In [1]:
```
import pandas as pd
```

In [2]:
```
import warnings
warnings.filterwarnings("ignore")
```

In [3]:
```
data=pd.read_csv("/home/placement/Desktop/EEE(222)/fiat500.csv")
```

In [4]:
```
data.describe()
```

Out[4]:
```
                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 [5]:
```
data=data.drop(['ID','lat','lon'],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]:
```
data=pd.get_dummies(data)
```

In [8]:
```
data.shape
```

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

In [9]:
```
data
```

Out[9]:

	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 [10]:
```
y=data['km']
x=data.drop('km',axis=1)
```

In [11]:
```
y
```

Out[11]:
```
0        25000
1        32500
2       142228
3       160000
4       106880
         ...  
1533    115280
1534    112000
1535     60457
1536     80750
1537     54276
Name: km, Length: 1538, dtype: int64
```

In [12]:
```
x
```

Out[12]:

	engine_power	age_in_days	previous_owners	price	model_lounge	model_pop	model_sport
0	51	882	1	8900	1	0	0
1	51	1186	1	8800	0	1	0
2	74	4658	1	4200	0	0	1
3	51	2739	1	6000	1	0	0
4	73	3074	1	5700	0	1	0
...	...	...	...	...	...	...	...
1533	51	3712	1	5200	0	0	1
1534	74	3835	1	4600	1	0	0
1535	51	2223	1	7500	0	1	0
1536	51	2557	1	5990	1	0	0
1537	51	1766	1	7900	0	1	0

1538 rows × 7 columns

In [13]:
```
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 [14]:
```
x_test.head(5)
```

Out[14]:

	engine_power	age_in_days	previous_owners	price	model_lounge	model_pop	model_sport
481	51	3197	2	7900	0	1	0
76	62	2101	1	7900	0	1	0
1502	51	670	1	9400	1	0	0
669	51	913	1	8500	1	0	0
1409	51	762	1	9700	1	0	0

In [15]:
```
y_train.head(10)
```

Out[15]:
```
527     13111
129     21400
602     57039
331     40700
323     16783
1358    29378
522     18443
584     11997
1236    66900
535     35000
Name: km, dtype: int64
```

linear regression

In [16]:
```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()#creating object of linearregression
reg.fit(x_train,y_train)#traning and fitting lr object using traning data
```

Out[16]:
```
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 [17]:
```
ypred=reg.predict(x_test)
```

In [18]:
```
ypred
```

Out[18]:
```
array([ 75731.24452341,  66029.84772251,  34401.2360973 ,  47789.21545234,
        31445.26338572,  28940.45224947,  29985.55310965,  27782.98986147,
        18389.94810051,  43647.33173542,  26400.6856128 ,  65315.51718461,
        78520.35323068,  79990.40592647,  40933.24832683,  17977.18600208,
        21520.86799789,  86390.77726774, 134696.69941688,  18247.17039096,
        28441.83272222,  17977.18600208,  84917.61592634,  31095.18096823,
       102601.42053493,  18683.76101173, 126929.83769068,  80736.28377642,
        61845.82202184,  25427.09241246,  75352.29888237, 107607.91154532,
       102849.71048838, 131742.45012204,  54385.26759139,  96392.07925672,
        17490.98815001,  72194.05616406, 102326.85312655,  58616.36626514,
        23192.26566508,  82637.99477227,  27133.01992404,  22420.51239391,
        61892.93067256,  12228.9994177 ,  44144.63447273,  33196.66505858,
       107132.46976595,  66916.67221792,  38015.67509214,  13863.75090251,
        18037.8995605 ,  83477.63824197,  40401.9887229 ,  30272.50416449,
        27657.65528172,  13593.76651363,  21795.11626488,  57818.16924686,
        21156.03445181,  92459.0954649 ,  25427.09241246,  83317.89386646,
        82265.53558981,  28670.46786059,  29579.53406254,  36591.80158707,
        36351.95918295, 103152.37732341,  82659.4275832 ,  96271.05349949,
        51009.76245379,  18247.17039096,  68027.75183387,  44136.37698061,
        44839.37871845,  51727.60934316,  93542.47699691,  43198.01229024,
```

In [19]:
```
from sklearn.metrics import r2_score
r2_score(y_test,ypred)
```

Out[19]:
```
0.7691178949173114
```

In [20]:

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

In [21]:

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

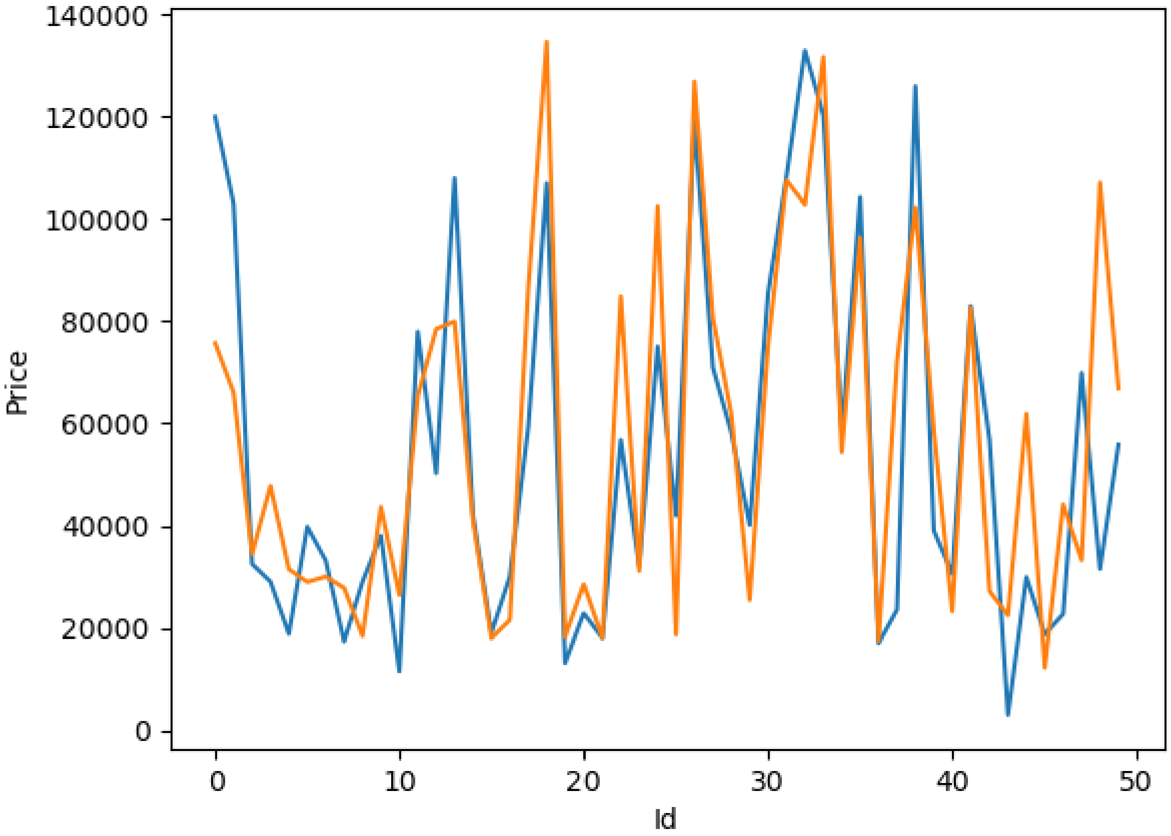
Out[21]:

	index	Price	Predicted	Id
0	481	120000	75731.244523	0
1	76	103000	66029.847723	1
2	1502	32473	34401.236097	2
3	669	29000	47789.215452	3
4	1409	18800	31445.263386	4
5	1414	39751	28940.452249	5
6	1089	33160	29985.553110	6
7	1507	17324	27782.989861	7
8	970	29000	18389.948101	8
9	1198	38000	43647.331735	9
10	1088	11511	26400.685613	10
11	576	78000	65315.517185	11
12	965	50247	78520.353231	12
13	1488	108000	79990.405926	13
14	1432	42095	40933.248327	14

In [22]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [23]:
```
sns.lineplot(x='Id',y='Price',data=Results.head(50))#blue
sns.lineplot(x='Id',y='Predicted',data=Results.head(50))#orange
plt.plot()
```

Out[23]:
```
[]
```



In [24]:
```
Results['DIFF']=Results.apply(lambda row: row.Price-row.Predicted,axis=1)
```

```
In [25]: Results
```

```
Out[25]:
       index   Price      Predicted    Id           DIFF
  0      481  120000   75731.244523     0   44268.755477
  1       76  103000   66029.847723     1   36970.152277
  2     1502   32473   34401.236097     2   -1928.236097
  3      669   29000   47789.215452     3  -18789.215452
  4     1409   18800   31445.263386     4  -12645.263386
...      ...     ...            ...   ...            ...
503      291   22000   15885.136964   503    6114.863036
504      596   85500  100920.913776   504  -15420.913776
505     1489   22148   27354.930930   505   -5206.930930
506     1436   61000   74399.449201   506  -13399.449201
507      575   19112   12967.563730   507    6144.436270

508 rows × 5 columns
```

ridge regression

In [26]:

```
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[26]:

```
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 [27]:

```
ridge_regressor.best_params_
```

Out[27]:

```
{'alpha': 30}
```

In [28]:

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

In [29]:

```
from sklearn.metrics import mean_squared_error
Ridge_Error=mean_squared_error(y_pred_ridge,y_test)
Ridge_Error
```

Out[29]:

```
346741563.1891953
```

In [30]:

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

Out[30]:

```
0.7691472836651715
```

In [31]:

```
Results=pd.DataFrame(columns=['Actual','predicted'])
Results['Actual']=y_test
Results['predicted']=y_pred_ridge
#Result['km']=x_test['km']
Results=Results.reset_index()
Results['Id']=Results.index
Results.head(10)
```

Out[31]:

	index	Actual	predicted	Id
0	481	120000	75610.216839	0
1	76	103000	66438.222154	1
2	1502	32473	34390.439217	2
3	669	29000	47783.153021	3
4	1409	18800	31444.515496	4
5	1414	39751	28942.281540	5
6	1089	33160	29995.447469	6
7	1507	17324	27781.363705	7
8	970	29000	18397.986372	8
9	1198	38000	43663.963925	9

In [32]:

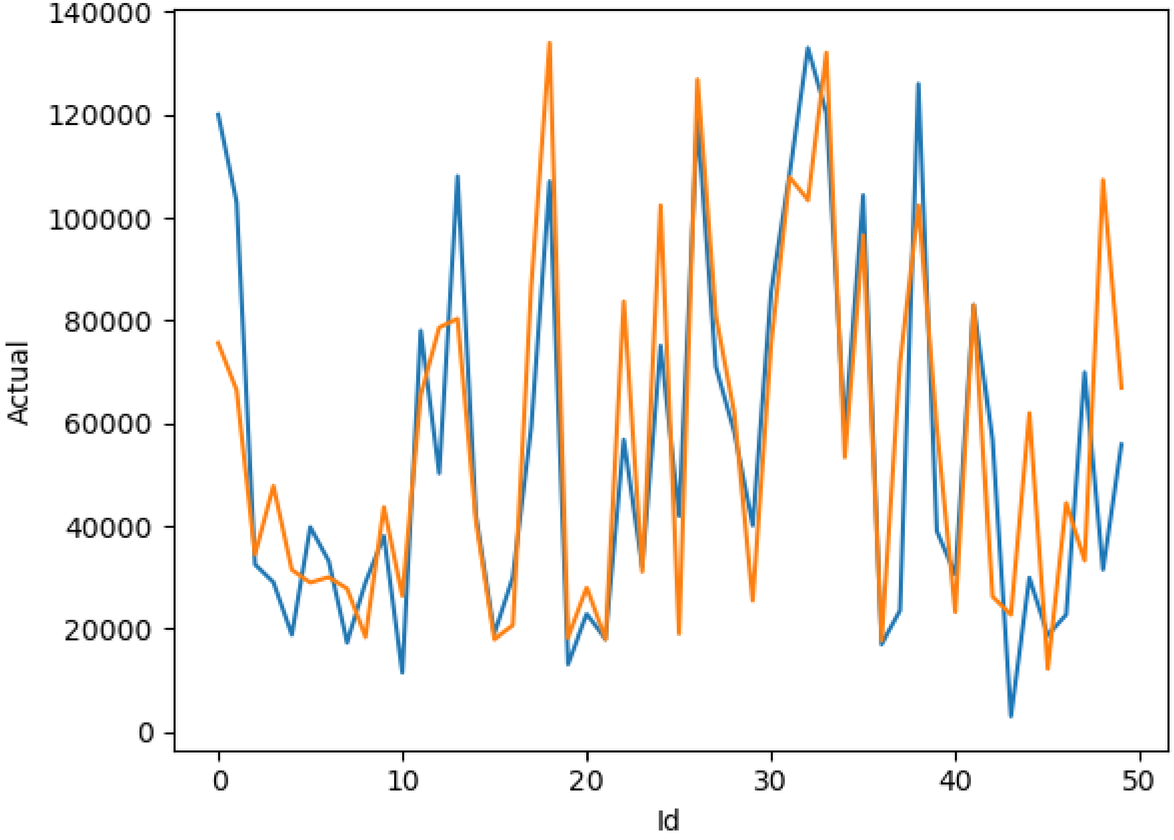
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [33]:

```
sns.lineplot(x='Id',y='Actual',data=Results.head(50))#bule
sns.lineplot(x='Id',y='predicted',data=Results.head(50))#orange
plt.plot()
```

Out[33]:

```
[]
```



elastic net

In [34]:

```
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[34]:

```
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 [35]:

```
elastic_regressor.best_params_
```

Out[35]:

```
{'alpha': 1}
```

In [36]:

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

In [37]:

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

Out[37]:

```
0.7691157601736113
```

In [38]:
```
from sklearn.metrics import mean_squared_error
elastic_Error=mean_squared_error(y_pred_elastic,y_test)
elastic_Error
```

Out[38]:
```
346788911.580474
```

In [39]:
```
Results=pd.DataFrame(columns=['Actual','predicted'])
Results['Actual']=y_test
Results['predicted']=y_pred_elastic
#Result['km']=x_test['km']
Results=Results.reset_index()
Results['price']=Results.index
Results.head(10)
```

Out[39]:

	index	Actual	predicted	price
0	481	120000	75710.809218	0
1	76	103000	66113.691091	1
2	1502	32473	34399.544383	2
3	669	29000	47789.104506	3
4	1409	18800	31445.472680	4
5	1414	39751	28941.068113	5
6	1089	33160	29987.853086	6
7	1507	17324	27782.851276	7
8	970	29000	18391.334150	8
9	1198	38000	43651.631475	9

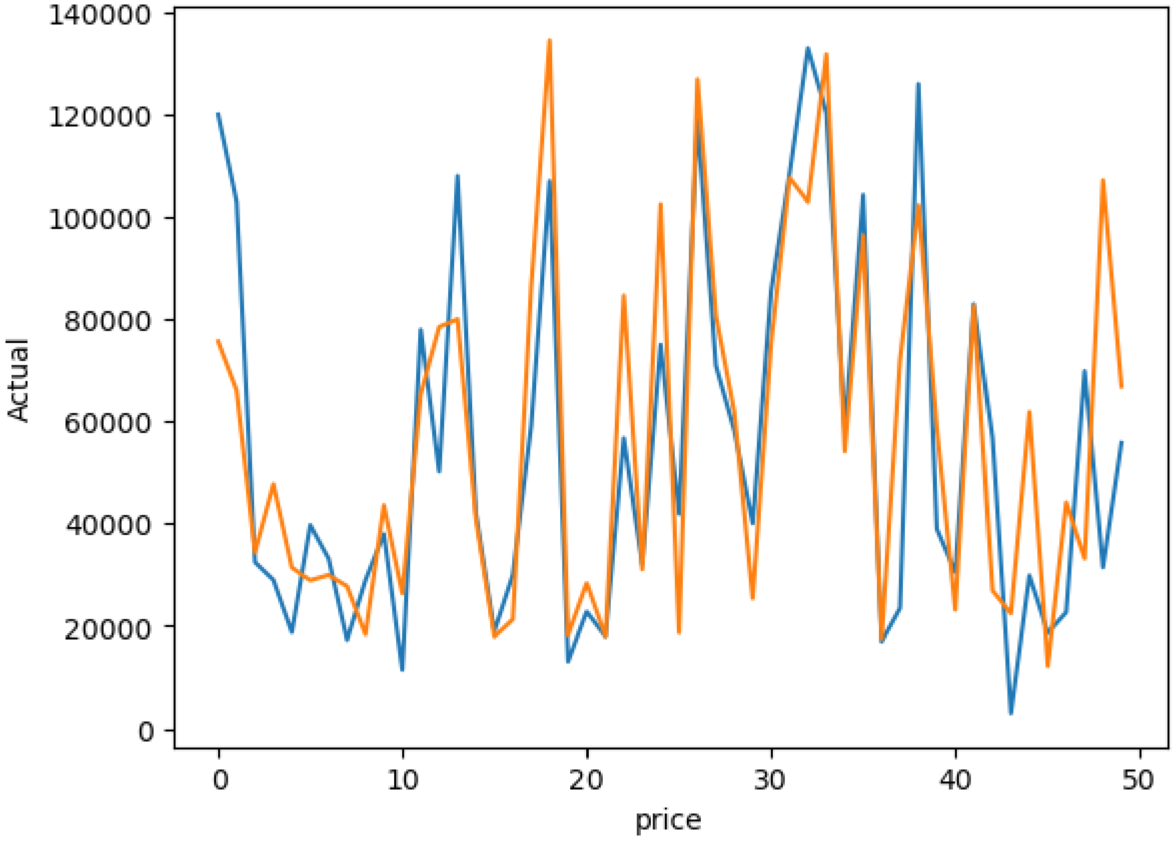
In [40]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [41]:

```
sns.lineplot(x='price',y='Actual',data=Results.head(50))#bule
sns.lineplot(x='price',y='predicted',data=Results.head(50))#orange
plt.plot()
```

Out[41]:

```
[]
```



In [ ]: