In [36]:
```
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

In [37]:
```
data=pd.read_csv("/home/placement/Downloads/fiat500.csv")
```

In [38]:
```
data.describe()
```

Out[38]:

| | 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 [39]:
```
data.head()
```

Out[39]:

| | 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 [40]:
```
data1=data.drop(['lat','lon','ID'],axis=1)
data1
```

Out[40]:

| | 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 [41]:
```
data2=pd.get_dummies(data1)
```

In [42]:
```
data2
```

Out[42]:

| | 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 [43]:
```
data2.shape
```

Out[43]:
```
(1538, 8)
```

In [44]:
```
y=data2['price']
x=data2.drop('price',axis=1)
```

In [45]:
```
y
```

Out[45]:
```
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 [46]:
```
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 [47]:
```
x_test.head(5)
```

Out[47]:

| | 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 [48]:
```
x_train.head(5)
```

Out[48]:

| | 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 [49]:
```
y_test.head(5)
```

Out[49]:
```
481     7900
76      7900
1502    9400
669     8500
1409    9700
Name: price, dtype: int64
```

In [50]:
```
y_train.head(5)
```

Out[50]:
```
527    9990
129    9500
602    7590
331    8750
323    9100
Name: price, dtype: int64
```

In [51]:
```
#for linear regrestion
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 and the is created by training d
```

Out[51]:
```
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 [52]:
```
#prediction price
y_pred=reg.predict(x_test)
```

In [53]:
```
y_pred
```

Out[53]:
```
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 [54]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred)#y_pred=actual price,y_pred=predict price
```

Out[54]:
```
0.8415526986865394
```

In [55]:
```
from sklearn.metrics import mean_squared_error#calculating MSE
mean_squared_error(y_pred,y_test)
```

Out[55]:
```
581887.727391353
```

In [56]:
```
#Results=pd.DataFrame(columns=['Actual','predicted'])
#Result['Actual']=y_test
Results=pd.DataFrame(columns=['price','predicted'])
Results['price']=y_test
Results['predicted']=y_pred
Results=Results.reset_index()
Results['ID']=Results.index
Results.head(10)
```

Out[56]:

| | 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 |

# linear regrestion ends

In [57]:
```
#for Ridge Regression
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[57]:
```
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 [58]:
```
ridge_regressor.best_params_
```

Out[58]:
```
{'alpha': 30}
```

In [59]:
```
ridge=Ridge(alpha=30)
ridge.fit(x_train,y_train)
y_pred_ridge=ridge.predict(x_test)
```

In [60]:
```
from sklearn.metrics import mean_squared_error
Ridge_Error=mean_squared_error(y_pred_ridge,y_test)
Ridge_Error
```

Out[60]:
```
579521.7970897449
```

In [61]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_ridge)
```

Out[61]:
```
0.8421969385523054
```

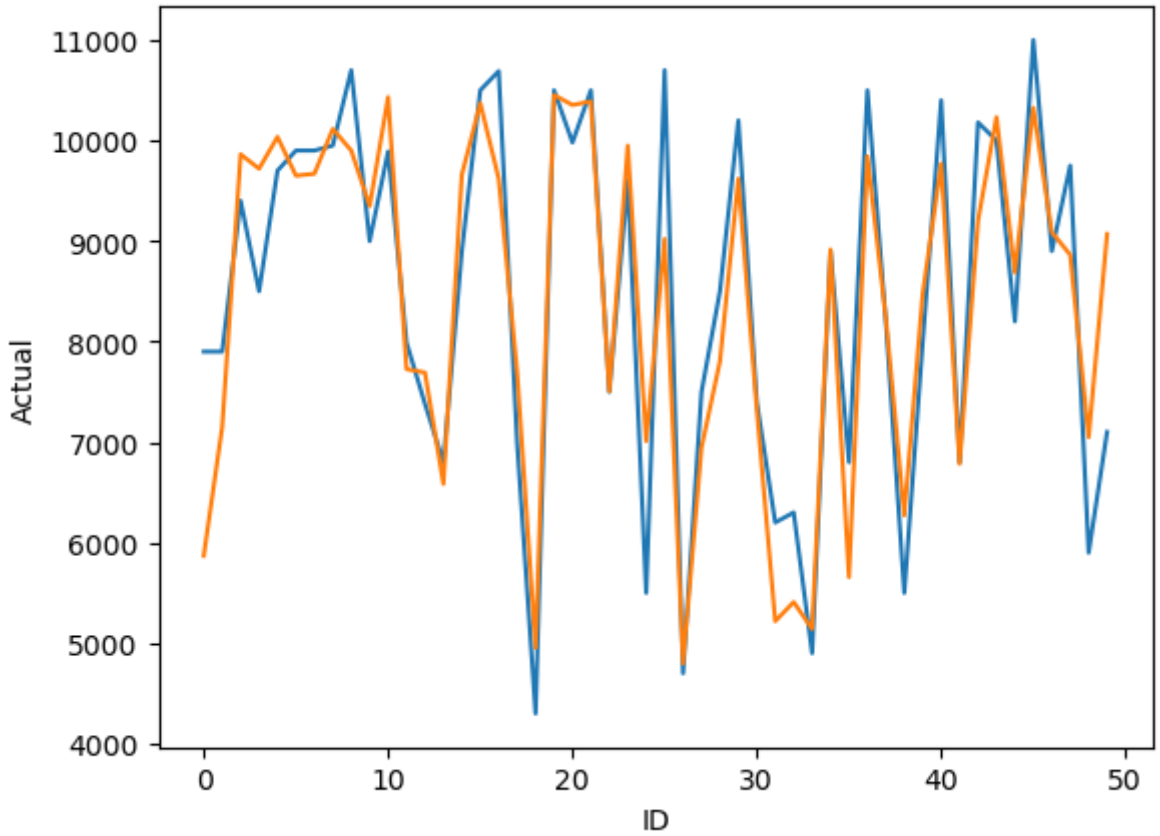
In [62]:
```
Results=pd.DataFrame(columns=['Actual','predicted'])
Results['Actual']=y_test
#Results=pd.DataFrame(columns=['price','predicted'])
#Results['price']=y_test
Results['predicted']=y_pred_ridge
#Results['km']=x_test['km']
Results=Results.reset_index()
Results['ID']=Results.index
Results.head(10)
```

Out[62]:

| | 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 [63]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.lineplot(x='ID',y='Actual',data=Results.head(50))#blue
sns.lineplot(x='ID',y='predicted',data=Results.head(50))#orange
plt.plot()
```

Out[63]: []



# Ridge regression ends

In [64]:
```
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[64]:
```
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 [65]:
```
elastic_regressor.best_params_
```

Out[65]:
```
{'alpha': 0.01}
```

In [66]:
```
elastic=ElasticNet(alpha=0.01)
elastic.fit(x_train,y_train)
y_pred_elastic=elastic.predict(x_test)
```

In [67]:
```
from sklearn.metrics import mean_squared_error
Elastic_Error=mean_squared_error(y_pred_elastic,y_test)
Elastic_Error
```

Out[67]:
```
581390.7642825295
```

In [68]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_elastic)
```

Out[68]:
```
0.841688021120299
```

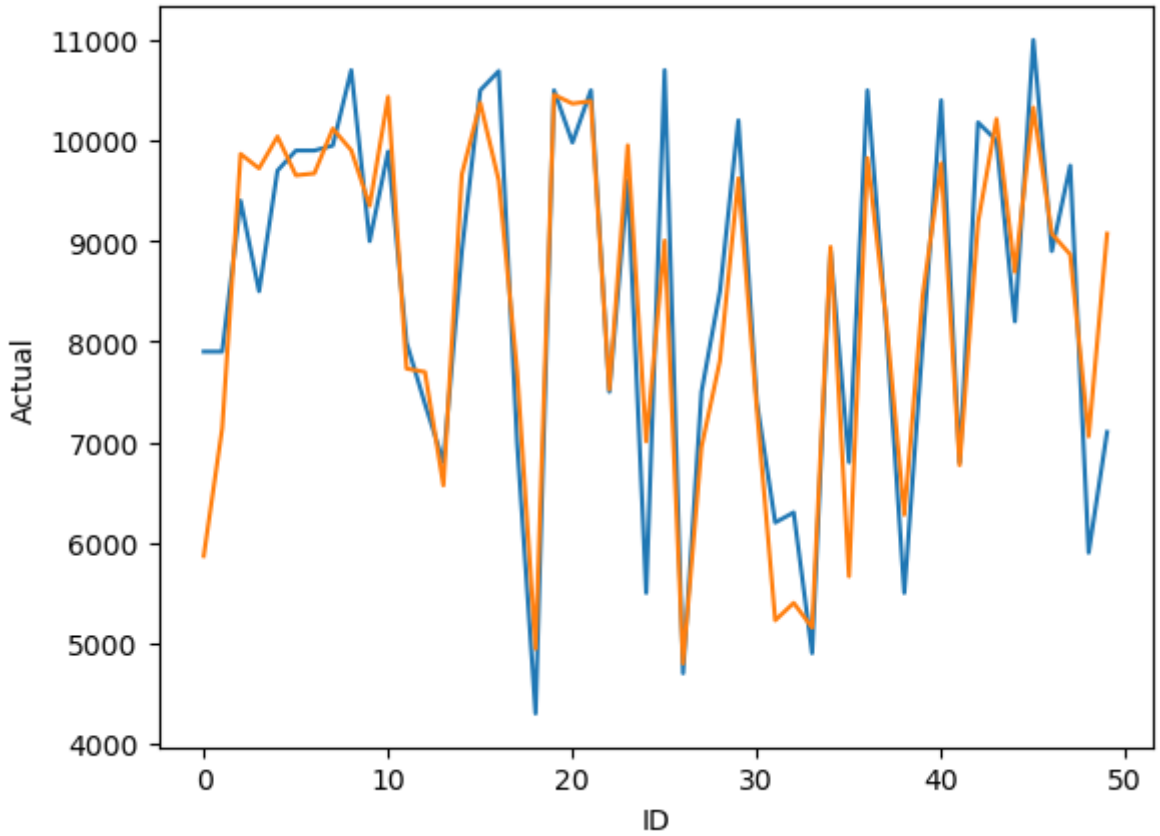
In [69]:
```
Results=pd.DataFrame(columns=['Actual','predicted'])
Results['Actual']=y_test
#Results=pd.DataFrame(columns=['price','predicted'])
#Results['price']=y_test
Results['predicted']=y_pred_elastic
#Results['km']=x_test['km']
Results=Results.reset_index()
Results['ID']=Results.index
Results.head(10)
```

Out[69]:

| | 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 [70]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.lineplot(x='ID',y='Actual',data=Results.head(50))#blue
sns.lineplot(x='ID',y='predicted',data=Results.head(50))#orange
plt.plot()
```

Out[70]: []



# Elastic Net ends

In [ ]: