In [24]:

```python
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

In [25]:

```python
data=pd.read_csv("/home/placement/Downloads/sid.csv")
```

In [26]:

```python
data
```

Out[26]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon
0	1	lounge	51	882	25000	1	44.907242	8.611560
1	2	pop	51	1186	32500	1	45.666359	12.241890
2	3	sport	74	4658	142228	1	45.503300	11.417840
3	4	lounge	51	2739	160000	1	40.633171	17.634609
4	5	pop	73	3074	106880	1	41.903221	12.495650
...	...	...	...	...	...	...	...	.
1533	1534	sport	51	3712	115280	1	45.069679	7.704920
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870
1535	1536	pop	51	2223	60457	1	45.481541	9.413480
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270
1537	1538	pop	51	1766	54276	1	40.323410	17.568270

1538 rows × 9 columns

In [27]:

```python
list(data)
```

Out[27]:

```
['ID',
 'model',
 'engine_power',
 'age_in_days',
 'km',
 'previous_owners',
 'lat',
 'lon',
 'price']
```

In [28]:

```python
data1=data.drop(['lat',
 'lon','ID'],axis=1)
```

In [29]:

```
data1
```

Out[29]:

	model	engine_power	age_in_days	km	previous_owners	price
0	lounge	51	882	25000	1	8900
1	pop	51	1186	32500	1	8800
2	sport	74	4658	142228	1	4200
3	lounge	51	2739	160000	1	6000
4	pop	73	3074	106880	1	5700
...	...	...	...	...	...	...
1533	sport	51	3712	115280	1	5200
1534	lounge	74	3835	112000	1	4600
1535	pop	51	2223	60457	1	7500
1536	lounge	51	2557	80750	1	5990
1537	pop	51	1766	54276	1	7900

1538 rows × 6 columns

In [30]:

```
data1=pd.get_dummies(data1)
```

In [31]:

```
data1
```

Out[31]:

	engine_power	age_in_days	km	previous_owners	price	model_lounge	model_pop
0	51	882	25000	1	8900	1	0
1	51	1186	32500	1	8800	0	1
2	74	4658	142228	1	4200	0	0
3	51	2739	160000	1	6000	1	0
4	73	3074	106880	1	5700	0	1
...	...	...	...	...	...	...	...
1533	51	3712	115280	1	5200	0	0
1534	74	3835	112000	1	4600	1	0
1535	51	2223	60457	1	7500	0	1
1536	51	2557	80750	1	5990	1	0
1537	51	1766	54276	1	7900	0	1

1538 rows × 8 columns

In [32]:

```python
y=data1['price']
x=data1.drop('price',axis=1)
```

In [33]:

```python
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.33,random_state=42)
```

In [34]:

```python
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(x_train,y_train)
ypred=reg.predict(x_test)
```

In [35]:

```python
from sklearn.metrics import r2_score
r2_score(y_test,ypred)
```

Out[35]:

0.8415526986865394

In [36]:

```python
from sklearn.metrics import mean_squared_error
mean_squared_error(ypred,y_test)
```

Out[36]:

581887.727391353

In [37]:

```python
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Ridge
alpha=[1e-15,1e-10,1e-8,1e-4,1e-3,1e-2,1,5,10,20,30]
ridge=Ridge()
parameters={'alpha':alpha}
ridge_regressor=GridSearchCV(ridge,parameters)
ridge_regressor.fit(x_train,y_train)
```

Out[37]:

> **GridSearchCV**
> **estimator: Ridge**
>   > Ridge

In [38]:

```python
ridge_regressor.best_params_
```

Out[38]:

{'alpha': 30}

In [39]:

```python
ridge=Ridge(alpha=30)
ridge.fit(x_train,y_train)
y_pred_ridge=ridge.predict(x_test)
```

In [40]:

```python
r2_score(y_test,y_pred_ridge)
```

Out[40]:

0.8421969385523054

In [41]:

```python
mean_squared_error(y_pred_ridge,y_test)
```
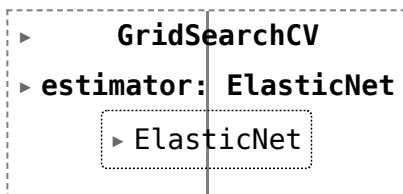
Out[41]:

579521.7970897449

In [42]:

```python
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import ElasticNet
elastic=ElasticNet()
parameters = {'alpha': [1e-15, 1e-10, 1e-8, 1e-4, 1e-3,1e-2, 1, 5, 10, 20]}
elastic_regressor = GridSearchCV(elastic, parameters)
elastic_regressor.fit(x_train, y_train)
```

Out[42]:

```
▸        GridSearchCV
▸ estimator:  ElasticNet
      ▸ ElasticNet
```

In [43]:

```python
elastic_regressor.best_params_
```

Out[43]:

{'alpha': 0.01}

In [44]:

```python
elas=ElasticNet(alpha=0.1)
elas.fit(x_train,y_train)#create elas constant with training data
y_pred_elas=elas.predict(x_test)#create predicted values in y_pred_elas
```

In [45]:

```python
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_elas)
```

Out[45]:

0.8425222843073693

In [46]:

```python
from sklearn.metrics import mean_squared_error
elastic_error=mean_squared_error(y_test,y_pred_elas)
elastic_error
```

Out[46]:

578326.9853103004

In [ ]: