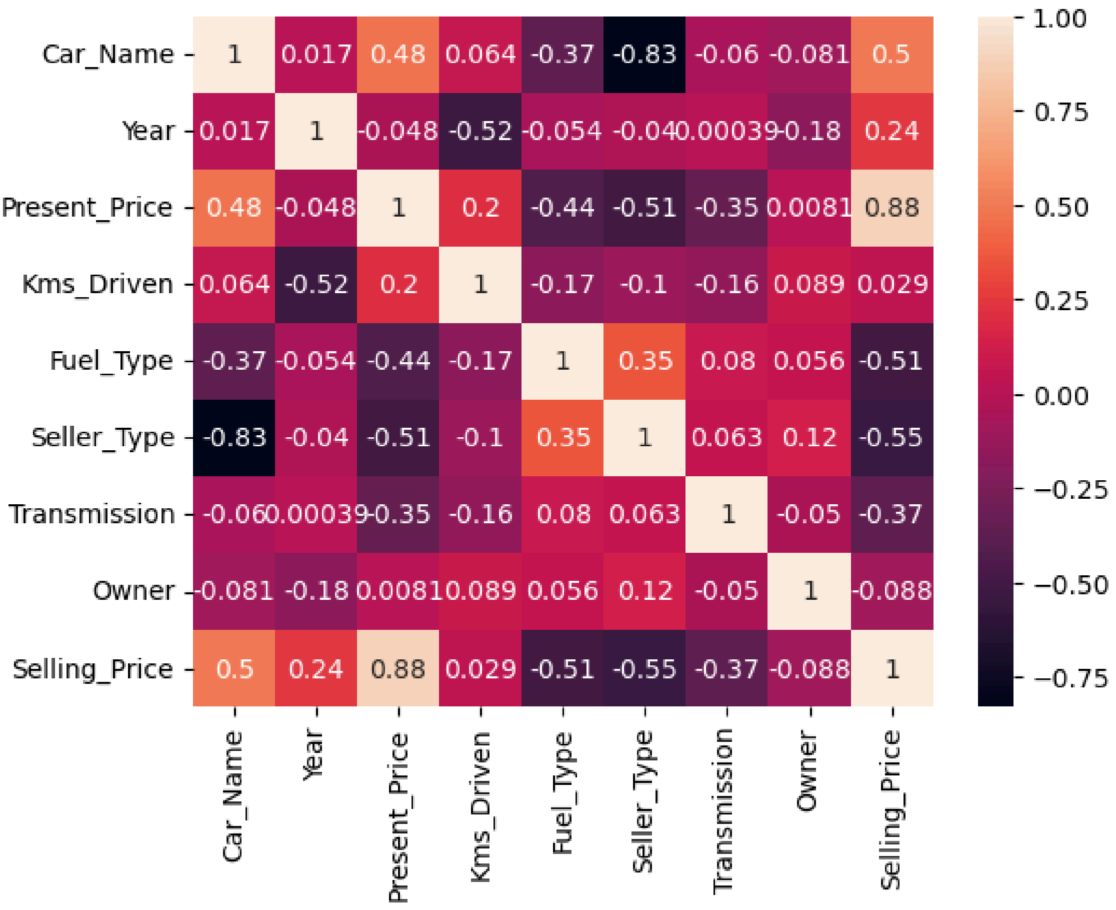
In [84]:
```
from sklearn.model_selection import train_test_split
```

In [85]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_sta
```

In [86]:
```
from sklearn.linear_model import LinearRegression, Lasso, Ridge, ElasticNet
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor
```

In [87]:
```
sns.heatmap(data=dataset.corr(),annot=True)
```

Out[87]:
```
<Axes: >
```



LinearRegression()

In [88]:
```
LR=LinearRegression()
```

In [89]:
```
LR
```

Out[89]:
```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [90]:
```
LR.fit(x_train,y_train)
LR.score(x_train,y_train)*100 , LR.score(x_test,y_test)*100
```

Out[90]:
```
(88.40630578239454, 84.65539666857805)
```

Lasso

In [91]:

```
LR1= Lasso(alpha=0.05)
LR1.fit(x_train,y_train)
LR1.score(x_train,y_train)*100 , LR1.score(x_test,y_test)*100
```

Out[91]:

```
(88.35433202380113, 84.42023265451037)
```

Ridge

In [92]:

```
LR2= Ridge(alpha=10)
LR2.fit(x_train,y_train)
LR2.score(x_train,y_train)*100 , LR2.score(x_test,y_test)*100
```

Out[92]:

```
(88.28628537091495, 84.16213595432282)
```

ElasticNet

In [93]:

```
LR3= ElasticNet(alpha=0.5)
LR3.fit(x_train,y_train)
LR3.score(x_train,y_train)*100 , LR3.score(x_test,y_test)*100
```

Out[93]:

```
(84.00059239671332, 78.3177718663528)
```

Decision Tree

In [94]:

```
dt=DecisionTreeRegressor(max_depth=13)
dt.fit(x_train,y_train)
dt.score(x_train,y_train)*100 , dt.score(x_test,y_test)*100
```

Out[94]:

```
(99.99983614119861, 95.47372436648206)
```

In [95]:

```
from sklearn.metrics import mean_squared_error, mean_absolute_error
```

In [96]:

```
mean_squared_error(y_test,dt.predict(x_test)), mean_absolute_error(y_test,d
```

Out[96]:

```
(1.042654049180328, 0.6585737704918032)
```

Random Forest Regressor

In [97]:

```
rf=RandomForestRegressor(n_estimators=100)
rf.fit(x_train,y_train)
rf.score(x_train,y_train)*100 , rf.score(x_test,y_test)*100
```

Out[97]:

```
(98.30481566012408, 96.41847578267432)
```

In [98]:

```
mean_squared_error(y_test,rf.predict(x_test)), mean_absolute_error(y_test,r
```

Out[98]:

```
(0.8250250381967219, 0.5829622950819674)
```

Support Vector regression

In [99]:

```
SV=SVR()
SV.fit(x_train,y_train)
SV.score(x_train,y_train)*100 , SV.score(x_test,y_test)*100
```

Out[99]:

```
(66.00840380338376, 78.48466914602926)
```

KNeighborsRegressor

In [100]:

```
Knn=KNeighborsRegressor()
Knn.fit(x_train,y_train)
Knn.score(x_train,y_train)*100 , Knn.score(x_test,y_test)*100
```

Out[100]:

```
(91.06681012800678, 93.29996797075346)
```

In [ ]:

In [101]:

```
rf.predict([[-1.275759,0.821718,-0.817924,-0.333500,0.500183,1.356327,-2.55
```

```
C:\Users\adity\anaconda3\New folder\Lib\site-packages\sklearn\base.py:464:
UserWarning: X does not have valid feature names, but RandomForestRegresso
r was fitted with feature names
  warnings.warn(
```

Out[101]:

```
array([0.4543])
```

In [102]:

```
y_test
```

Out[102]:

```
177     0.35
289    10.11
228     4.95
198     0.15
60      6.95
       ...  
234     5.50
296     9.50
281     2.10
285     7.40
182     0.30
Name: Selling_Price, Length: 61, dtype: float64
```

In [107]:

```
new_data=pd.DataFrame([['ritz',2014,5.59,27000,'Petrol','Dealer','Manual',0
new_data
```

Out[107]:

	Car_Name	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owne
0	0	2014	5.59	27000	Petrol	Dealer	Manual	

In [ ]:

```
```

In [109]:

```
new_data['Fuel_Type']=LE.fit_transform(new_data['Fuel_Type'])
```

In [110]:

```
new_data['Seller_Type']=LE.fit_transform(new_data['Seller_Type'])
```

In [111]:

```
new_data['Transmission']=LE.fit_transform(new_data['Transmission'])
```

In [112]:

```
new_data
```

Out[112]:

	Car_Name	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owne
0	0	2014	5.59	27000	0	0	0	(

In [113]:

```
new_data=pd.DataFrame(SS.transform(new_data), columns=new_data.columns)
```

In [114]:

```
new_data
```

Out[114]:

	Car_Name	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	
0	-2.4508	0.128897	-0.236215	-0.256224	-4.204665	-0.737285	-2.554408	-0

In [115]:

```
rf.predict(new_data)
```

Out[115]:

```
array([3.4745])
```

In [ ]:

```
```

In [ ]:

```
```