In [1]:
```
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
data=pd.read_csv("/home/placement/Downloads/fiat500.csv")
```

In [2]:
```
data1=data.drop(['lat','lon','ID'],axis=1)
data1
```

Out[2]:

	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 [3]:

```
data1=pd.get_dummies(data1)
data1
```

Out[3]:

	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 [4]:

```
y=data1['price'] #adding to seperate dataframe the value,we want to predict
x=data1.drop('price',axis=1)#removing the value we want to predict from the original dataframe
```

In [5]:

```
#divide data into training and testing
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)
```

linear regression

In [6]:
```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()#creating object of Linear Regression
reg.fit(x_train,y_train)#training and fitting LR object using training data
```

Out[6]:
```
▾ LinearRegression

LinearRegression()
```

In [7]:
```
ypred=reg.predict(x_test)
```

In [8]:
```
from sklearn.metrics import r2_score
r2_score(y_test,ypred)
```

Out[8]:
```
0.8415526986865394
```

In [9]:

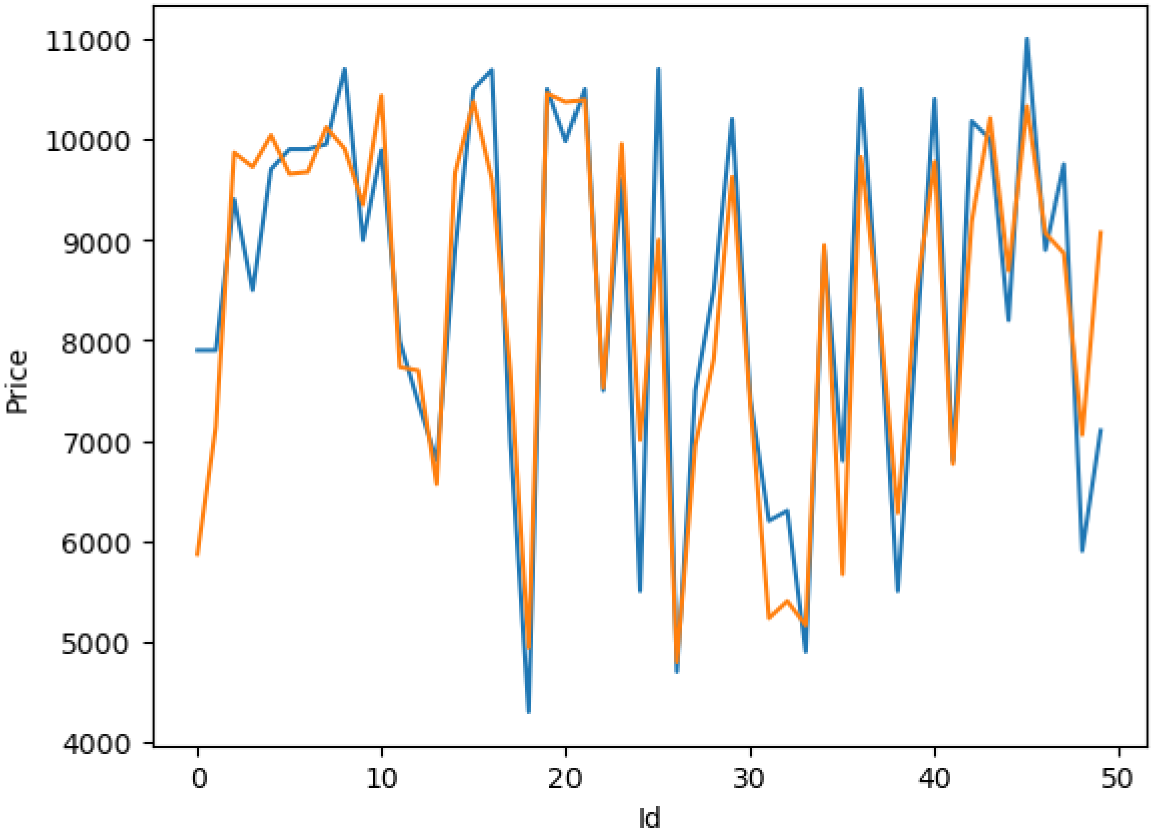
```
Results=pd.DataFrame(columns=['Price','Predicted'])
Results['Price']=y_test
Results['Predicted']=ypred
Results=Results.reset_index()
Results['Id']=Results.index
Results.head(15)
```

Out[9]:

	index	Price	Predicted	Id
0	481	7900	5867.650338	0
1	76	7900	7133.701423	1
2	1502	9400	9866.357762	2
3	669	8500	9723.288745	3
4	1409	9700	10039.591012	4
5	1414	9900	9654.075826	5
6	1089	9900	9673.145630	6
7	1507	9950	10118.707281	7
8	970	10700	9903.859527	8
9	1198	8999	9351.558284	9
10	1088	9890	10434.349636	10
11	576	7990	7732.262557	11
12	965	7380	7698.672401	12
13	1488	6800	6565.952404	13
14	1432	8900	9662.901035	14

In [10]:
```
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[10]:
```
[]
```



ridge regression

In [11]:
```
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[11]:
```
▸  GridSearchCV
▸ estimator: Ridge
      ▸ Ridge
```

In [12]:
```
ridge_regressor.best_params_
```

Out[12]:
```
{'alpha': 30}
```

In [13]:
```
ridge=Ridge(alpha=30)
ridge.fit(x_train,y_train)
y_pred_ridge=ridge.predict(x_test)
```

In [14]:
```
from sklearn.metrics import mean_squared_error #calculating MSE
Ridge_Error=mean_squared_error(y_pred_ridge,y_test)
Ridge_Error
```

Out[14]:
```
579521.7970897449
```

In [15]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_ridge)
```

Out[15]:
```
0.8421969385523054
```

In [16]:

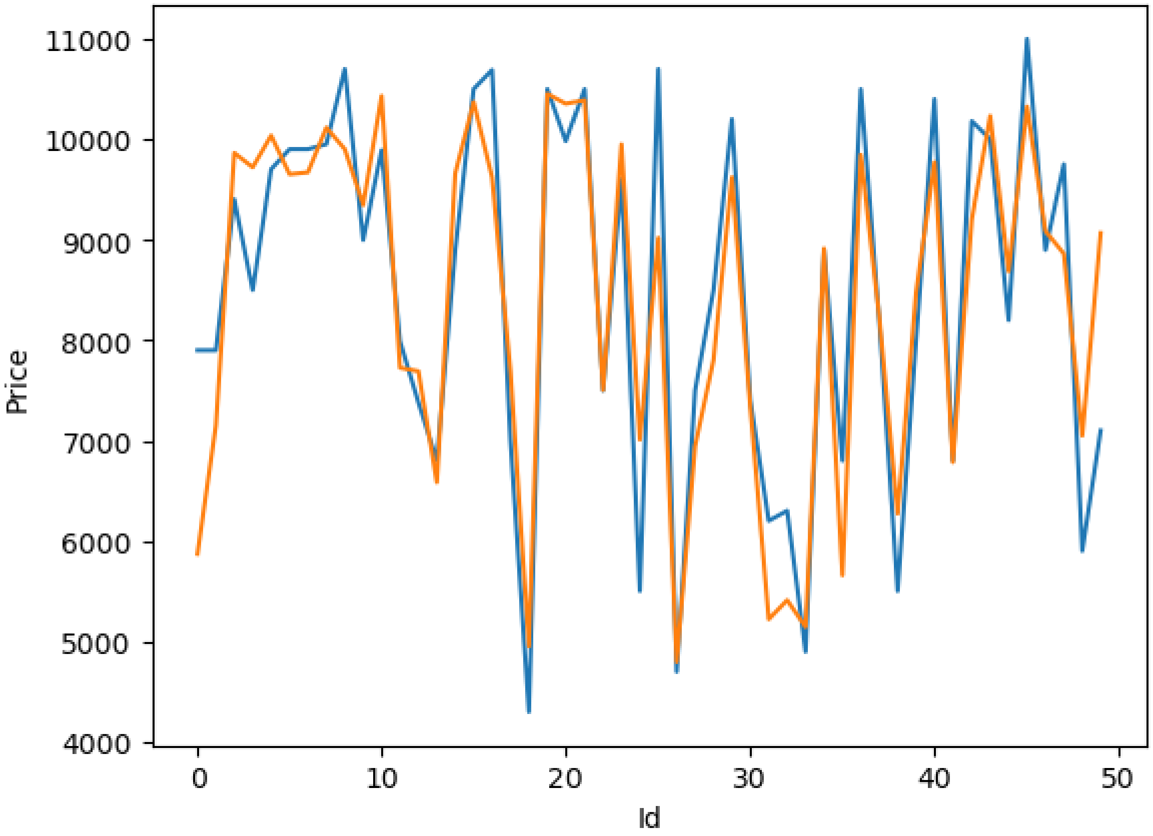
```
Results=pd.DataFrame(columns=['Price','Predicted'])
Results['Price']=y_test
Results['Predicted']=y_pred_ridge
Results=Results.reset_index()
Results['Id']=Results.index
Results.head(15)
```

Out[16]:

	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

In [17]:
```
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[17]:
```
[]
```



elastic net

In [18]:

```
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[18]:

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

In [19]:

```
elastic_regressor.best_params_
```

Out[19]:

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

In [20]:

```
elastic=ElasticNet(alpha=0.01)
elastic.fit(x_train,y_train)
y_pred_elastic=elastic.predict(x_test)
```

In [21]:

```
from sklearn.metrics import mean_squared_error #calculating MSE
elastic_error=mean_squared_error(y_pred_elastic,y_test)
elastic_error
```

Out[21]:

```
581390.7642825295
```

In [22]:

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

Out[22]:

```
0.841688021120299
```

In [23]:

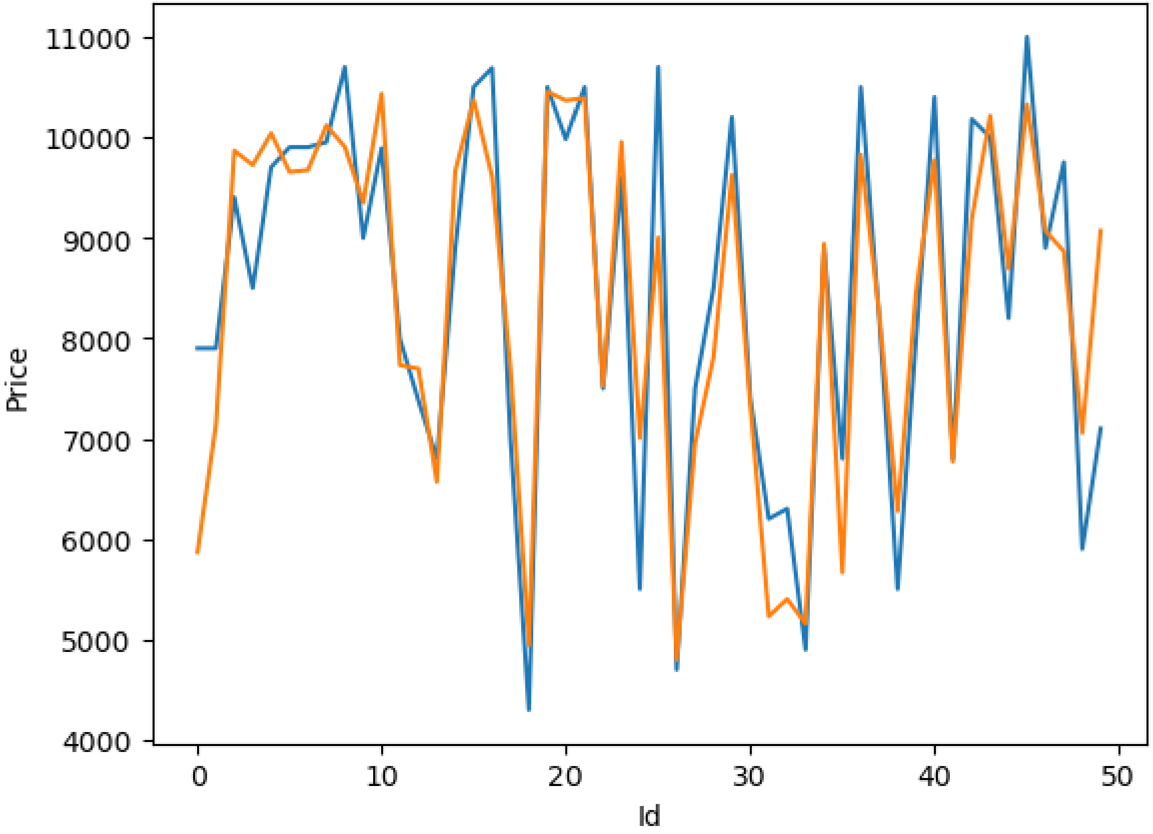
```
Results=pd.DataFrame(columns=['Price','Predicted'])
Results['Price']=y_test
Results['Predicted']=y_pred_elastic
Results=Results.reset_index()
Results['Id']=Results.index
Results.head(15)
```

Out[23]:

```
    index  Price     Predicted  Id
0     481   7900   5867.742075   0
1      76   7900   7136.527402   1
2    1502   9400   9865.726723   2
3     669   8500   9722.573593   3
4    1409   9700  10038.936496   4
5    1414   9900   9653.407122   5
6    1089   9900   9672.438692   6
7    1507   9950  10118.075470   7
8     970  10700   9903.219809   8
9    1198   8999   9350.750929   9
10   1088   9890  10433.808937  10
11    576   7990   7731.059127  11
12    965   7380   7697.260395  12
13   1488   6800   6569.177338  13
14   1432   8900   9662.252449  14
```

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



In [ ]: