In [49]:
```
import pandas as r
d=r.read_csv("/home/placement/Downloads/fiat500") #reading the file into the jupyter
d.describe()
```

Out[49]:

	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 [50]:
```
#Creating a new dataframe using loc[]
d1=d.loc[(d.model=='lounge')]
d1
```

Out[50]:

	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
3	4	lounge	51	2739	160000	1	40.633171	17.634609	6000
6	7	lounge	51	731	11600	1	44.907242	8.611560	10750
7	8	lounge	51	1521	49076	1	41.903221	12.495650	9190
11	12	lounge	51	366	17500	1	45.069679	7.704920	10990
...	...	...	...	...	...	...	...	...	...
1528	1529	lounge	51	2861	126000	1	43.841980	10.515310	5500
1529	1530	lounge	51	731	22551	1	38.122070	13.361120	9900
1530	1531	lounge	51	670	29000	1	45.764648	8.994500	10800
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870	4600
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270	5990

1094 rows × 9 columns

In [51]:
```
#Removing the columns
d1=d1.drop(['lat','ID','lon'],axis=1)
d1
```

Out[51]:

	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 [52]:
```
#Converting the strings in the data frame into integers
d1=r.get_dummies(d1)
d1.describe()
```

Out[52]:

	engine_power	age_in_days	km	previous_owners	price	model_lounge
count	1094.000000	1094.000000	1094.000000	1094.000000	1094.000000	1094.0
mean	51.565814	1437.570384	46873.445155	1.117002	8949.486289	1.0
std	3.181137	1203.670113	37051.887997	0.406966	1798.159691	0.0
min	51.000000	366.000000	1232.000000	1.000000	2900.000000	1.0
25%	51.000000	640.000000	18826.500000	1.000000	7800.000000	1.0
50%	51.000000	790.000000	33105.500000	1.000000	9500.000000	1.0
75%	51.000000	2192.000000	67000.000000	1.000000	10400.000000	1.0
max	77.000000	4658.000000	235000.000000	4.000000	11100.000000	1.0

In [53]:
```
#Counting the rows and columns
d1.shape
```

Out[53]: (1094, 6)

In [54]:
```
y=d1['price']
x=d1.drop('price',axis=1)
x
```

Out[54]:

	engine_power	age_in_days	km	previous_owners	model_lounge
0	51	882	25000	1	1
3	51	2739	160000	1	1
6	51	731	11600	1	1
7	51	1521	49076	1	1
11	51	366	17500	1	1
...	...	...	...	...	...
1528	51	2861	126000	1	1
1529	51	731	22551	1	1
1530	51	670	29000	1	1
1534	74	3835	112000	1	1
1536	51	2557	80750	1	1

1094 rows × 5 columns

In [55]:
```
#spliting data to create the model
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 [56]:
```
#Ridge regression model
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[56]:
```
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 [57]:
```
ridge_regressor.best_params_
```

Out[57]:
```
{'alpha': 30}
```

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]:
```
519771.8129989745
```

In [61]:
```
from sklearn.metrics import r2_score #to  check the efficiency
r2_score(y_test,y_pred_ridge)
```

Out[61]: 0.8373030813683994