Ridge Model

In [140]:
```
import pandas as pd
import numpy as np
data=pd.read_csv("/home/placement/Downloads/fiat500 (2).csv")
data.describe()
```

Out[140]:

	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 [141]:
```
data=data.drop(['ID','lat','lon'],axis=1)
```

In [142]:
```
data['model']=data['model'].map({"lounge":1,"pop":2,"sport":3})
data
```

Out[142]:

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

1538 rows × 6 columns

In [143]:
```
data
```

Out[143]:

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

1538 rows × 6 columns

In [144]:
```
y=data['price']
x=data.drop("price",axis=1)
y
```

Out[144]:
```
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 [ ]:
```

```

In [145]:
```
x
```

Out[145]:

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

1538 rows × 5 columns

In [146]:
```
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 [147]:
```
x_test.head()
```

Out[147]:

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

In [148]:
```
x_train.head()
```

Out[148]:

	model	engine_power	age_in_days	km	previous_owners
527	1	51	425	13111	1
129	1	51	1127	21400	1
602	2	51	2039	57039	1
331	1	51	1155	40700	1
323	1	51	425	16783	1

In [149]:
```
y_test.head()
```

Out[149]:
```
481     7900
76      7900
1502    9400
669     8500
1409    9700
Name: price, dtype: int64
```

In [150]:
```
y_train.head()
```

Out[150]:
```
527    9990
129    9500
602    7590
331    8750
323    9100
Name: price, dtype: int64
```

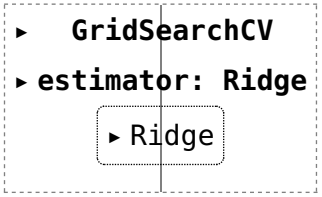
In [ ]:
```

```

importing GridsearchCv

In [151]:
```
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[151]:



In [152]:
```
ridge_regressor.best_params_
```

Out[152]:
```
{'alpha': 30}
```

In [153]:
```
ridge=Ridge(alpha=30)
ridge.fit(x_train,y_train)
y_pred_ridge=ridge.predict(x_test)
```

In [154]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_ridge)
```

Out[154]:
```
0.8391885506165899
```

In [155]:
```
from sklearn.metrics import mean_squared_error#mean_squared error
Ridge_Error=mean_squared_error(y_pred_ridge,y_test)
kk=Ridge_Error
kk
```

Out[155]:
```
590569.9121697355
```

In [156]:
```
import math as m
k=m.sqrt(kk)
print(k)
```

```
768.4854664661756
```

For launge model

In [157]:
```
data=data.loc[data.model==1]
data
```

Out[157]:

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

1094 rows × 6 columns

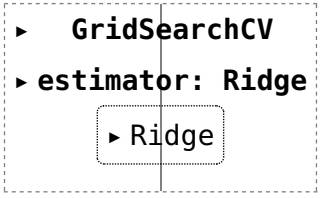
In [158]:
```
y=data['price']
x=data.drop("price",axis=1)
y
```

Out[158]:
```
0        8900
3        6000
6       10750
7        9190
11      10990
        ...
1528     5500
1529     9900
1530    10800
1534     4600
1536     5990
Name: price, Length: 1094, dtype: int64
```

In [159]:
```
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 [166]:
```
import warnings
warnings.filterwarnings("ignore")
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[166]:



In [161]:
```
ridge_regressor.best_params_
```

Out[161]:
```
{'alpha': 30}
```

In [162]:
```
ridge=Ridge(alpha=30)
ridge.fit(x_train,y_train)
y_pred_ridge=ridge.predict(x_test)
```

In [163]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_ridge)
```

Out[163]:
```
0.8373030813683995
```

In [164]:
```
from sklearn.metrics import mean_squared_error#mean_squared error
Ridge_Error=mean_squared_error(y_pred_ridge,y_test)
kk=Ridge_Error
kk
```

Out[164]:
```
519771.8129989742
```

In [176]:
```
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(10)
```

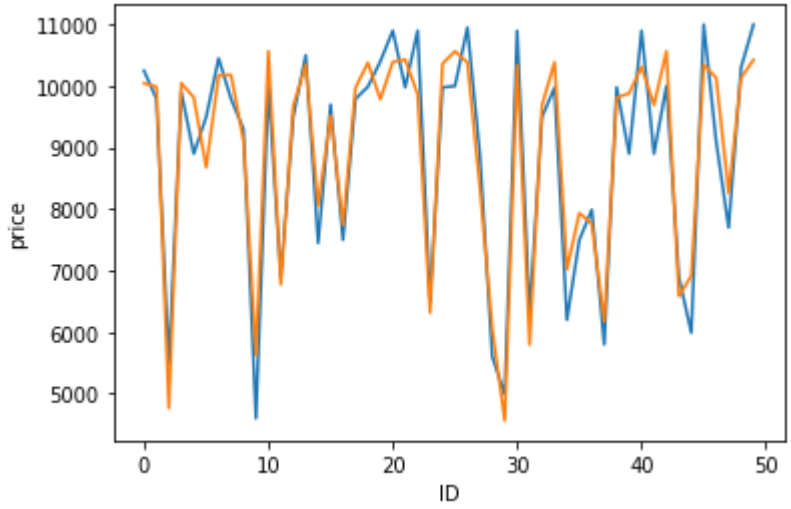
Out[176]:

	index	price	predicted	ID
0	676	10250	10045.347779	0
1	215	9790	9989.171535	1
2	146	5500	4769.099603	2
3	1319	9900	10048.683238	3
4	1041	8900	9813.944798	4
5	1425	9500	8678.143561	5
6	409	10450	10173.797921	6
7	617	9790	10180.627008	7
8	1526	9300	9107.315259	8
9	1010	4600	5625.007407	9

plots b/w actual&predicted price

In [179]:
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.lineplot(x="ID",y="price",data=results.head(50))
sns.lineplot(x="ID",y="predicted",data=results.head(50))
```

Out[179]: <Axes: xlabel='ID', ylabel='price'>



In [ ]: