In [48]:
```
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

In [49]:
```
data=pd.read_csv("/home/placement/Desktop/csv/fiat500.csv")
```

In [50]:
```
data.describe()
```

Out[50]:

	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 [51]:
```
data.head()
```

Out[51]:

	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

In [52]:
```
data=data.drop(['ID','lat','lon'],axis=1)
data
```

Out[52]:

	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 [53]:
```
data=pd.get_dummies(data)
data
```

Out[53]:

	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

1538 rows × 8 columns

In [54]:
```
y=data['price']
x=data.drop(['price'],axis=1)
```

In [55]:

```
x
```

Out[55]:

	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

1538 rows × 7 columns

In [56]:
```
y
```

Out[56]:
```
0       8900
1       8800
2       4200
3       6000
4       5700
        ... 
1533    5200
1534    4600
1535    7500
1536    5990
1537    7900
Name: price, Length: 1538, dtype: int64
```

In [57]:
```
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 [58]:
```
x_test.shape
```

Out[58]:
```
(508, 7)
```

In [59]:
```
x_train.head(5)
```

Out[59]:

	engine_power	age_in_days	km	previous_owners	model_lounge	model_pop	model_sport
527	51	425	13111	1	1	0	0
129	51	1127	21400	1	1	0	0
602	51	2039	57039	1	0	1	0
331	51	1155	40700	1	1	0	0
323	51	425	16783	1	1	0	0

```
In [60]: y_test
```

```
Out[60]: 481      7900
         76       7900
         1502     9400
         669      8500
         1409     9700
                  ... 
         291     10900
         596      5699
         1489     9500
         1436     6990
         575     10900
         Name: price, Length: 508, dtype: int64
```

```
In [61]: y_test.head(5)
```

```
Out[61]: 481     7900
         76      7900
         1502    9400
         669     8500
         1409    9700
         Name: price, dtype: int64
```

```
In [62]: y_train.head(5)
```

```
Out[62]: 527    9990
         129    9500
         602    7590
         331    8750
         323    9100
         Name: price, dtype: int64
```

In [63]:

```
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[63]:

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

In [64]:

```
elastic_regressor.best_params_
```

Out[64]:

```
{'alpha': 0.01}
```

In [65]:

```
elastic=ElasticNet(alpha=30)
elastic.fit(x_train,y_train)
ypred=elastic.predict(x_test)
```

In [66]:

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

Out[66]:

```
0.8419757289065801
```

In [67]:

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

Out[67]:

```
580334.1755711779
```

In [72]:

```
results=pd.DataFrame(columns=['Actual','predicate'])
results['Actual']=y_test
results['predicate']=ypred
results=results.reset_index()
results['ID']=results.index
results.head(15)
```

Out[72]:

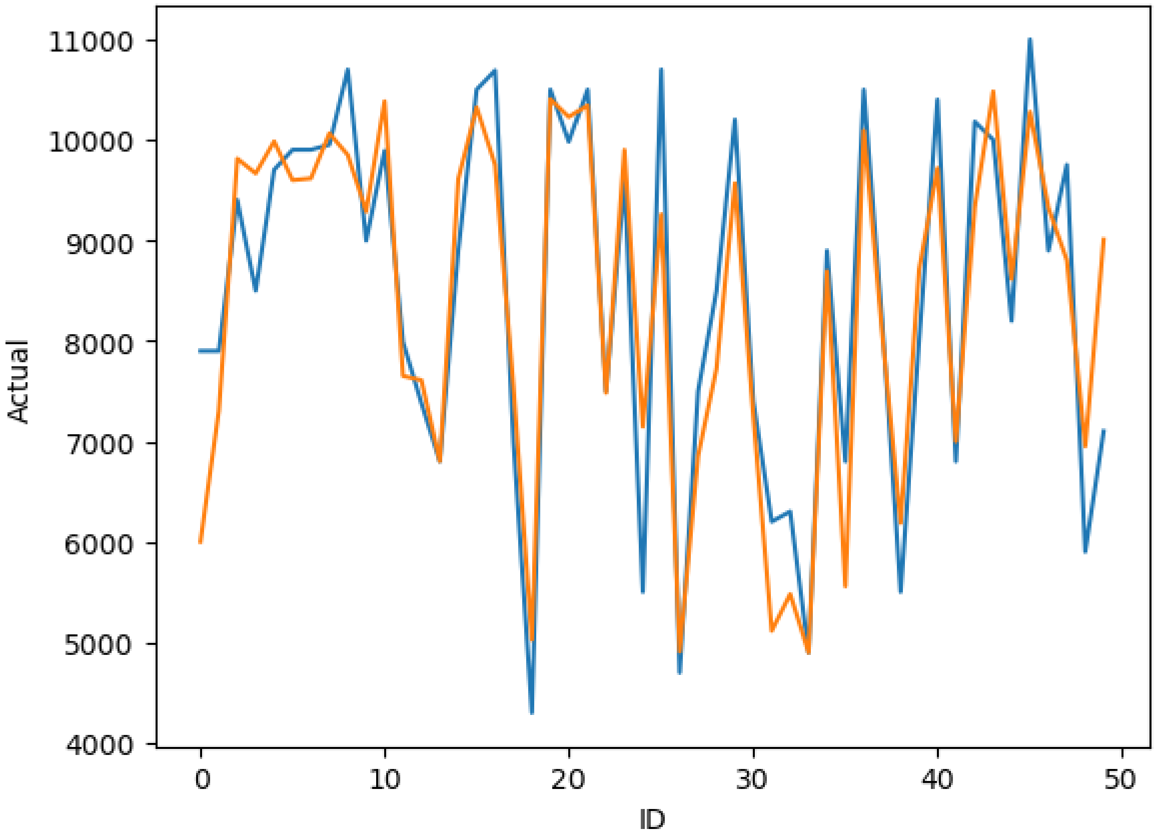
	index	Actual	predicate	ID
0	481	7900	5999.772939	0
1	76	7900	7307.696255	1
2	1502	9400	9811.206661	2
3	669	8500	9664.419998	3
4	1409	9700	9983.473801	4
5	1414	9900	9597.210309	5
6	1089	9900	9614.618393	6
7	1507	9950	10063.607164	7
8	970	10700	9848.342378	8
9	1198	8999	9288.542203	9
10	1088	9890	10383.330451	10
11	576	7990	7651.423422	11
12	965	7380	7608.722296	12
13	1488	6800	6802.226941	13
14	1432	8900	9606.916173	14

In [74]:

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.lineplot(x='ID' ,y='Actual' ,data=results.head(50))
sns.lineplot(x='ID' ,y='predicate' ,data=results.head(50))
```

Out[74]:

```
<Axes: xlabel='ID', ylabel='Actual'>
```



In [ ]: