In [ ]:
```python
import pandas as pd
data=pd.read_csv("C:\\Users\\rama\\Downloads\\fiat500.csv")
data
```

In [129…
```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1538 entries, 0 to 1537
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ID               1537 non-null   float64
 1   model            1538 non-null   object 
 2   engine_power     1538 non-null   int64  
 3   age_in_days      1538 non-null   int64  
 4   km               1538 non-null   int64  
 5   previous_owners  1538 non-null   int64  
 6   lat              1538 non-null   float64
 7   lon              1538 non-null   float64
 8   price            1538 non-null   int64  
dtypes: float64(3), int64(5), object(1)
memory usage: 108.3+ KB
```

In [130…
```python
data.head(94)
```

Out[130]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	NaN	lounge	51	882	25000	1	44.907242	8.611560	8900
1	2.0	pop	51	1186	32500	1	45.666359	12.241890	8800
2	3.0	sport	74	4658	142228	1	45.503300	11.417840	4200
3	4.0	lounge	51	2739	160000	1	40.633171	17.634609	6000
4	5.0	pop	73	3074	106880	1	41.903221	12.495650	5700
...	...	...	...	...	...	...	...	...	...
89	90.0	lounge	51	397	17912	1	41.832649	12.752600	9970
90	91.0	lounge	51	1066	69916	1	41.802990	12.598930	8980
91	92.0	lounge	51	2647	97700	1	44.508839	11.469080	7450
92	93.0	pop	51	366	11500	1	45.707249	11.477600	9500
93	94.0	lounge	51	397	17250	1	45.467960	9.181780	10050

94 rows × 9 columns

In [131…
```python
data['km'].corr(data['price'])
```

Out[131]: -0.8593732823337393

In [132…
```python
data['km'].cov(data['price'])
```

Out[132]: -66764018.77363063

In [133…
```python
data.describe()
```

Out[133]:

	ID	engine_power	age_in_days	km	previous_owners	lat	lon	price
count	1537.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000
mean	770.000000	51.904421	1650.980494	53396.011704	1.123537	43.541361	11.563428	8576.003901
std	443.837996	3.988023	1289.522278	40046.830723	0.416423	2.133518	2.328190	1939.958641
min	2.000000	51.000000	366.000000	1232.000000	1.000000	36.855839	7.245400	2500.000000
25%	386.000000	51.000000	670.000000	20006.250000	1.000000	41.802990	9.505090	7122.500000
50%	770.000000	51.000000	1035.000000	39031.000000	1.000000	44.394096	11.869260	9000.000000
75%	1154.000000	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 [134…
```python
data['engine_power'].mean()
```

Out[134]: 51.904421326397916

In [135…
```python
data['engine_power'].mode()
```

Out[135]:
```
0    51
Name: engine_power, dtype: int64
```

In [136…
```python
data['model'].unique()
```

Out[136]: array(['lounge', 'pop', 'sport'], dtype=object)

In [137…
```python
data.shape
```

Out[137]: (1538, 9)

In [138…
```python
data['previous_owners'].unique()
```

Out[138]: array([1, 2, 3, 4], dtype=int64)

In [139…
```python
data.groupby(data['previous_owners']).count()
```

Out[139]:

previous_owners	ID	model	engine_power	age_in_days	km	lat	lon	price
1	1388	1389	1389	1389	1389	1389	1389	1389
2	117	117	117	117	117	117	117	117
3	23	23	23	23	23	23	23	23
4	9	9	9	9	9	9	9	9

In [140…
```python
data.groupby(data['model']).count()
```

Out[140]:

model	ID	engine_power	age_in_days	km	previous_owners	lat	lon	price
lounge	1093	1094	1094	1094	1094	1094	1094	1094
pop	358	358	358	358	358	358	358	358
sport	86	86	86	86	86	86	86	86

In [141…
```python
data
```

Out[141]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	NaN	lounge	51	882	25000	1	44.907242	8.611560	8900
1	2.0	pop	51	1186	32500	1	45.666359	12.241890	8800
2	3.0	sport	74	4658	142228	1	45.503300	11.417840	4200
3	4.0	lounge	51	2739	160000	1	40.633171	17.634609	6000
4	5.0	pop	73	3074	106880	1	41.903221	12.495650	5700
...	...	...	...	...	...	...	...	...	...
1533	1534.0	sport	51	3712	115280	1	45.069679	7.704920	5200
1534	1535.0	lounge	74	3835	112000	1	45.845692	8.666870	4600
1535	1536.0	pop	51	2223	60457	1	45.481541	9.413480	7500
1536	1537.0	lounge	51	2557	80750	1	45.000702	7.682270	5990
1537	1538.0	pop	51	1766	54276	1	40.323410	17.568270	7900

1538 rows × 9 columns

In [142…
```python
#linear regression
df=data.drop(data.columns[[0,5,6,7]],axis=1)
df
```

Out[142]:

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

1538 rows × 5 columns

In [143…
```python
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()

df['model']= label_encoder.fit_transform(df['model'])
df['model'].unique()
```

Out[143]: array([0, 1, 2])

In [144…
```python
y=df['price']
x=df.drop(['price'],axis=1)
```

In [145…
```python
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.35,random_state=0)
```

In [146…
```python
x_train.shape
```

Out[146]: (999, 4)

In [147…
```python
print(x_train)
```

```
      model  engine_power  age_in_days      km
70        1            51         1096   41950
500       0            51         2527   25191
10        1            51          790   43286
789       0            51          821   19000
505       0            51          456   11186
...     ...           ...          ...     ...
763       0            62         3227   79000
835       1            51          366   16500
1216      0            51         3653  122649
559       1            51          913   20000
684       1            51          943   35481

[999 rows x 4 columns]
```

In [148…
```python
x_test.shape
```

Out[148]: (539, 4)

In [149…
```python
print(y_train)
```

```
70      10700
500      8800
10       8950
789     10050
505     10800
        ...  
763      8500
835     10500
1216     5300
559      8499
684      8650
Name: price, Length: 999, dtype: int64
```

In [150…
```python
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(x,y)
```

Out[150]:

▾ LinearRegression

LinearRegression()

In [152…
```python
reg.coef_
```

Out[152]: array([-1.58929473e+02, 1.03613679e+01, -8.67074079e-01, -1.79050329e-02])

In [153…
```python
reg.intercept_
```

Out[153]: 10480.5504707346

In [154…
```python
from sklearn.metrics import r2_score
r2_score(y_test,reg.predict(x_test))
```

Out[154]: 0.8526571241727813

In [155…
```python
pred=reg.predict(x_test)
from matplotlib import pyplot as plt
print(pred)
```

```
[ 4553.15656425 10200.62493379 10319.43535753 10459.76094262
  9825.61918101  6890.47390478 10414.43946225 10405.72086166
  9997.33113838  6666.78489357  9609.29441519  5215.96154046
  9772.89270105 10025.72857215 10372.89978586  5930.15174514
  9941.2206072  10275.36827683 10449.94514255 10006.92938647
  9926.69967704  7021.78778613  9711.83928189  8361.62450412
  8438.4516223   9832.97814954  9829.65060809  9919.15891505
  7571.03525302  5708.08795565  7891.49849346 10406.0279388
  6817.72091881  8648.16103373  9396.18082304  8744.01761762
  5198.9494916   9880.50090161  8522.40048303  5081.01784088
 10364.71449423  4519.95888108  6877.2711267   9227.46523051
  9116.07751364 10536.76011362  9747.64934775  9087.80597367
  9323.59112797  6937.62240755  7723.1450929   5451.21497848
  7991.35071081  5929.41097722  7210.42036043  5069.19223342
 10406.45112599  5013.60259723  8849.2319917   9460.00751246
  5465.46079954  9954.00095869  8830.34218196 10270.98548887
  8645.94521024  8921.69745043  8127.90352416  8749.58608286
  8153.57855175  9910.02179723  6744.93530249 10035.16178137
 10304.67891882  9519.4067493  10192.48427348  9715.12996593
  4723.90877532  7083.13752954  9670.1384602   6904.10403127
  6877.56535926  6697.15506376  9596.96458728  5166.59721004
  5267.50982099  7995.08612291  9047.62309155 10326.758516
  9033.38904584  9358.53333012  9837.17451239 10342.12103425
  9925.09130135  9809.77602154  9852.28636019  9749.61500781
  9963.45486762  8951.3194404   9957.0958378   5123.08352789
  5613.00943613  8848.98211086  9693.2651776   9329.33424295
  8363.62607734  7484.54949184 10198.40430578  5076.86062309
  9589.77352871  8558.64057895 10345.556109    7973.22527126
 10394.8334512  10369.34052633  7730.87058421  9906.01547044
  7165.85162953 10344.03687278  9923.45886608  9649.97349956
  9501.80879351 10037.48943565  7248.04969059  5954.13146517
  9950.68967804  9463.38464439 10298.5938992   7286.11263511
  9225.60196507  7916.27050715  6845.77686031  7434.9064947
  8537.48001039  9998.90793171  5927.72619991  9602.07697791
  6343.82578806  7738.03259739 10222.11097331 10314.011283
  9386.40472661  6838.82985495  9890.88961118  7633.02964561
  6971.11209081  9863.72388575 10428.81451211 10414.29622199
  6982.63990628 10052.03490754  6330.90372164 10407.4896179
  9750.82985846  6498.92521402  9695.66714359 10271.01745691
  8997.50072316  5775.36832961  9380.32393471 10455.64278504
  7436.04910668  9725.14266981  9749.81206626  6711.04034551
  9992.51033548 10201.80044163  9361.70191084  6787.52492044
  9408.41345017  9911.45975089  4738.96347165 10039.19315691
 10049.25573387  9382.97817708  9863.93808449  9872.855401
  9941.14519659  9955.75130287 10376.35276563  5412.28269711
  9243.9662961   6776.71028055  5725.51391016  9108.0601502
 10423.05562511  9674.25666932  8506.78001936  7969.29201951
  6689.57189454  9940.57218399  5630.48090624  9968.31829679
  9909.36091693  5509.13496533  9621.68749264  9846.61046475
  9767.83963975  7048.41205464  6621.825966   10138.64554847
  7364.45415168 10391.59379068  9608.14678407 10171.00995778
  4777.77146341  9452.41015585 10399.82895538  9097.60381715
  8255.85367845  4685.37714446  4756.56023215  5838.97942056
  6038.95884961 10067.97651652 10357.86070864  9149.671756
  9612.19388008 10330.94165701 10384.93196799  6121.57992678
  9825.67289611  8992.44994951  5593.01101652  9728.97165528
  9990.1691252  10377.32232899  9689.2789404   5095.66674752
 10327.11392508  9307.98083702  4344.004664    9464.18762774
 10294.15075946  3932.58208287  5232.97745141  6160.70134972
 10435.92550177  5565.76316143 10368.00902029  6167.85634893
 10346.8094613   7701.8806269   5471.67773954  8107.51301822
 10382.3753903   8209.48772698  9589.48376472 10184.35923056
 10355.89000458  7322.30160442 10087.10697348  9574.30798084
 10281.83593882  9974.05459557  9424.40886891 10365.63149293
 10062.62026321 10301.20747545  9768.55584106 10174.92114519
  9717.5661493   9643.95746004  9536.39701963  9239.78733126
 10060.96567695  5915.92225466  5047.76550315  4117.92376867
  7777.2033615   9851.26577331 10029.56019765  9647.44894146
  8689.5198876   7633.09353859  9733.08212881  9980.41477582
  9473.07292466  9953.53547938  5936.92132553 10425.59160619
  6629.52622905  7057.33453405 10348.59996459  4953.95263839
  7719.0448919   5856.65769456  8252.03448581  9305.62065613
  9780.30538469 10501.20840227  8832.83098153  9775.61321871
  9856.22167695  7774.96958142 10015.07502601  8453.95608418
  6281.25251092  7105.58328869  5787.33850249 10201.77869458
  7945.99223162  8582.45401502  9856.30669844  5906.99148525
  8088.76265827  6266.6106935   9911.11850791  9990.1691252
  9328.48547784  6088.34141357  6987.31303324 10175.7229781
  9389.95226233  8827.55326765  9787.84614668 10046.8576099
  9804.6052575   6562.84556974  8540.66761327  6403.98919253
  9605.99988913  7770.46135515 10192.50602054  8716.89589332
  9799.02231689  8859.99675851  6077.89113098  4138.08298003
  8363.66957144  4212.53396267  9840.17871591  5437.47133951
  8937.12121319  5555.22208458  9881.91534766  6368.57654591
  7145.92271777  9786.96880006  6631.8513338   7016.10784249
  9989.9577029   6934.84999854  8999.58311502 10221.34889557
  9899.82043214  5408.77277876 10427.59910289  9066.67249221
  9908.79464411 10264.55374003 10237.58876044 10541.77667954
 10030.58462655  8895.84258289  9933.82977371 10093.7715393
  9826.28825237 10439.59603352 10440.96065804  8069.13048312
  5068.79275159  9908.79464411  8683.23003083  9798.65804344
  7833.93080331  5243.95193995 10030.32742248 10025.20937774
  9772.37614668  8811.50608731 10429.89534769  6936.94918489
 10356.42046733  7992.79427869  5209.65762067  9989.1675938
  5380.90427825  7051.2332196   8655.24965457  5605.94373945
  8861.29646759  7378.7341769  10310.53655617 10330.71667872
  6417.07350141 10357.26984255  6841.21664374  6814.33961071
  9541.03493018  9009.7475595  10162.05744132  9970.19100202
 10374.11463652  9785.75125782  9930.08098514 10103.88783136
  8256.66479717 10365.09434194 10343.93059302 10011.65516472
 10432.00814158  6371.5910055   3608.59682421  9995.27211113
  9011.07558434  9791.90674713 10326.7937675   5487.76367924
  9772.37614668  8614.39329207  4971.60275491  9456.22263122
 10466.29627964  8058.07618339  9851.97818415  6512.51408315
  8550.6549858   8648.16103373 10321.83847397 10315.42463016
  9403.35255019  9702.2355991   9780.28198017  5040.17165625
  9183.47200604  5351.1576827   8765.83731073  7840.06050579
  9297.36643595  6047.50593102 10037.68913414  9035.52661252
  6909.24630847  7689.80981515  6639.27857102  8266.27770752
  6980.81651557 10008.0243366  10434.40741599  6931.66750102
  7654.61932485  9804.4978273   9772.37614668 10217.23458002
  7460.48283994  7084.05012766 10283.80434201  5893.24784628
 10396.87188182  5836.06632324 10130.52667833  9964.72556639
  5242.64898875 10260.02760872  8424.36986918  6377.1592379
  9963.33332287  5916.69891207 10331.5206962   4856.57520072
  9721.46560446  8832.83098153  8816.19308369  9755.07614594
  9791.835127    4669.20208352 10377.23664585 10360.14481727
  7189.08755348  6462.04750604 10326.43737585  8804.66326079
  9035.44433415  5835.58695201 10052.03490754 10329.37380125
  6751.56743636  6901.22193108  9416.79277452  8346.98436841
  5122.55134448  6249.19151209 10107.96582982  5501.82192477
  9622.79481001  9341.97928752  9788.21831035 10371.89825446
  5224.86651634  9796.59507109  6502.5418312   7888.67389884
 10078.87070988  8825.56891294 10433.47251226  9994.01870728
  8645.95170072  9585.58705269  6648.32409385 10353.94578231
  7474.1829848   9981.41471454 10396.28375886 10268.96051114
  9863.06139953  9743.59612207 10079.7160369   9468.23815341
  7963.95029312  7116.90590619  5668.41750328 10430.10636607
  8888.60081006  9937.24299831 10104.94038628  4790.21029597
  7160.04307563  8060.80153887  6500.03466268 10386.32586897
  9971.44051231  6596.83637129  7444.7936305   9730.78349327
  5717.58998554  6534.40843234 10028.31068736  7306.29751427
  9964.72556639  4389.14226822 10332.16527738  9600.14379292
  6543.42198713  8034.62053871  7159.96026359  9890.88961118
  9981.98383357 10454.04923711  4420.74744601]
```

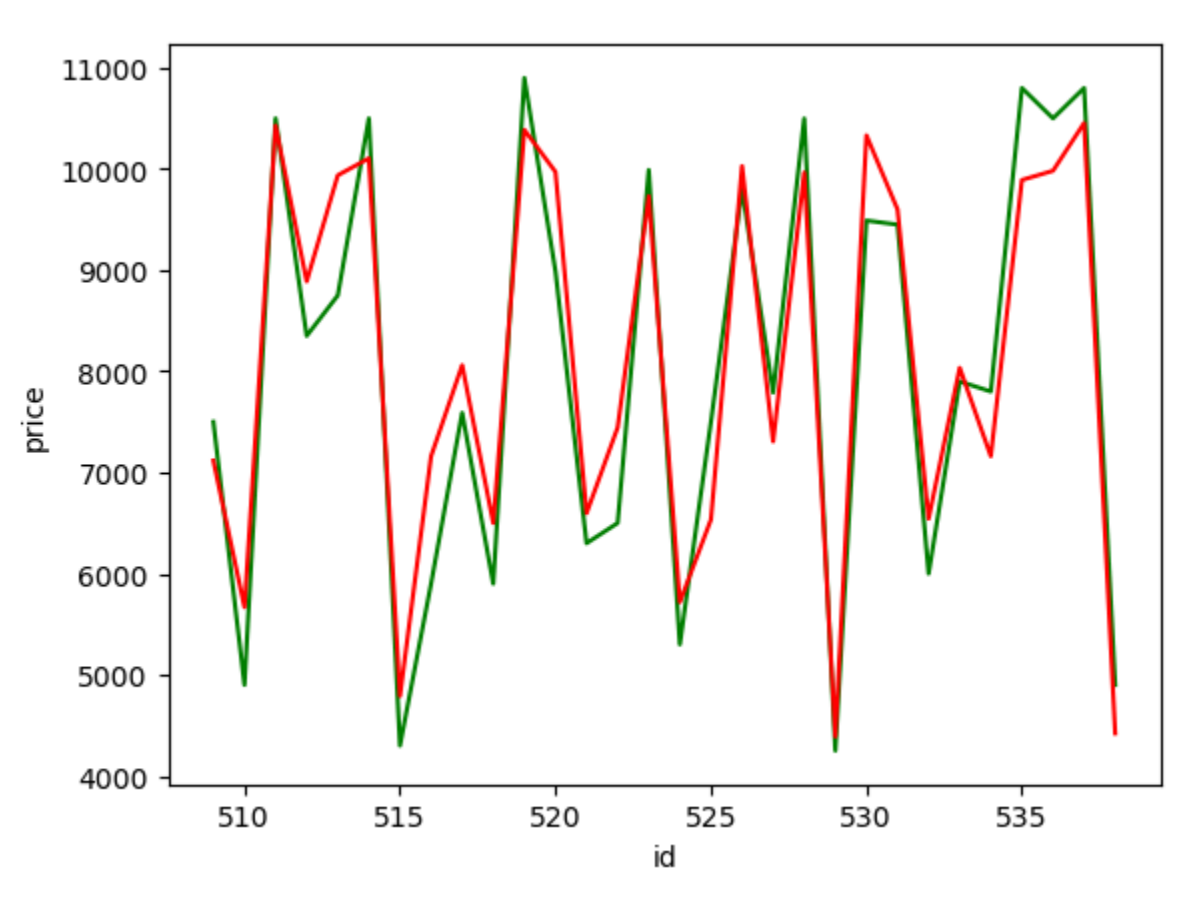
In [156…
```python
results=pd.DataFrame(columns=['price','predicted'])
results['price']=y_test
results['predicted']=pred
results=results.reset_index()
results['id']=results.index
results.head()
```

Out[156]:

	index	price	predicted	id
0	1440	5400	4553.156564	0
1	54	10600	10200.624934	1
2	298	9980	10319.435358	2
3	1335	10400	10459.760943	3
4	861	9800	9825.619181	4

In [157…
```python
import seaborn as sns
sns.lineplot(x='id',y='price',data=results.tail(30),color='green')
sns.lineplot(x='id',y='predicted',data=results.tail(30),color='red')
```

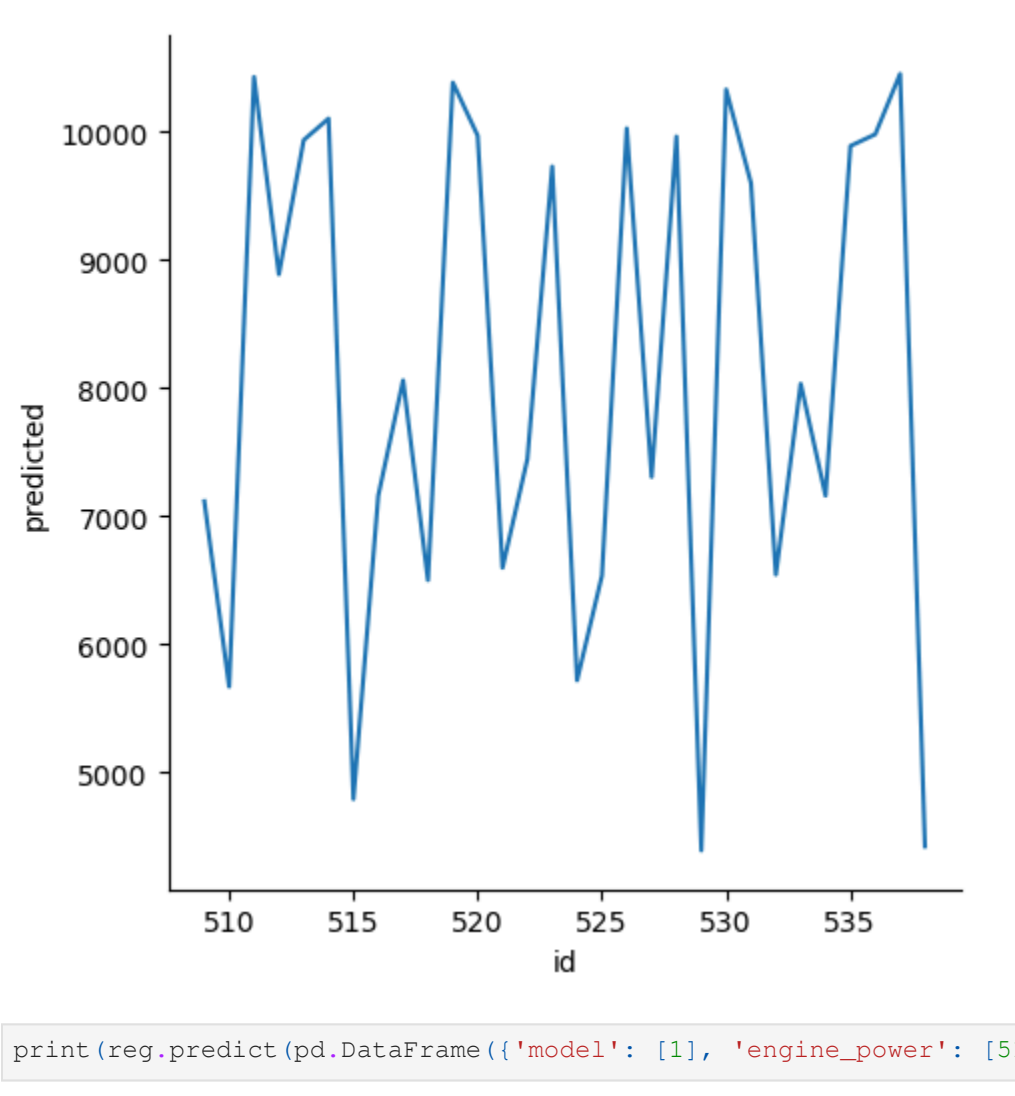
Out[157]: <Axes: xlabel='id', ylabel='price'>



In [158…
```python
sns.relplot(x='id',y='predicted',data=results.tail(30),kind='line')
```

```
C:\Users\rama\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[158]: <seaborn.axisgrid.FacetGrid at 0x28cbc6992d0>



In [169…
```python
print(reg.predict(pd.DataFrame({'model': [1], 'engine_power': [51], 'age_in_days': [10], 'km': [3000]})))
```

```
[10787.66491951]
```

In [160…
```python
print(data)
```

```
          ID   model  engine_power  age_in_days      km  previous_owners  \
0        NaN  lounge            51          882   25000                1   
1        2.0     pop            51         1186   32500                1   
2        3.0   sport            74         4658  142228                1   
3        4.0  lounge            51         2739  160000                1   
4        5.0     pop            73         3074  106880                1   
...      ...     ...           ...          ...     ...              ...   
1533  1534.0   sport            51         3712  115280                1   
1534  1535.0  lounge            74         3835  112000                1   
1535  1536.0     pop            51         2223   60457                1   
1536  1537.0  lounge            51         2557   80750                1   
1537  1538.0     pop            51         1766   54276                1   

            lat        lon  price  
0     44.907242   8.611560   8900  
1     45.666359  12.241890   8800  
2     45.503300  11.417840   4200  
3     40.633171  17.634609   6000  
4     41.903221  12.495650   5700  
...         ...        ...    ...  
1533  45.069679   7.704920   5200  
1534  45.845692   8.666870   4600  
1535  45.481541   9.413480   7500  
1536  45.000702   7.682270   5990  
1537  40.323410  17.568270   7900  

[1538 rows x 9 columns]
```

In [ ]: