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

In [133]:
```
data=pd.read_csv("/home/placement/Desktop/reddy/fiat500.csv")
```

In [134]:
```
data.describe()
```

Out[134]:

	ID	engine_power	age_in_days	km	previous_owners	lat	lon	price
count	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000
mean	769.500000	51.904421	1650.980494	53396.011704	1.123537	43.541361	11.563428	8576.003901
std	444.126671	3.988023	1289.522278	40046.830723	0.416423	2.133518	2.328190	1939.958641
min	1.000000	51.000000	366.000000	1232.000000	1.000000	36.855839	7.245400	2500.000000
25%	385.250000	51.000000	670.000000	20006.250000	1.000000	41.802990	9.505090	7122.500000
50%	769.500000	51.000000	1035.000000	39031.000000	1.000000	44.394096	11.869260	9000.000000
75%	1153.750000	51.000000	2616.000000	79667.750000	1.000000	45.467960	12.769040	10000.000000
max	1538.000000	77.000000	4658.000000	235000.000000	4.000000	46.795612	18.365520	11100.000000

In [135]:

```
data.tail(10)
```

Out[135]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
1528	1529	lounge	51	2861	126000	1	43.841980	10.51531	5500
1529	1530	lounge	51	731	22551	1	38.122070	13.36112	9900
1530	1531	lounge	51	670	29000	1	45.764648	8.99450	10800
1531	1532	sport	73	4505	127000	1	45.528511	9.59323	4750
1532	1533	pop	51	1917	52008	1	45.548000	11.54947	9900
1533	1534	sport	51	3712	115280	1	45.069679	7.70492	5200
1534	1535	lounge	74	3835	112000	1	45.845692	8.66687	4600
1535	1536	pop	51	2223	60457	1	45.481541	9.41348	7500
1536	1537	lounge	51	2557	80750	1	45.000702	7.68227	5990
1537	1538	pop	51	1766	54276	1	40.323410	17.56827	7900

In [5]:

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

In [6]:
```
data1
```

Out[6]:

	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 [7]:
```
data1['model']=data1['model'].map({'lounge':1,'pop':2,'sport':3})
```

In [8]:
```
data1
```

Out[8]:
```
      model  engine_power  age_in_days      km  previous_owners  price
   0      1            51          882   25000                1   8900
   1      2            51         1186   32500                1   8800
   2      3            74         4658  142228                1   4200
   3      1            51         2739  160000                1   6000
   4      2            73         3074  106880                1   5700
 ...    ...           ...          ...     ...              ...    ...
1533      3            51         3712  115280                1   5200
1534      1            74         3835  112000                1   4600
1535      2            51         2223   60457                1   7500
1536      1            51         2557   80750                1   5990
1537      2            51         1766   54276                1   7900

1538 rows × 6 columns
```

In [23]:
```
data2=data1.loc[(data.previous_owners==1)]
```

In [24]:
```
data2
```

Out[24]:
```
      model  engine_power  age_in_days      km  previous_owners  price
   0      1            51          882   25000                1   8900
   1      2            51         1186   32500                1   8800
   2      3            74         4658  142228                1   4200
   3      1            51         2739  160000                1   6000
   4      2            73         3074  106880                1   5700
 ...    ...           ...          ...     ...              ...    ...
1533      3            51         3712  115280                1   5200
1534      1            74         3835  112000                1   4600
1535      2            51         2223   60457                1   7500
1536      1            51         2557   80750                1   5990
1537      2            51         1766   54276                1   7900

1389 rows × 6 columns
```

In [32]:
```
y=data2['price']
x=data2.drop('price',axis=1)
```

In [33]:
```
y
```

Out[33]:
```
0       8900
1       8800
2       4200
3       6000
4       5700
        ... 
1533    5200
1534    4600
1535    7500
1536    5990
1537    7900
Name: price, Length: 1389, dtype: int64
```

In [34]:
```
x
```

Out[34]:

	model	engine_power	age_in_days	km	previous_owners
0	1	51	882	25000	1
1	2	51	1186	32500	1
2	3	74	4658	142228	1
3	1	51	2739	160000	1
4	2	73	3074	106880	1
...	...	...	...	...	...
1533	3	51	3712	115280	1
1534	1	74	3835	112000	1
1535	2	51	2223	60457	1
1536	1	51	2557	80750	1
1537	2	51	1766	54276	1

1389 rows × 5 columns

In [116]:

```
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 [117]:

```
x_test.head(5)
```

Out[117]:

	model	engine_power	age_in_days	km	previous_owners
625	1	51	3347	148000	1
187	1	51	4322	117000	1
279	2	51	4322	120000	1
734	2	51	974	12500	1
315	1	51	1096	37000	1

In [118]:

```
x_train.shape
```

Out[118]:

```
(930, 5)
```

In [119]:

```
y_train.shape
```

Out[119]:

```
(930,)
```

In [120]:

```
x_train.head()
```

Out[120]:

	model	engine_power	age_in_days	km	previous_owners
915	1	51	397	17081	1
12	1	51	456	18450	1
638	1	51	397	21276	1
190	1	51	821	19000	1
701	1	51	701	27100	1

In [121]:
```
y_train.head()
```

Out[121]:
```
915    10900
12      9700
638    10850
190     9990
701    10300
Name: price, dtype: int64
```

In [122]:
```
x_test.head()
```

Out[122]:

	model	engine_power	age_in_days	km	previous_owners
625	1	51	3347	148000	1
187	1	51	4322	117000	1
279	2	51	4322	120000	1
734	2	51	974	12500	1
315	1	51	1096	37000	1

In [123]:
```
y_test.head()
```

Out[123]:
```
625     5400
187     5399
279     4900
734    10500
315     9300
Name: price, dtype: int64
```

In [124]:
```
from sklearn.linear_model import ElasticNet
from sklearn.model_selection import GridSearchCV
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[124]:
```
GridSearchCV(estimator=ElasticNet(),
             param_grid={'alpha': [1e-15, 1e-10, 1e-08, 0.0001, 0.001, 0.01, 1,
                                   5, 10, 20]})
```

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 [125]:
```
elastic_regressor.best_params_
```

Out[125]:
```
{'alpha': 1e-15}
```

In [126]:
```
elastic=ElasticNet(alpha=1e-15)
elastic.fit(x_train,y_train)
y_pred_elastic=elastic.predict(x_test)
```

In [127]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_elastic)
```

Out[127]:
```
0.8582526737355334
```

In [128]:
```
from sklearn.metrics import mean_squared_error
Elasticnet_Error=mean_squared_error(y_pred_elastic,y_test)
Elasticnet_Error
```

Out[128]:
```
522589.16921946756
```

In [129]:

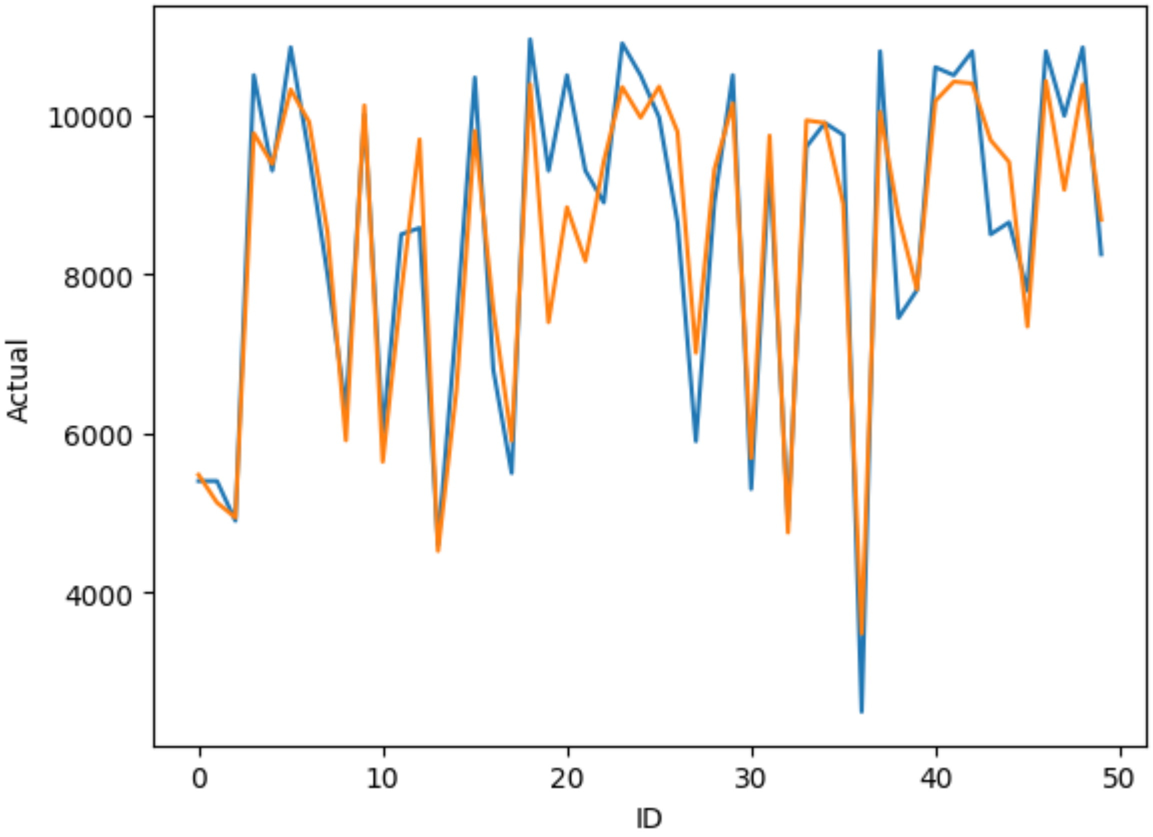
```
Results=pd.DataFrame(columns=['Actual','predicted'])
Results['Actual']=y_test
Results['predicted']=y_pred_elastic
Results=Results.reset_index()
Results['ID']=Results.index
Results.head(10)
```

Out[129]:

	index	Actual	predicted	ID
0	625	5400	5478.082470	0
1	187	5399	5128.749813	1
2	279	4900	4939.964669	2
3	734	10500	9770.938056	3
4	315	9300	9383.407921	4
5	652	10850	10319.804281	5
6	1472	9500	9912.760894	6
7	619	7999	8526.411840	7
8	992	6300	5910.610353	8
9	1154	10000	10119.997990	9

```
In [132]: import seaborn as sns#plot
          import matplotlib.pyplot as plt
          sns.lineplot(x='ID',y='Actual',data=Results.head(50))
          sns.lineplot(x='ID',y='predicted',data=Results.head(50))
          plt.plot()

Out[132]: []
```



```
In [ ]:

In [ ]:
```