Elastic Regression

In [1]:
```
import pandas as pd
dt=pd.read_csv("/home/placemnet/YUVA/fiat500.csv")
dt
```

Out[1]:

	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 [2]:
```
dt1=dt.loc[(dt.previous_owners==1)]
dt1
```

Out[2]:

	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 [3]:
```
dt1=dt1.drop(['lat','lon','ID'],axis=1)
```

In [4]:
```
dt1=pd.get_dummies(dt1)
dt1
```

Out[4]:

	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 [5]:
```
a=dt1['price']
b=dt1.drop('price',axis=1)
b
```

Out[5]:

	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 [6]:
```
a
```

Out[6]:
```
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 [7]:
```
from sklearn.model_selection import train_test_split
a_train,a_test,b_train,b_test=train_test_split(a,b,test_size=0.1,random_state=42)
b_test
```

Out[7]:

	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
...	...	...	...	...	...	...	...
1507	51	701	17324	1	1	0	0
806	51	701	28975	1	1	0	0
1090	51	821	30510	1	1	0	0
436	51	2527	114000	1	1	0	0
937	51	2861	77550	1	0	1	0

139 rows × 7 columns

In [8]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import ElasticNet
import warnings
warnings.filterwarnings("ignore")
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(b_train, a_train)
```

Out[8]:
```
▾ GridSearchCV

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

▾ estimator: ElasticNet

ElasticNet()

▾ ElasticNet

ElasticNet()
```

In [9]:
```
elastic_regressor.best_params_
```

Out[9]:
```
{'alpha': 0.01}
```

In [10]:
```
elastic=ElasticNet(alpha=.01)
elastic.fit(b_train,a_train)
a_pred_elastic=elastic.predict(b_test)
```

In [11]:
```
from sklearn.metrics import r2_score    #to check the efficiency
r2_score(a_test,a_pred_elastic)
```

Out[11]:
```
0.8488682857174344
```

In [12]:
```
from sklearn.metrics import mean_squared_error
Elastic_error=mean_squared_error(a_pred_elastic,a_test)
Elastic_error
```

Out[12]:
```
603966.023413073
```

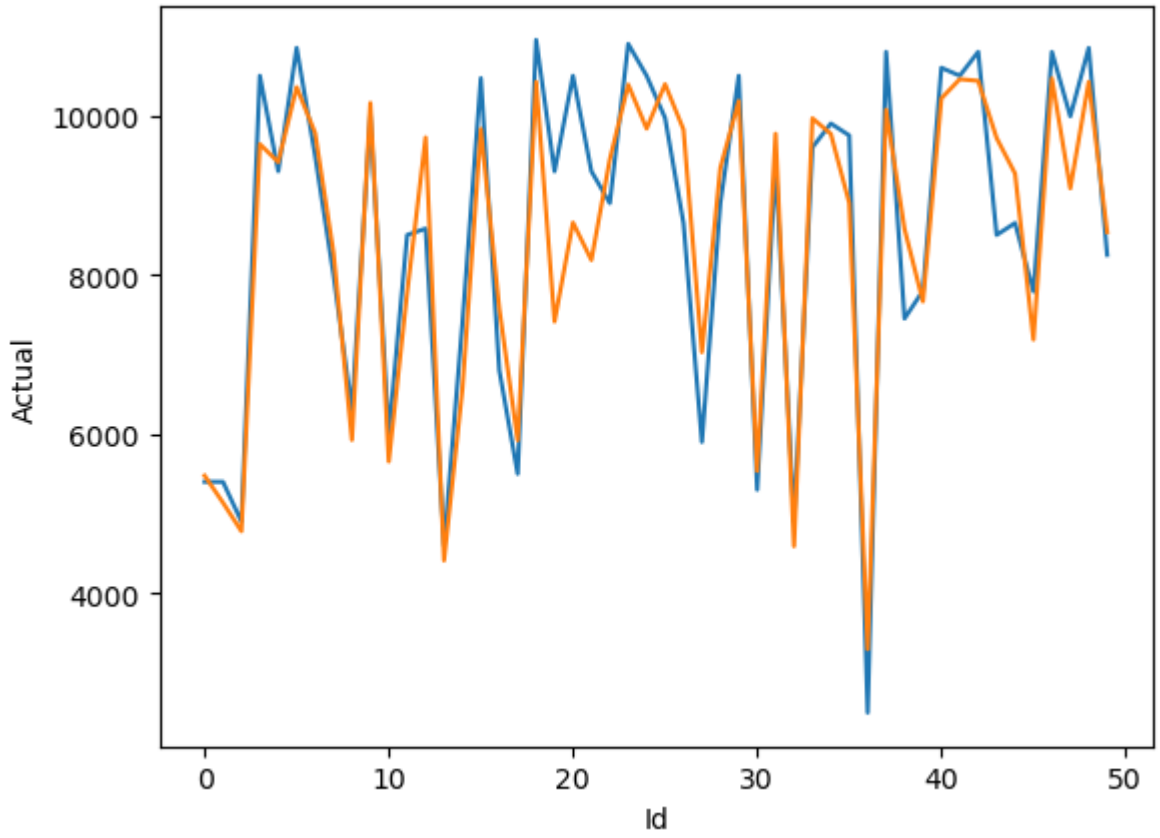
In [13]:
```
results=pd.DataFrame(columns=['Actual','Predicted'])   #to compare the actual and pedicted price
results['Actual']=a_test
results['Predicted']=a_pred_elastic
results=results.reset_index()
results['Id']=results.index
results.head()
```

Out[13]:

	index	Actual	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

In [14]:
```
import seaborn as sns
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[14]: []



In [ ]: