In [119]:
```
import pandas as pd
```

In [120]:
```
data=pd.read_csv("/home/placement/Desktop/data analysis(413)/fiat500.csv")
```

In [121]:
```
data
```

Out[121]:

	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

1538 rows × 9 columns

In [122]:
```
z=data1.loc[(data1.previous_owners==1)]
```

In [123]:
```
z
```

Out[123]:

	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

1389 rows × 8 columns

In [124]:
```
data1=data.drop(["lat","lon","ID"],axis=1)
```

In [125]:
```
data1=pd.get_dummies(data1)
```

In [126]:
```
data1
```

Out[126]:

	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 [127]:
```
 y=z['price']
x=z.drop(['price'],axis=1)
```

In [128]:
```
#z.drop(['model'],axis=1)
```

In [129]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.1,random_state=42)
```

In [ ]:
```

```

In [130]:
```
from sklearn.linear_model import ElasticNet

from sklearn.model_selection import GridSearchCV

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)
```

```
e+05
  model = cd_fast.enet_coordinate_descent(
/home/placement/anaconda3/lib/python3.10/site-packages/sklearn/linear_model/_coordinate_descent.py:631: C
onvergenceWarning: Objective did not converge. You might want to increase the number of iterations, check
the scale of the features or consider increasing regularisation. Duality gap: 2.703e+08, tolerance: 3.517
e+05
  model = cd_fast.enet_coordinate_descent(
/home/placement/anaconda3/lib/python3.10/site-packages/sklearn/linear_model/_coordinate_descent.py:631: C
onvergenceWarning: Objective did not converge. You might want to increase the number of iterations, check
the scale of the features or consider increasing regularisation. Duality gap: 2.854e+08, tolerance: 3.711
e+05
  model = cd_fast.enet_coordinate_descent(
/home/placement/anaconda3/lib/python3.10/site-packages/sklearn/linear_model/_coordinate_descent.py:631: C
onvergenceWarning: Objective did not converge. You might want to increase the number of iterations, check
the scale of the features or consider increasing regularisation. Duality gap: 2.909e+08, tolerance: 3.576
e+05
  model = cd_fast.enet_coordinate_descent(
/home/placement/anaconda3/lib/python3.10/site-packages/sklearn/linear_model/_coordinate_descent.py:631: C
onvergenceWarning: Objective did not converge. You might want to increase the number of iterations, check
```

In [131]:
```
elastic=ElasticNet(alpha=.01)
elastic.fit(x_train,y_train)
y_pred_elastic=elastic.predict(x_test)
```

In [132]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_elastic)
```

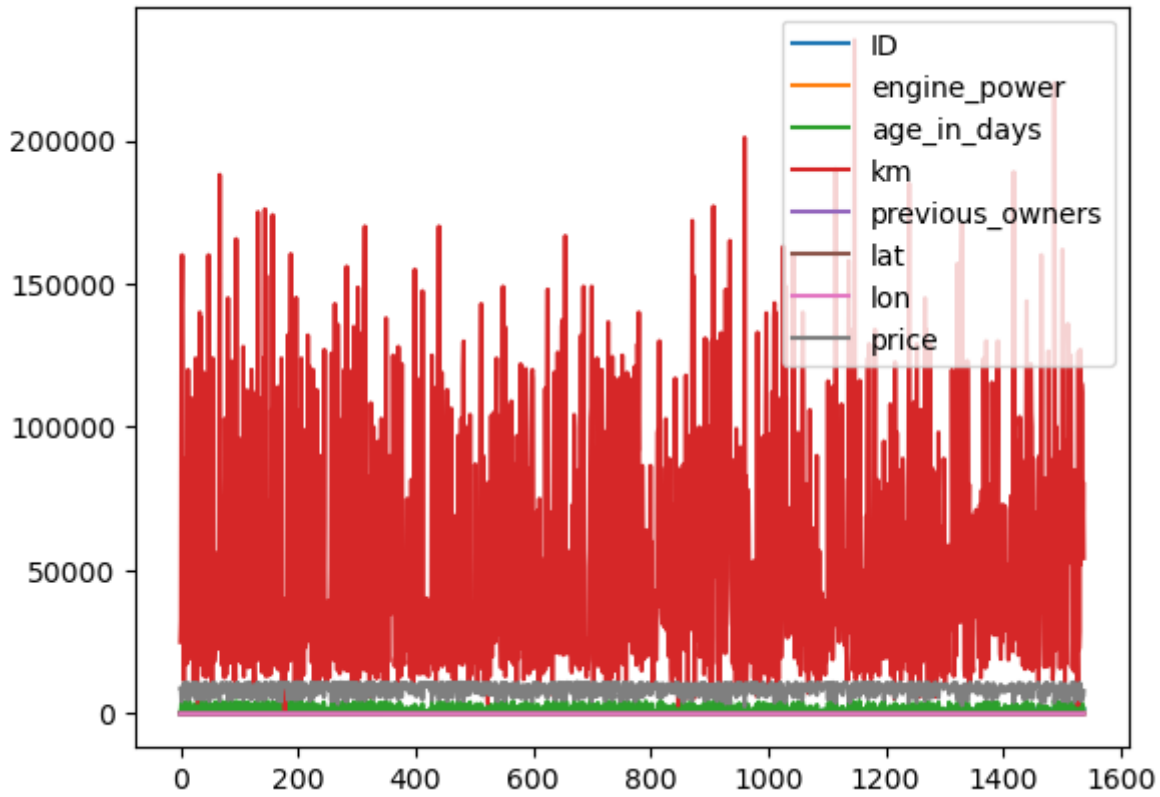
Out[132]: 0.8488682857174344

In [134]:
```
from sklearn.metrics import mean_squared_error
elastic_Error=mean_squared_error(y_pred_elastic,y_test)
elastic_Error
```

Out[134]: 603966.023413073

In [135]:
```
data.plot()
```

Out[135]: <Axes: >



In [136]:
```
elastic_regressor.best_params_
```

Out[136]:
```
{'alpha': 0.01}
```

In [ ]:

In [ ]: