In [1]:

```
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
data=pd.read_csv("/home/placement/Downloads/sid.csv")
```

In [3]:

```
data.describe()
```

Out[3]:

	ID	engine_power	age_in_days	km	previous_owners	lat
count	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
std	444.126671	3.988023	1289.522278	40046.830723	0.416423	2.133518
min	1.000000	51.000000	366.000000	1232.000000	1.000000	36.855839
25%	385.250000	51.000000	670.000000	20006.250000	1.000000	41.802990
50%	769.500000	51.000000	1035.000000	39031.000000	1.000000	44.394096
75%	1153.750000	51.000000	2616.000000	79667.750000	1.000000	45.467960
max	1538.000000	77.000000	4658.000000	235000.000000	4.000000	46.795612

In [4]:

```
list(data)
```

Out[4]:

```
['ID',
 'model',
 'engine_power',
 'age_in_days',
 'km',
 'previous_owners',
 'lat',
 'lon',
 'price']
```

In [5]:

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

In [6]:

```
data
```

Out[6]:

	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 [7]:

```
data1=data.loc[(data.model=='lounge')]
data1
```

Out[7]:

	model	engine_power	age_in_days	km	previous_owners	price
0	lounge	51	882	25000	1	8900
3	lounge	51	2739	160000	1	6000
6	lounge	51	731	11600	1	10750
7	lounge	51	1521	49076	1	9190
11	lounge	51	366	17500	1	10990
...	...	...	...	...	...	...
1528	lounge	51	2861	126000	1	5500
1529	lounge	51	731	22551	1	9900
1530	lounge	51	670	29000	1	10800
1534	lounge	74	3835	112000	1	4600
1536	lounge	51	2557	80750	1	5990

1094 rows × 6 columns

In [ ]:

```

```

In [8]:

```
data=pd.get_dummies(data)
```

In [9]:

```
data.shape
```

Out[9]:

```
(1538, 8)
```

In [10]:

```
data.groupby(['previous_owners']).count()
```

Out[10]:

	engine_power	age_in_days	km	price	model_lounge	model_pop	model_s
previous_owners							
1	1389	1389	1389	1389	1389	1389	1
2	117	117	117	117	117	117	
3	23	23	23	23	23	23	
4	9	9	9	9	9	9	

In [11]:

```
y=data["price"]
```

In [12]:

```
x=data.drop('price',axis=1)
```

In [13]:

```
x
```

Out[13]:

	engine_power	age_in_days	km	previous_owners	model_lounge	model_pop	model_
0	51	882	25000	1	1	0	
1	51	1186	32500	1	0	1	
2	74	4658	142228	1	0	0	
3	51	2739	160000	1	1	0	
4	73	3074	106880	1	0	1	
...	...	...	...	...	...	...	
1533	51	3712	115280	1	0	0	
1534	74	3835	112000	1	1	0	
1535	51	2223	60457	1	0	1	
1536	51	2557	80750	1	1	0	
1537	51	1766	54276	1	0	1	

1538 rows × 7 columns

In [14]:

```
y
```

Out[14]:

```
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 [15]:

```
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 [16]:

```
x_train.head()
```

Out[16]:

	engine_power	age_in_days	km	previous_owners	model_lounge	model_pop	model_s
527	51	425	13111	1	1	0	
129	51	1127	21400	1	1	0	
602	51	2039	57039	1	0	1	
331	51	1155	40700	1	1	0	
323	51	425	16783	1	1	0	

In [17]:

```
data
```

Out[17]:

	engine_power	age_in_days	km	previous_owners	price	model_lounge	model_pop
0	51	882	25000	1	8900	1	0
1	51	1186	32500	1	8800	0	1
2	74	4658	142228	1	4200	0	0
3	51	2739	160000	1	6000	1	0
4	73	3074	106880	1	5700	0	1
...	...	...	...	...	...	...	...
1533	51	3712	115280	1	5200	0	0
1534	74	3835	112000	1	4600	1	0
1535	51	2223	60457	1	7500	0	1
1536	51	2557	80750	1	5990	1	0
1537	51	1766	54276	1	7900	0	1

1538 rows × 8 columns

In [18]:

```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(x_train,y_train)
```

Out[18]:

```
▾ LinearRegression
LinearRegression()
```

In [19]:

```
ypred=reg.predict(x_test)
```

In [20]:

```
ypred
```

```
7,
       10108.31707235,  7009.6597206 ,  9853.90699412,  7146.8741496
5,
        6417.69133992,  9996.97382441,  9781.18795953,  8515.8325527
7,
        8456.30006203,  6499.76668237,  7768.57829985,  6832.8640612
2,
        8347.96113362, 10439.02404036,  7356.43463051,  8562.5656205
3,
        9820.78555199, 10035.83571539,  7370.77198022,  9411.4589400
6,
       10352.85155564,  8045.21588007, 10446.80664758,  3736.2011886
8,
       10348.63930496, 10435.96627494,  6167.80169017, 10390.1131780
4,
        6527.69471073,  9116.4755691 , 10484.52829    ,  9335.6988985
5,
        6709.57413543,  3390.72353093, 10106.33753331,  9792.4673200
8,
        6239.49568346,  4996.26346266,  9044.38667681,  9868.0995944
```

In [21]:

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

Out[21]:

```
0.8415526986865394
```

In [22]:

```
from sklearn.metrics import mean_squared_error
mean_squared_error(ypred,y_test)
```

Out[22]:

```
581887.727391353
```

In [23]:

```
print(mean_squared_error(ypred,y_test)**(1/2))
```

```
762.8156575420782
```

In [24]:

```
ypred
```

```
        6417.69133992,  9996.97382441,  9781.18795953,  8515.8325527
7,
        8456.30006203,  6499.76668237,  7768.57829985,  6832.8640612
2,
        8347.96113362, 10439.02404036,  7356.43463051,  8562.5656205
3,
        9820.78555199, 10035.83571539,  7370.77198022,  9411.4589400
6,
       10352.85155564,  8045.21588007, 10446.80664758,  3736.2011886
8,
       10348.63930496, 10435.96627494,  6167.80169017, 10390.1131780
4,
        6527.69471073,  9116.4755691 , 10484.52829   ,  9335.6988985
5,
        6709.57413543,  3390.72353093, 10106.33753331,  9792.4673200
8,
        6239.49568346,  4996.26346266,  9044.38667681,  9868.0995944
8,
        5484.13199252,  5698.5954821 , 10086.86206874,  8115.8169347
9,
```

In [25]:

```
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[25]:

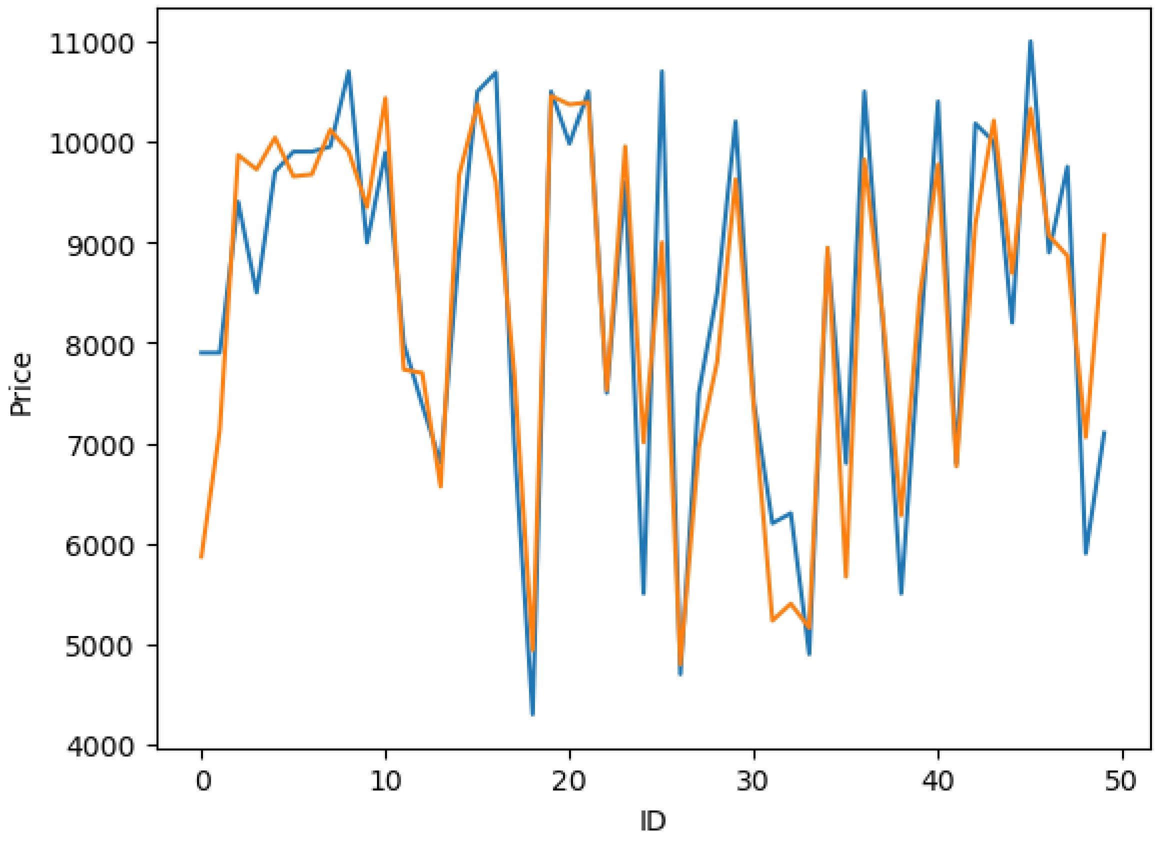
	index	Price	Predicted	ID
0	481	7900	5867.650338	0
1	76	7900	7133.701423	1
2	1502	9400	9866.357762	2
3	669	8500	9723.288745	3
4	1409	9700	10039.591012	4
5	1414	9900	9654.075826	5
6	1089	9900	9673.145630	6
7	1507	9950	10118.707281	7
8	970	10700	9903.859527	8
9	1198	8999	9351.558284	9
10	1088	9890	10434.349636	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 [26]:

```
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[26]:

```
<Axes: xlabel='ID', ylabel='Price'>
```



In [27]:

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

In [ ]:

In [ ]:

In [28]:

```
results
```

Out[28]:

	index	Price	Predicted	ID	dif
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 [31]:

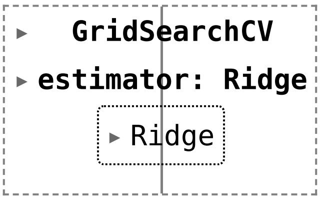
```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Ridge

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[31]:



▸ GridSearchCV

▸ estimator: Ridge

▸ Ridge

In [ ]:

```
ridge_regressor.best_params_
```

In [ ]:

```
ridge=Ridge(alpha=30)
ridge.fit(x_train,y_train)
y_pred_ridge=ridge.predict(x_test)
```

In [ ]:

```
Ridge_Error=mean_squared_error(y_pred_ridge,y_test)
Ridge_Error
```

In [ ]:

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

In [ ]:

```
res=pd.DataFrame(columns=['actual','predicted'])
res['actual']=y_test
res['predicted']=y_pred_ridge
res=res.reset_index()
res['ID']=res.index
```

In [ ]:

In [ ]:

```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [ ]:

In [ ]:

In [ ]:

```
res
```

In [ ]:

```
sns.lineplot(x='ID',y='actual',data=res.head(50))
sns.lineplot(x='ID',y='predicted',data=res.head(50))
plt.plot()
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: