In [74]:
```
import pandas as pd
```

In [75]:
```
data=pd.read_csv('fiat500.csv')
```

In [76]:
```
data.head()
```

Out[76]:

	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 [77]:
```
list(data)
```

Out[77]:
```
['ID',
 'model',
 'engine_power',
 'age_in_days',
 'km',
 'previous_owners',
 'lat',
 'lon',
 'price']
```

In [78]:
```
data2=data.loc[(data.previous_owners)==1]
```

In [79]:
```
data2
```

Out[79]:

	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
...	...	...	...	...	...	...	...	...	...
1533	1534	sport	51	3712	115280	1	45.069679	7.704920	5200
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870	4600
1535	1536	pop	51	2223	60457	1	45.481541	9.413480	7500
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270	5990
1537	1538	pop	51	1766	54276	1	40.323410	17.568270	7900

1389 rows × 9 columns

In [80]:
```
data3=data.drop(['ID','lat','lon'],axis=1)
```

In [81]:
```
data3
```
Out[81]:

	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 [82]:
```
data4=pd.get_dummies(data3)
```

In [83]:
```
data4
```

Out[83]:

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

1538 rows × 8 columns

In [84]:
```
y=data4['price']
x=data4.drop(['price'],axis=1)
```

In [85]:
```
y
```

Out[85]:
```
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 [86]:
```
x
```

Out[86]:

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

1538 rows × 7 columns

In [87]:
```
#pip install scikit-learn
```

In [88]:
```
from sklearn.model_selection import train_test_split
```

In [89]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.33,random_state=42)
```

In [90]:
```
x_test.head(5)
```

Out[90]:

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

In [91]:
```
x_train.shape
```

Out[91]:
```
(1030, 7)
```

In [92]:
```
import warnings
warnings.filterwarnings("ignore")
```

In [93]:
```
from sklearn.model_selection import GridSearchCV
```

In [94]:
```
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[94]:
```
▸      GridSearchCV
▸ estimator: ElasticNet
      ▸ ElasticNet
```

In [95]:
```
elastic_regressor.best_params_
```

Out[95]:
```
{'alpha': 0.01}
```

In [96]:
```
elastic=ElasticNet(alpha=0.1)
elastic.fit(x_train,y_train)
y_pred_elastic=elastic.predict(x_test)
```

In [97]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_elastic)
```

Out[97]:
```
0.8425222843073694
```

In [103]:
```
from sklearn.metrics import mean_squared_error
elastic_Error=mean_squared_error(y_pred_elastic,y_test)
elastic_Error
```

Out[103]:
```
578326.9853103001
```

In [104]:
```
import math
```

```
In [105]: math.sqrt(elastic_Error)
```

```
Out[105]: 760.4781294095841
```

```
In [ ]:
```