In [1]:
```
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
data=pd.read_csv("/home/placement/Downloads/fiat500.csv")
```

In [3]:
```
data.describe()
```

Out[3]:

	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 [4]:
```
data.head(50)
```

Out[4]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1	lounge	51	882	25000	1	44.907242	8.611560	8900
1	2	pop	51	1186	32500	1	45.666359	12.241890	8800
2	3	sport	74	4658	142228	1	45.503300	11.417840	4200
3	4	lounge	51	2739	160000	1	40.633171	17.634609	6000
4	5	pop	73	3074	106880	1	41.903221	12.495650	5700
5	6	pop	74	3623	70225	1	45.000702	7.682270	7900
6	7	lounge	51	731	11600	1	44.907242	8.611560	10750
7	8	lounge	51	1521	49076	1	41.903221	12.495650	9190
8	9	sport	73	4049	76000	1	45.548000	11.549470	5600
9	10	sport	51	3653	89000	1	45.438301	10.991700	6000
10	11	pop	51	790	43286	1	40.871429	14.438960	8950
11	12	lounge	51	366	17500	1	45.069679	7.704920	10990
12	13	lounge	51	456	18450	1	45.426571	11.788130	9700
13	14	pop	51	3835	120000	1	40.531590	17.436159	4800
14	15	lounge	51	1035	40500	1	40.911362	14.211200	9300
15	16	lounge	51	1096	28200	1	45.697208	9.845970	9500
16	17	lounge	73	4200	110000	1	41.082352	14.254250	5250
17	18	pop	51	2223	96848	1	43.782372	11.254990	7990
18	19	lounge	51	2861	31000	1	45.069679	7.704920	7300
19	20	lounge	51	425	20030	1	45.354389	11.869260	10500
20	21	lounge	51	397	19037	1	45.707249	11.477600	10500
21	22	lounge	51	1886	110000	1	40.835812	14.504410	6990
22	23	lounge	51	1035	8000	1	44.506088	12.044170	10600
23	24	lounge	51	790	27595	1	45.688259	8.731450	10200

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
24	25	lounge	51	1583	14900	1	45.069679	7.704920	9990
25	26	lounge	51	366	9218	1	45.438110	12.318150	10800
26	27	pop	51	3592	124000	1	40.966179	17.116480	6800
27	28	sport	51	3531	100000	1	40.976452	14.172280	4950
28	29	lounge	51	762	28900	1	45.131672	8.449170	10640
29	30	lounge	51	670	4000	1	41.349751	13.353320	9500
30	31	lounge	62	2769	59216	1	43.782372	11.254990	6990
31	32	lounge	51	4169	99477	2	40.550564	14.225625	5900
32	33	lounge	51	821	21730	2	41.903221	12.495650	10500
33	34	sport	51	3927	140000	2	40.755932	14.690190	5200
34	35	lounge	51	640	32033	2	44.283878	11.888140	9790
35	36	pop	51	3653	138116	2	40.633171	17.634609	5000
36	37	pop	51	852	17000	1	45.505161	8.939100	8990
37	38	pop	51	3013	58527	1	45.688259	8.731450	7200
38	39	sport	51	790	43100	1	45.334080	11.376870	9950
39	40	pop	51	1858	13373	1	41.903221	12.495650	9000
40	41	sport	51	4139	119000	1	45.349319	7.742600	4890
41	42	pop	51	609	28500	1	45.746021	9.049970	10900
42	43	pop	51	1096	83000	1	41.959721	12.798056	7900
43	44	lounge	73	4049	98000	1	38.218128	15.240240	5999
44	45	lounge	51	456	12693	1	45.393600	10.482240	10900
45	46	lounge	51	762	14586	1	42.341969	12.358490	10500
46	47	lounge	51	821	27640	1	41.903221	12.495650	10400
47	48	lounge	51	2039	49000	1	41.903221	12.495650	7500
48	49	sport	51	3684	160000	1	45.405472	10.278290	4900
49	50	sport	51	4596	107000	1	40.845901	14.369270	4300

In [ ]:

In [5]:
```
data=data.loc[(data.previous_owners==1)]
```

In [6]:
```
data
```

Out[6]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1	lounge	51	882	25000	1	44.907242	8.611560	8900
1	2	pop	51	1186	32500	1	45.666359	12.241890	8800
2	3	sport	74	4658	142228	1	45.503300	11.417840	4200
3	4	lounge	51	2739	160000	1	40.633171	17.634609	6000
4	5	pop	73	3074	106880	1	41.903221	12.495650	5700
...	...	...	...	...	...	...	...	...	...
1533	1534	sport	51	3712	115280	1	45.069679	7.704920	5200
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870	4600
1535	1536	pop	51	2223	60457	1	45.481541	9.413480	7500
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270	5990
1537	1538	pop	51	1766	54276	1	40.323410	17.568270	7900

1389 rows × 9 columns

In [7]:
```
data1=data.drop(['lat','lon','ID'],axis=1)
```

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

Out[8]:

	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

1389 rows × 6 columns

In [9]:
```
data1=pd.get_dummies(data1)
```

In [10]:
```
data1.shape
```

Out[10]:
```
(1389, 8)
```

In [11]:
```
data1
```

Out[11]:

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

1389 rows × 8 columns

In [12]:
```
y=data1['price']
```

In [13]:
```
x=data1.drop('price',axis=1)
```

In [14]:
```
x
```

Out[14]:

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

1389 rows × 7 columns

In [ ]:
```

```

In [15]:
```
!pip3 install scikit_learn
```

```
Requirement already satisfied: scikit_learn in ./anaconda3/lib/python3.10/site-packages (1.2.1)
Requirement already satisfied: scipy>=1.3.2 in ./anaconda3/lib/python3.10/site-packages (from scikit_learn)
(1.10.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in ./anaconda3/lib/python3.10/site-packages (from sciki
t_learn) (2.2.0)
Requirement already satisfied: joblib>=1.1.1 in ./anaconda3/lib/python3.10/site-packages (from scikit_lear
n) (1.1.1)
Requirement already satisfied: numpy>=1.17.3 in ./anaconda3/lib/python3.10/site-packages (from scikit_lear
n) (1.23.5)
```

In [16]:
```
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 [17]:
```
x_test.head(5)
```

Out[17]:

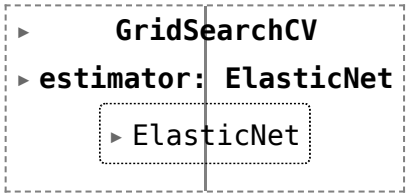
	engine_power	age_in_days	km	previous_owners	model_lounge	model_pop	model_sport
625	51	3347	148000	1	1	0	0
187	51	4322	117000	1	1	0	0
279	51	4322	120000	1	0	1	0
734	51	974	12500	1	0	1	0
315	51	1096	37000	1	1	0	0

In [18]:
```
y_test.head(5)
```

Out[18]:
```
625     5400
187     5399
279     4900
734    10500
315     9300
Name: price, dtype: int64
```

In [19]:
```
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[19]:



In [20]:
```
elastic_regressor.best_params_
```

Out[20]:
```
{'alpha': 0.01}
```

In [21]:
```
elastic=ElasticNet(alpha=.01)
elastic.fit(x_train,y_train)
y_pred_elastic=elastic.predict(x_test)
```

In [22]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_elastic)
```

Out[22]:
```
0.8602162350730707
```

In [23]:
```
from sklearn.metrics import mean_squared_error
```

In [24]:
```
elastic_Error=mean_squared_error(y_pred_elastic,y_test)
elastic_Error
```

Out[24]:
```
515349.9787871871
```

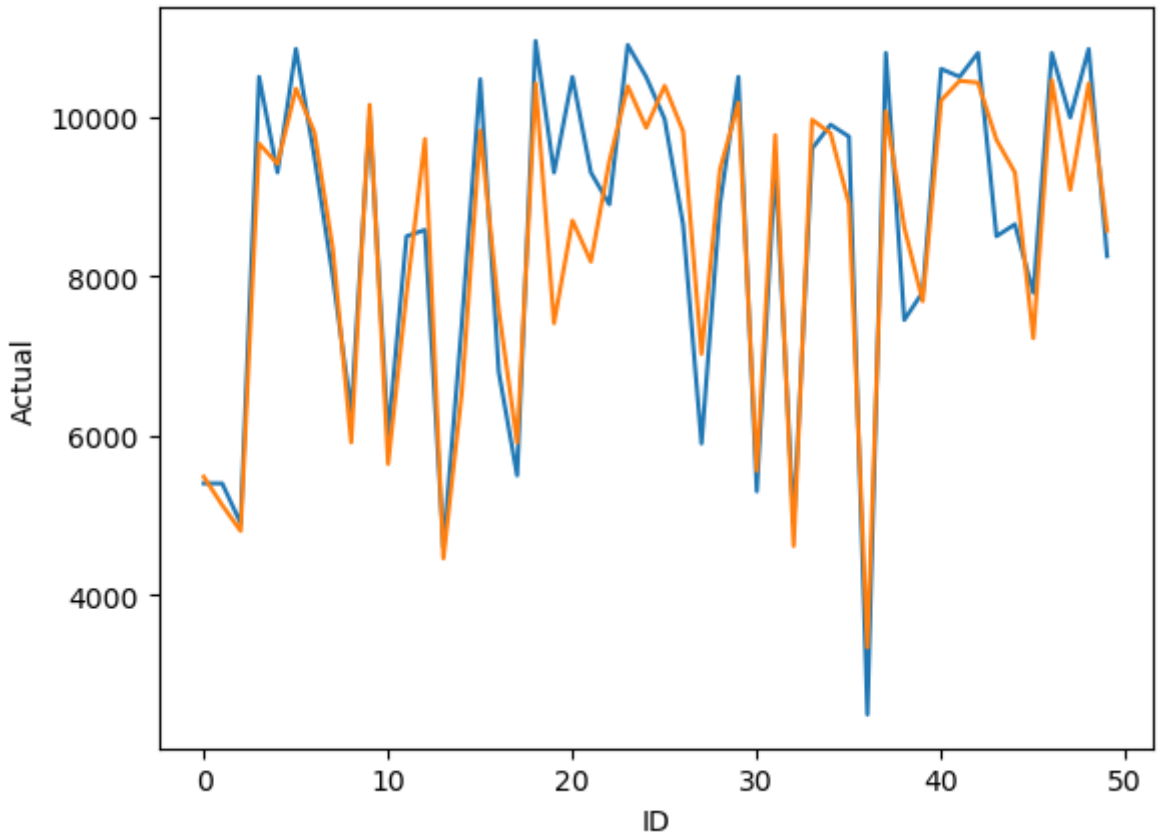
In [25]:
```
import seaborn as sns
Results=pd.DataFrame(columns=['Actual','Predicted'])
Results['Actual']=y_test
Results['Predicted']=y_pred_elastic
Results=Results.reset_index()
Results['ID']=Results.index
Results.head(10)
```

Out[25]:

	index	Actual	Predicted	ID
0	625	5400	5482.171479	0
1	187	5399	5127.531740	1
2	279	4900	4803.203231	2
3	734	10500	9662.825235	3
4	315	9300	9408.645424	4
5	652	10850	10350.952605	5
6	1472	9500	9806.127960	6
7	619	7999	8341.142824	7
8	992	6300	5913.786719	8
9	1154	10000	10149.093829	9

In [26]:
```
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[26]: []



In [ ]: