In [12]:
```
import pandas as pd
```

In [13]:
```
import warnings
warnings.filterwarnings("ignore")
```

In [14]:
```
data=pd.read_csv("/home/placement/Desktop/EEE(238)/fiat500.csv")
```

In [15]:
```
data.describe()
```

Out[15]:

	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 [16]:
```
data=data.drop(['lat','lon','ID'],axis=1)
```

In [17]:
```
1 data
```

Out[17]:

	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 [18]:
```
1 data=pd.get_dummies(data)
```

In [19]:
```
1 data1
```

Out[19]:

	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 [20]:
```
1 data.shape
```

Out[20]: (1538, 8)

In [21]:
```
data
```

Out[21]:

	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 [23]:
```
y=data['price']
x=data.drop('price',axis=1)
```

In [24]:
```
1 y
```

Out[24]:
```
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 [25]:
```
1 x
```

Out[25]:

	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 [26]:
```
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 [27]:
```
x_test.head(5)
```

Out[27]:

	engine_power	age_in_days	km	previous_owners	model_lounge	model_pop	model_sport
481	51	3197	120000	2	0	1	0
76	62	2101	103000	1	0	1	0
1502	51	670	32473	1	1	0	0
669	51	913	29000	1	1	0	0
1409	51	762	18800	1	1	0	0

In [28]:
```
x_train.shape
```

Out[28]: (1030, 7)

In [29]:
```
x_train.head()
```

Out[29]:

	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 [30]:
```
1 y_train.head()
```

Out[30]:
```
527    9990
129    9500
602    7590
331    8750
323    9100
Name: price, dtype: int64
```

In [31]:
```
1 x_test.head()
```

Out[31]:

	engine_power	age_in_days	km	previous_owners	model_lounge	model_pop	model_sport
481	51	3197	120000	2	0	1	0
76	62	2101	103000	1	0	1	0
1502	51	670	32473	1	1	0	0
669	51	913	29000	1	1	0	0
1409	51	762	18800	1	1	0	0

In [32]:
```
1 y_test.head()
```

Out[32]:
```
481     7900
76      7900
1502    9400
669     8500
1409    9700
Name: price, dtype: int64
```

In [33]:
```
1 x_train
```

Out[33]:

	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
...	...	...	...	...	...	...	...
1130	51	1127	24000	1	1	0	0
1294	51	852	30000	1	1	0	0
860	51	3409	118000	1	0	1	0
1459	51	762	16700	1	1	0	0
1126	51	701	39207	1	1	0	0

1030 rows × 7 columns

In [34]:
```
1 y_train
```

Out[34]:
```
527      9990
129      9500
602      7590
331      8750
323      9100
         ...
1130    10990
1294     9800
860      5500
1459     9990
1126     8900
Name: price, Length: 1030, dtype: int64
```

In [35]:
```
1 x_test
```

Out[35]:

	engine_power	age_in_days	km	previous_owners	model_lounge	model_pop	model_sport
481	51	3197	120000	2	0	1	0
76	62	2101	103000	1	0	1	0
1502	51	670	32473	1	1	0	0
669	51	913	29000	1	1	0	0
1409	51	762	18800	1	1	0	0
...	...	...	...	...	...	...	...
291	51	701	22000	1	1	0	0
596	51	3347	85500	1	0	1	0
1489	51	366	22148	1	0	1	0
1436	51	1797	61000	1	1	0	0
575	51	366	19112	1	1	0	0

508 rows × 7 columns

In [36]:
```
1 y_test
```

Out[36]:
```
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
```

linear regression

In [37]:
```python
from sklearn.linear_model import LinearRegression
reg=LinearRegression()#creating object of LinearRegression
reg.fit(x_train,y_train)#training and fitting LR object using training data
```

Out[37]:
```
LinearRegression()
```

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 [38]:
```python
ypred=reg.predict(x_test)
```

In [39]:
```python
ypred
```

Out[39]:
```
array([ 5867.6503378 ,  7133.70142341,  9866.35776216,  9723.28874535,
       10039.59101162,  9654.07582608,  9673.14563045, 10118.70728123,
        9903.85952664,  9351.55828437, 10434.34963575,  7732.26255693,
        7698.67240131,  6565.95240435,  9662.90103518, 10373.20344286,
        9599.94844451,  7699.34400418,  4941.33017994, 10455.2719478 ,
       10370.51555682, 10391.60424404,  7529.06622456,  9952.37340054,
        7006.13845729,  9000.1780961 ,  4798.36770637,  6953.10376491,
        7810.39767825,  9623.80497535,  7333.52158317,  5229.18705519,
        5398.21541073,  5157.65652129,  8948.63632836,  5666.62365159,
        9822.1231461 ,  8258.46551788,  6279.2040404 ,  8457.38443276,
        9773.86444066,  6767.04074749,  9182.99904787, 10210.05195479,
        8694.90545226, 10328.43369248,  9069.05761443,  8866.7826029 ,
        7058.39787506,  9073.33877162,  9412.68162121, 10293.69451263,
       10072.49011135,  6748.5794244 ,  9785.95841801,  9354.09969973,
        9507.9444386 , 10443.01608254,  9795.31884316,  7197.84932877,
       10108.31707235,  7009.6597206 ,  9853.90699412,  7146.87414965,
        6417.69133992,  9996.97382441,  9781.18795953,  8515.83255277,
        8456.30006203,  6499.76668237,  7768.57829985,  6832.86406122,
        8347.96113362, 10439.02404036,  7356.43463051,  8562.56562053,
```

In [40]:
```
from sklearn.metrics import r2_score
r2_score(y_test,ypred)
```

Out[40]: 0.8415526986865394

In [41]:
```
from sklearn.metrics import mean_squared_error
t=mean_squared_error(ypred,y_test)
```

In [42]:

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

Out[42]:

	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 [43]:

```
Results['diff']=Results.apply(lambda row: row.Price - row.Predicted,axis=1)
```

In [44]:
```
Results
```

Out[44]:

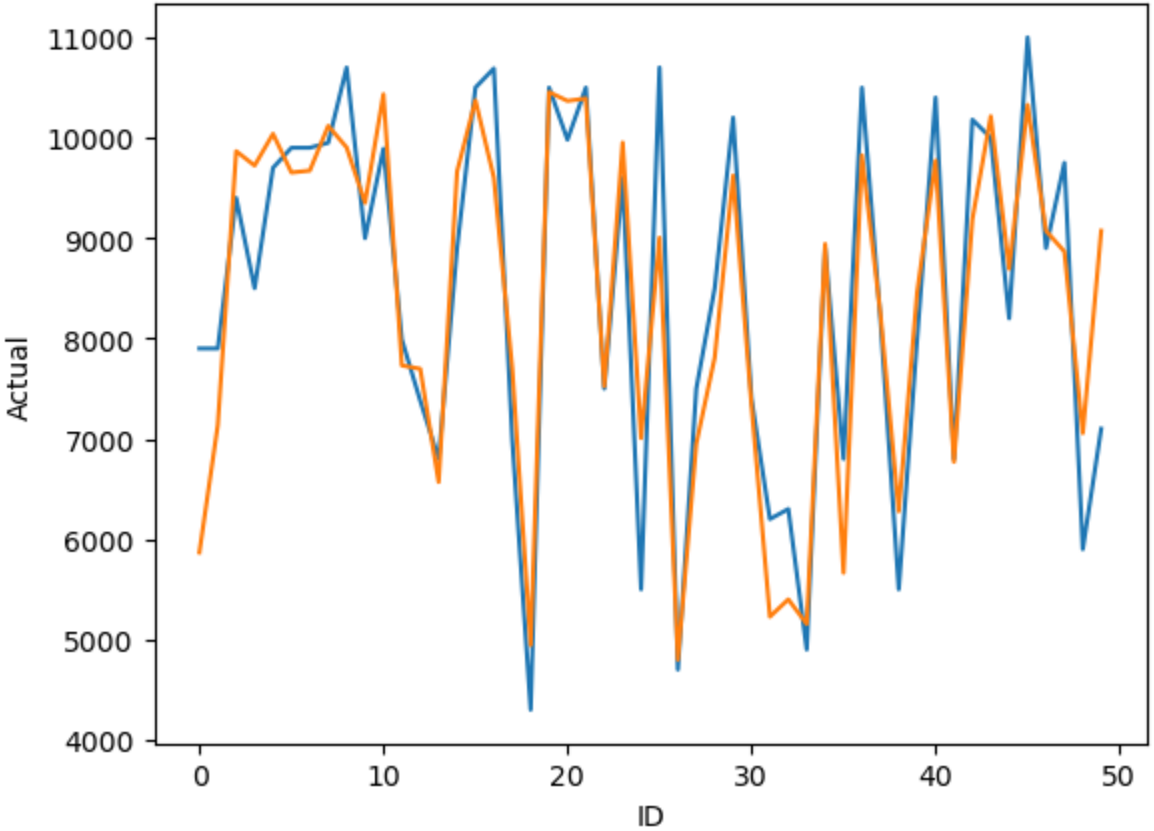
	index	Price	Predicted	Id	diff
0	481	7900	5867.650338	0	2032.349662
1	76	7900	7133.701423	1	766.298577
2	1502	9400	9866.357762	2	-466.357762
3	669	8500	9723.288745	3	-1223.288745
4	1409	9700	10039.591012	4	-339.591012
...	...	...	...	...	...
503	291	10900	10032.665135	503	867.334865
504	596	5699	6281.536277	504	-582.536277
505	1489	9500	9986.327508	505	-486.327508
506	1436	6990	8381.517020	506	-1391.517020
507	575	10900	10371.142553	507	528.857447

508 rows × 5 columns

In [63]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [64]:
```
sns.lineplot(x='ID',y='Actual',data=Results.head(50))
sns.lineplot(x='ID',y='Predicted',data=Results.head(50))
plt.plot
```

Out[64]: <function matplotlib.pyplot.plot(*args, scalex=True, scaley=True, data=None, **kwargs)>



ridge regression

In [45]:
```
from sklearn.model_selection import GridSearchCV#ridge regression
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[45]:
```
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 [46]:
```
ridge_regressor.best_params_
```

Out[46]:
```
{'alpha': 30}
```

In [47]:
```
ridge=Ridge(alpha=30)
ridge.fit(x_train,y_train)
y_pred_ridge=ridge.predict(x_test)
```

In [48]:
```
1 y_pred_ridge
```

Out[48]:
```
array([ 5869.74115507,  7149.56332694,  9862.78535486,  9719.28353248,
       10035.89568574,  9650.31109035,  9669.18331738, 10115.12838027,
        9900.24194354,  9347.08077182, 10431.23796139,  7725.75643127,
        7691.08984564,  6583.67468036,  9659.24006885, 10370.23151754,
        9620.42748841,  7689.18924428,  4954.59507446, 10452.26287068,
       10353.10779648, 10388.63563168,  7503.30240667,  9948.97058812,
        7009.04733578,  9020.73569412,  4798.12691579,  6944.67171049,
        7803.34446535,  9619.98788702,  7326.43443918,  5218.4077102 ,
        5408.53918256,  5141.35782797,  8914.90902841,  5656.63497772,
        9843.54231891,  8236.55007384,  6271.31566471,  8476.67006596,
        9770.02244191,  6784.29000107,  9203.55210535, 10231.79726073,
        8688.72507822, 10325.35487633,  9089.06645878,  8862.41881997,
        7048.7619628 ,  9068.9099975 ,  9409.53675932, 10290.6563444 ,
       10068.75380626,  6766.38650916,  9782.42178795,  9375.38267977,
        9528.4069177 , 10440.0567266 ,  9791.53263494,  7216.09577125,
       10104.686048  ,  7001.39195702,  9850.13133436,  7139.90750908,
        6408.14610807,  9993.32275333,  9777.34727934,  8535.02652876,
        8450.89417219,  6490.79570767,  7761.36847462,  6833.92199079,
        8342.12534099, 10436.01203789,  7349.55597282,  8557.12693543,
```

In [49]:
```
1 y_test
```

Out[49]:
```
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 [50]:
```
from sklearn.metrics import mean_squared_error
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,y_pred_ridge)
```

Out[51]: 0.8421969385523054

In [52]:
```
Results=pd.DataFrame(columns=['Actual','Predicted'])
Results['Actual']=y_test
Results['Predicted']=y_pred_ridge
#Results['Km']=x_test['Km']
Results=Results.reset_index()
Results['ID']=Results.index
Results.head(10)
```

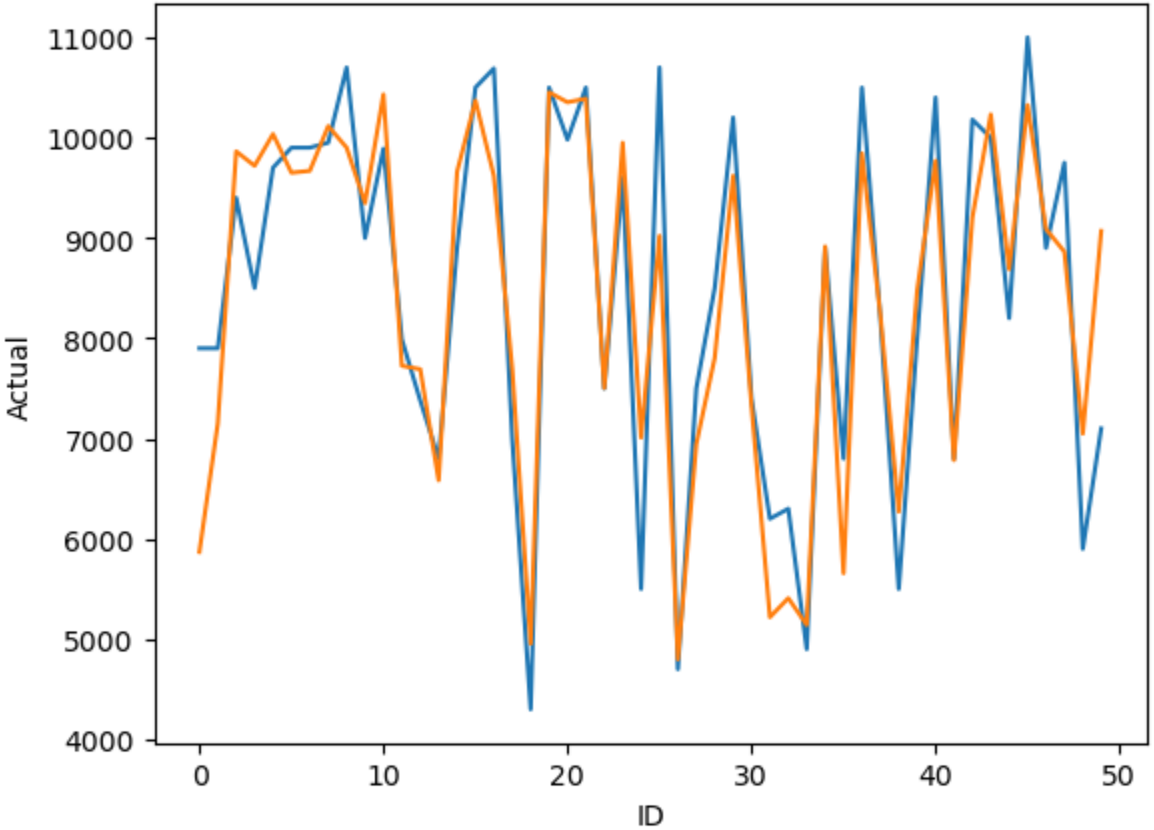
Out[52]:

	index	Actual	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

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

In [54]:
```
sns.lineplot(x='ID',y='Actual',data=Results.head(50))
sns.lineplot(x='ID',y='Predicted',data=Results.head(50))
plt.plot()
```

Out[54]: []



elastic net

In [55]:
```
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[55]:
```
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 [56]:
```
elastic_regressor.best_params_
```

Out[56]:
```
{'alpha': 0.01}
```

In [57]:
```
elastic=ElasticNet(alpha=0.01)
elastic.fit(x_train,y_train)
y_pred_elastic=elastic.predict(x_test)
```

In [58]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_elastic)
```

Out[58]:
```
0.841688021120299
```

In [59]:
```
from sklearn.metrics import mean_squared_error
elastic_Error=mean_squared_error(y_pred_elastic,y_test)
elastic_Error
```

Out[59]:
```
581390.7642825295
```

In [60]:
```
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)
```

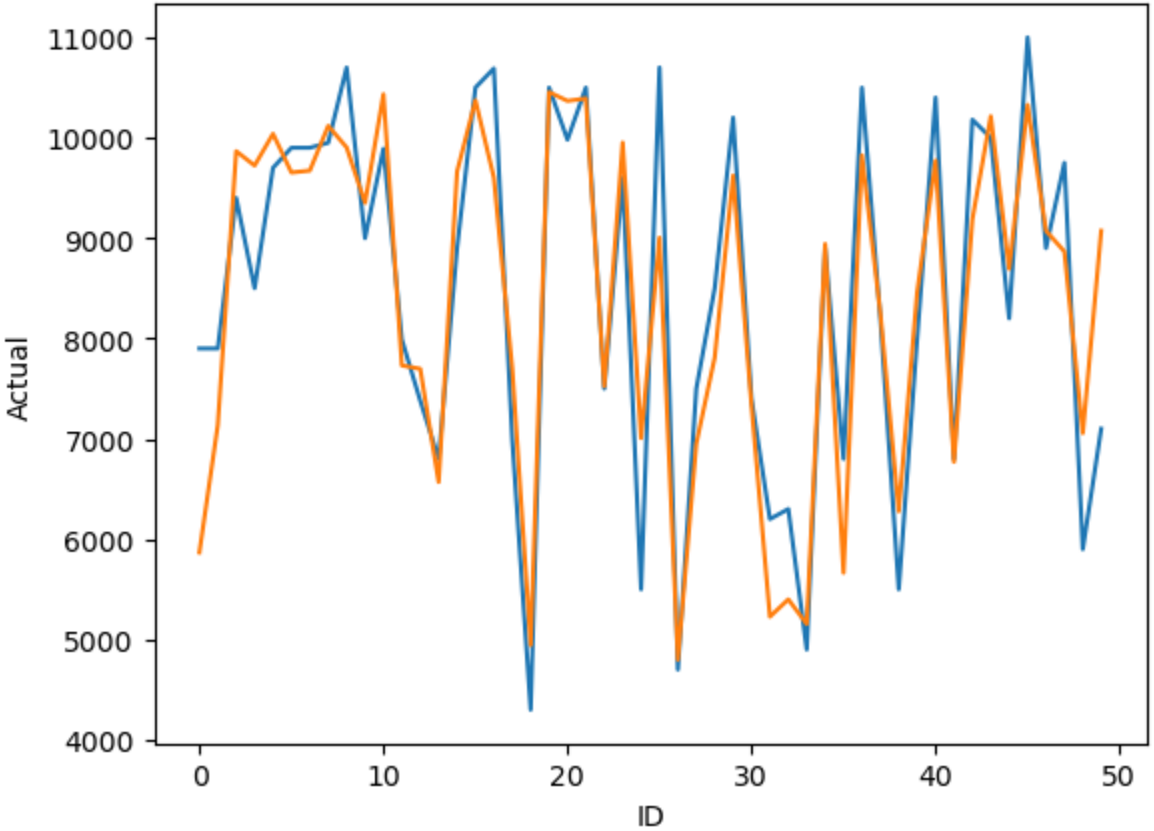
Out[60]:

	index	Actual	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

In [61]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [62]:
```
sns.lineplot(x='ID',y='Actual',data=Results.head(50))
sns.lineplot(x='ID',y='Predicted',data=Results.head(50))
plt.plot()
```

Out[62]: []



In [ ]:
```
 
```