In [23]:
```
import pandas as pd
```

In [24]:
```
data=pd.read_csv("fiat500 (1).csv")
```

In [25]:
```
data
```

Out[25]:

	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

1538 rows × 9 columns

In [26]:
```
data=data.drop(['lat','lon','ID',],axis=1)
```

In [27]:
```
data
```

Out[27]:

	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 [46]:
```
# ridge regression
```

In [47]:
```
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[47]:
```
GridSearchCV(estimator=Ridge(),
             param_grid={'alpha': [1e-15, 1e-10, 1e-08, 0.0001, 0.001, 0.01, 1,
                                   5, 10, 20, 30]})
```

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 [48]:
```
ridge_regressor.best_params_
```

Out[48]:
```
{'alpha': 30}
```

In [49]:
```
ridge=Ridge(alpha=30)
ridge.fit(X_train,y_train)
y_pred_ridge=ridge.predict(X_test)
```

In [50]:
```
Ridge_Error=mean_squared_error(y_pred_ridge,y_test)
Ridge_Error
```

Out[50]:
```
579521.7970897449
```

In [51]:
```
from sklearn.metrics import r2_score
r2_score(y_test,ypred)
```

Out[51]:
```
0.8415526986865394
```

In [52]:
```
Results= pd.DataFrame(columns=['Price','Predicted'])
Results['Price']=y_test
Results['Predicted']=y_pred_ridge
Results=Results.reset_index()
Results['Id']=Results.index
Results.head(25)
```

Out[52]:

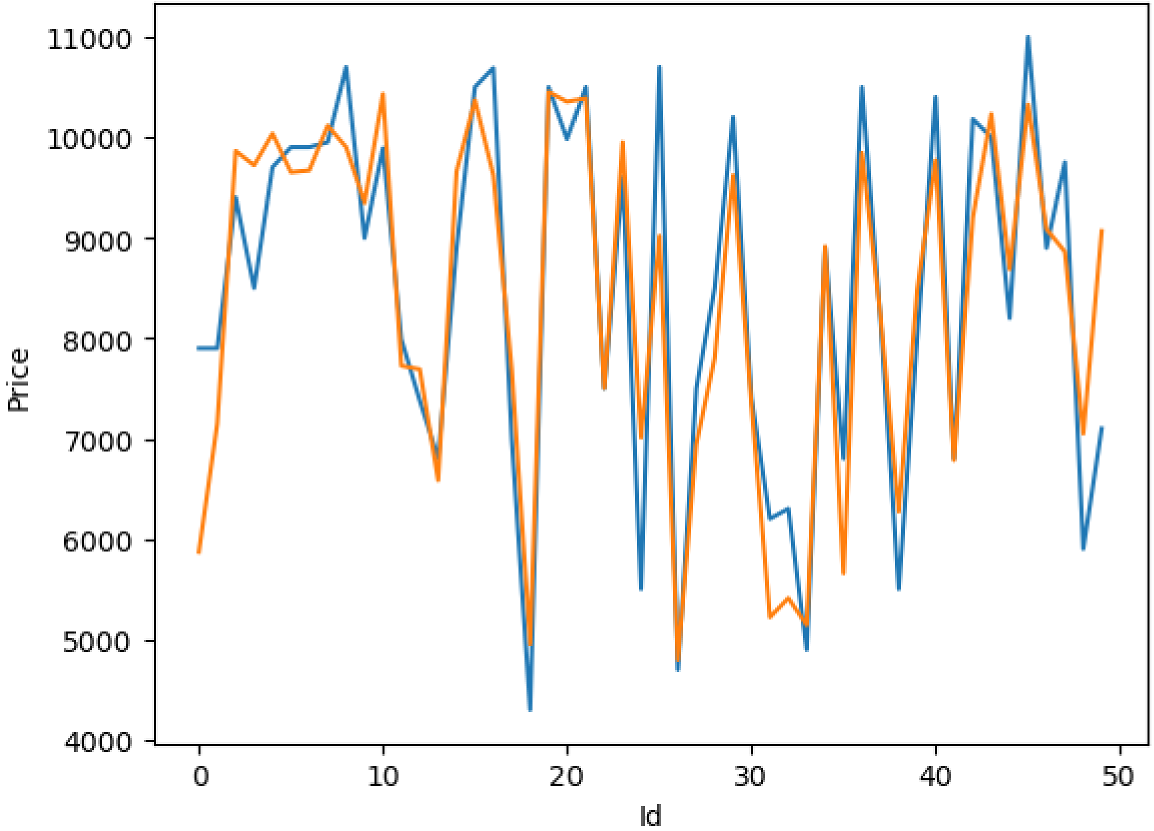
	index	Price	Predicted	Id
0	481	7900	5869.741155	0
1	76	7900	7149.563327	1
2	1502	9400	9862.785355	2
3	669	8500	9719.283532	3
4	1409	9700	10035.895686	4
5	1414	9900	9650.311090	5
6	1089	9900	9669.183317	6
7	1507	9950	10115.128380	7
8	970	10700	9900.241944	8
9	1198	8999	9347.080772	9
10	1088	9890	10431.237961	10
11	576	7990	7725.756431	11
12	965	7380	7691.089846	12
13	1488	6800	6583.674680	13
14	1432	8900	9659.240069	14
15	380	10500	10370.231518	15
16	754	10690	9620.427488	16
17	30	6990	7689.189244	17
18	49	4300	4954.595074	18
19	240	10500	10452.262871	19
20	344	9980	10353.107796	20

	index	Price	Predicted	Id
21	354	10500	10388.635632	21
22	124	7500	7503.302407	22
23	383	9600	9948.970588	23
24	1389	5500	7009.047336	24

In [53]:
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.lineplot(x='Id',y='Price',data=Results.head(50))
sns.lineplot(x='Id',y='Predicted',data=Results.head(50))
plt.plot()
```

Out[53]: []



In [54]:
```
# Elastic
```

In [55]:
```
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

In [56]:
```
data=pd.read_csv("fiat500 (1).csv")
```

In [57]:
```
data
```

Out[57]:

	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

1538 rows × 9 columns

In [58]:
```
data=data.loc[(data.previous_owners==1)]
```

In [59]:
```
data
```

Out[59]:

	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 [60]:
```
data=data.drop(['ID','lat','lon'],axis=1)
```

In [61]:
```
data
```

Out[61]:

	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 [62]:
```
data=pd.get_dummies(data)
```

In [63]:
```
data.shape
```

Out[63]: (1389, 8)

In [64]:
```
y=data['price']
x=data.drop('price',axis=1)
```

In [65]:
```
y
```

Out[65]:
```
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 [66]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.1,random_state=42)
```

In [67]:
```
X_test.head(5)
```

Out[67]:

	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 [68]:
```
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[68]:
```
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 [69]:
```
elastic_regressor.best_params_
```

Out[69]:
```
{'alpha': 0.01}
```

In [70]:
```
elastic=ElasticNet(alpha=0.01)
elastic.fit(X_train,y_train)
y_pred_elastic=elastic.predict(X_test)
```

In [71]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_elastic)
```

Out[71]:
```
0.8488682857174344
```

In [72]:
```
from sklearn.metrics import mean_squared_error
elastic_Error=mean_squared_error(y_pred_elastic,y_test)
elastic_Error
```

Out[72]:
```
603966.023413073
```

In [73]:
```
Results= pd.DataFrame(columns=['Price','Predicted'])
Results['Price']=y_test
Results['Predicted']=y_pred_elastic
Results=Results.reset_index()
Results['Id']=Results.index
Results.head(25)
```

Out[73]:

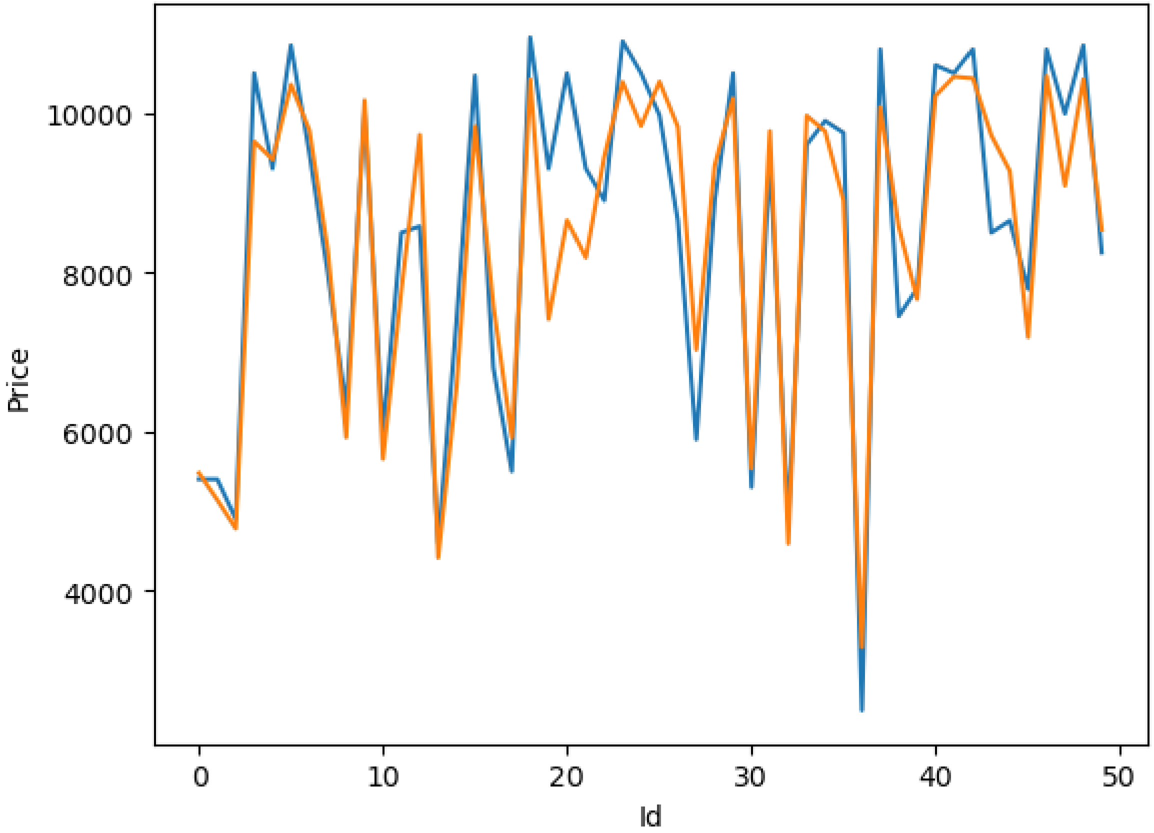
	index	Price	Predicted	Id
0	625	5400	5477.052458	0
1	187	5399	5137.435504	1
2	279	4900	4778.564980	2
3	734	10500	9640.895436	3
4	315	9300	9415.174300	4
5	652	10850	10356.323449	5
6	1472	9500	9781.272728	6
7	619	7999	8276.238400	7
8	992	6300	5925.267808	8
9	1154	10000	10158.433547	9
10	757	6000	5654.915390	10
11	1299	8500	7779.899617	11
12	400	8580	9724.510940	12
13	314	4600	4411.587148	13
14	72	7400	6568.196031	14
15	265	10470	9832.106012	15
16	800	6800	7576.247388	16
17	116	5500	5921.661919	17
18	181	10950	10422.823376	18
19	564	9300	7412.883090	19
20	1008	10500	8656.046516	20

	index	Price	Predicted	Id
21	1035	9300	8184.755615	21
22	1194	8900	9448.594403	22
23	131	10900	10388.473661	23
24	688	10499	9836.026696	24

In [74]:
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.lineplot(x='Id',y='Price',data=Results.head(50))
sns.lineplot(x='Id',y='Predicted',data=Results.head(50))
```

Out[74]: <Axes: xlabel='Id', ylabel='Price'>



In [ ]: