Linear Regression

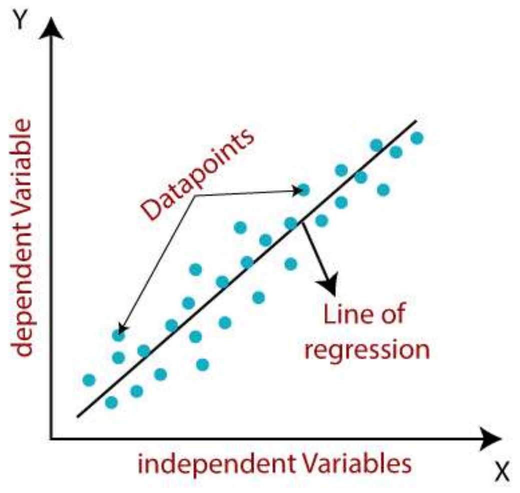
```
Linear Regression: A linear Regression which is used to predicts the values
x ---> Independent
y----> dependent
x--> change y also change
y=mx+c

Functions:
1) errors
2)visualize
3)model fit
4) r2_score
```

In [1]:
```
from IPython.display import Image
```

In [2]:
```
img=Image(filename="C:/Users/Ramakrishna/Desktop/linear-regression-in-machine-learning.png")
```

In [3]:
```
img
```

Out[3]:



In [4]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

In [5]:
```
data=pd.read_csv("C:/Users/Ramakrishna/Desktop/Data.csv")
```

In [6]:
```
data
```

Out[6]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	numb
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	1	
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	1	
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-73.94190	Private room	150	3	
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89	1	
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80	10	
...	...	...	...	...	...	...	...	...	...	...	...	
48890	36484665	Charming one bedroom - newly renovated rowhouse	8232441	Sabrina	Brooklyn	Bedford-Stuyvesant	40.67853	-73.94995	Private room	70	2	
48891	36485057	Affordable room in Bushwick/East Williamsburg	6570630	Marisol	Brooklyn	Bushwick	40.70184	-73.93317	Private room	40	4	
48892	36485431	Sunny Studio at Historical Neighborhood	23492952	Ilgar & Aysel	Manhattan	Harlem	40.81475	-73.94867	Entire home/apt	115	10	
48893	36485609	43rd St. Time Square-cozy single bed	30985759	Taz	Manhattan	Hell's Kitchen	40.75751	-73.99112	Shared room	55	1	
48894	36487245	Trendy duplex in the very heart of Hell's Kitchen	68119814	Christophe	Manhattan	Hell's Kitchen	40.76404	-73.98933	Private room	90	7	

48895 rows × 16 columns

In [7]:
```
data.shape
```

Out[7]:
```
(48895, 16)
```

Data Pre-processing

In [8]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48895 entries, 0 to 48894
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   id                              48895 non-null  int64  
 1   name                            48879 non-null  object 
 2   host_id                         48895 non-null  int64  
 3   host_name                       48874 non-null  object 
 4   neighbourhood_group             48895 non-null  object 
 5   neighbourhood                   48895 non-null  object 
 6   latitude                        48895 non-null  float64
 7   longitude                       48895 non-null  float64
 8   room_type                       48895 non-null  object 
 9   price                           48895 non-null  int64  
 10  minimum_nights                  48895 non-null  int64  
 11  number_of_reviews               48895 non-null  int64  
 12  last_review                     38843 non-null  object 
 13  reviews_per_month               38843 non-null  float64
 14  calculated_host_listings_count  48895 non-null  int64  
 15  availability_365                48895 non-null  int64  
dtypes: float64(3), int64(7), object(6)
memory usage: 6.0+ MB
```

In [12]:
```
data.describe().transpose()
```

Out[12]:

	count	mean	std	min	25%	50%	75%	max
id	48895.0	1.901714e+07	1.098311e+07	2539.00000	9.471945e+06	1.967728e+07	2.915218e+07	3.648724e+07
host_id	48895.0	6.762001e+07	7.861097e+07	2438.00000	7.822033e+06	3.079382e+07	1.074344e+08	2.743213e+08
latitude	48895.0	4.072895e+01	5.453008e-02	40.49979	4.069010e+01	4.072307e+01	4.076311e+01	4.091306e+01
longitude	48895.0	-7.395217e+01	4.615674e-02	-74.24442	-7.398307e+01	-7.395568e+01	-7.393627e+01	-7.371299e+01
price	48895.0	1.527207e+02	2.401542e+02	0.00000	6.900000e+01	1.060000e+02	1.750000e+02	1.000000e+04
minimum_nights	48895.0	7.029962e+00	2.051055e+01	1.00000	1.000000e+00	3.000000e+00	5.000000e+00	1.250000e+03
number_of_reviews	48895.0	2.327447e+01	4.455058e+01	0.00000	1.000000e+00	5.000000e+00	2.400000e+01	6.290000e+02
reviews_per_month	38843.0	1.373221e+00	1.680442e+00	0.01000	1.900000e-01	7.200000e-01	2.020000e+00	5.850000e+01
calculated_host_listings_count	48895.0	7.143982e+00	3.295252e+01	1.00000	1.000000e+00	1.000000e+00	2.000000e+00	3.270000e+02
availability_365	48895.0	1.127813e+02	1.316223e+02	0.00000	0.000000e+00	4.500000e+01	2.270000e+02	3.650000e+02

Duplicate values checking

In [14]:
```
data.duplicated().sum()
```

Out[14]:
```
0
```

In [17]:
```
data.drop_duplicates()
```

Out[17]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	numb
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	1	
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	1	
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-73.94190	Private room	150	3	
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89	1	
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80	10	
...	...	...	...	...	...	...	...	...	...	...	...	
48890	36484665	Charming one bedroom - newly renovated rowhouse	8232441	Sabrina	Brooklyn	Bedford-Stuyvesant	40.67853	-73.94995	Private room	70	2	
48891	36485057	Affordable room in Bushwick/East Williamsburg	6570630	Marisol	Brooklyn	Bushwick	40.70184	-73.93317	Private room	40	4	
48892	36485431	Sunny Studio at Historical Neighborhood	23492952	Ilgar & Aysel	Manhattan	Harlem	40.81475	-73.94867	Entire home/apt	115	10	
48893	36485609	43rd St. Time Square-cozy single bed	30985759	Taz	Manhattan	Hell's Kitchen	40.75751	-73.99112	Shared room	55	1	
48894	36487245	Trendy duplex in the very heart of Hell's Kitchen	68119814	Christophe	Manhattan	Hell's Kitchen	40.76404	-73.98933	Private room	90	7	

48895 rows × 16 columns

Checking Null values and NaN values

```
In [19]: data.isna().sum()   #checking Nan values using isna() function isna means is null values are there
```

```
Out[19]: id                                    0
         name                                 16
         host_id                               0
         host_name                            21
         neighbourhood_group                   0
         neighbourhood                         0
         latitude                              0
         longitude                             0
         room_type                             0
         price                                 0
         minimum_nights                        0
         number_of_reviews                     0
         last_review                       10052
         reviews_per_month                 10052
         calculated_host_listings_count        0
         availability_365                      0
         dtype: int64
```

```
In [21]: data.isnull().sum() # this fuction is used to check the null values in the given dataset
```

```
Out[21]: id                                    0
         name                                 16
         host_id                               0
         host_name                            21
         neighbourhood_group                   0
         neighbourhood                         0
         latitude                              0
         longitude                             0
         room_type                             0
         price                                 0
         minimum_nights                        0
         number_of_reviews                     0
         last_review                       10052
         reviews_per_month                 10052
         calculated_host_listings_count        0
         availability_365                      0
         dtype: int64
```

Deleting the null values

```
In [22]: data.dropna(inplace=True) # the dropna function is used to delete the all null values and inplace function is used to become the
```

Again checking for null values

```
In [24]: data.isna().sum()
```

```
Out[24]: id                                0
         name                              0
         host_id                           0
         host_name                         0
         neighbourhood_group               0
         neighbourhood                     0
         latitude                          0
         longitude                         0
         room_type                         0
         price                             0
         minimum_nights                    0
         number_of_reviews                 0
         last_review                       0
         reviews_per_month                 0
         calculated_host_listings_count    0
         availability_365                  0
         dtype: int64
```

Valusize the data

```
In [25]: import seaborn as sns
```

In [26]:
```
data
```

Out[26]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	1	
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	1	
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89	1	
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80	10	
5	5099	Large Cozy 1 BR Apartment In Midtown East	7322	Chris	Manhattan	Murray Hill	40.74767	-73.97500	Entire home/apt	200	3	
...	...	...	...	...	...	...	...	...	...	...	...	
48782	36425863	Lovely Privet Bedroom with Privet Restroom	83554966	Rusaa	Manhattan	Upper East Side	40.78099	-73.95366	Private room	129	1	
48790	36427429	No.2 with queen size bed	257683179	H Ai	Queens	Flushing	40.75104	-73.81459	Private room	45	1	
48799	36438336	Seas The Moment	211644523	Ben	Staten Island	Great Kills	40.54179	-74.14275	Private room	235	1	
48805	36442252	1B-1B apartment near by Metro	273841667	Blaine	Bronx	Mott Haven	40.80787	-73.92400	Entire home/apt	100	1	
48852	36455809	Cozy Private Room in Bushwick, Brooklyn	74162901	Christine	Brooklyn	Bushwick	40.69805	-73.92801	Private room	30	1	

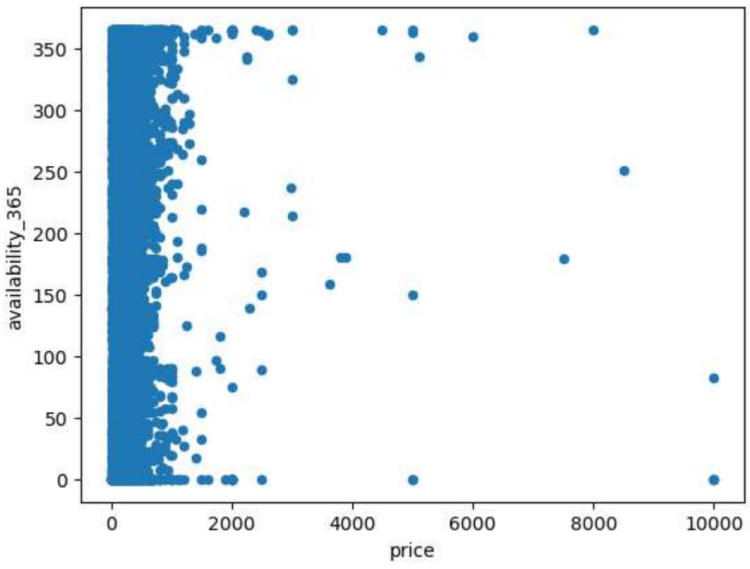
38821 rows × 16 columns

In [27]:
```
data.columns
```

Out[27]:
```
Index(['id', 'name', 'host_id', 'host_name', 'neighbourhood_group',
       'neighbourhood', 'latitude', 'longitude', 'room_type', 'price',
       'minimum_nights', 'number_of_reviews', 'last_review',
       'reviews_per_month', 'calculated_host_listings_count',
       'availability_365'],
      dtype='object')
```

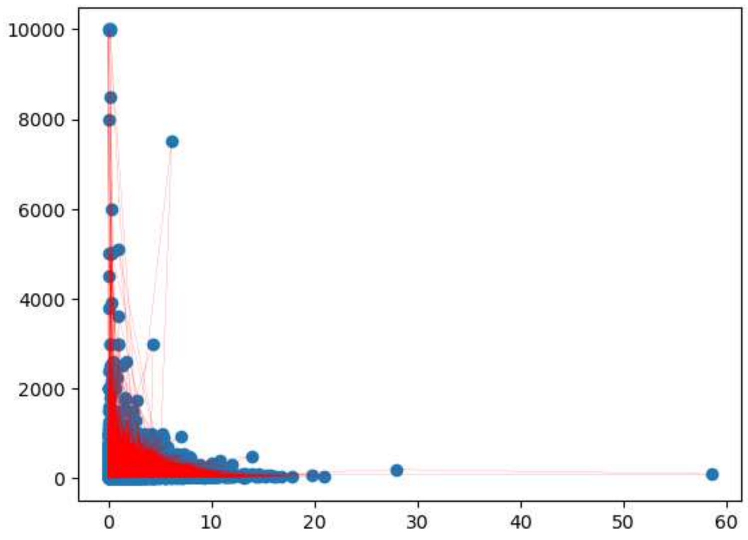
In [36]:
```
data.plot(kind="scatter",x="price",y="availability_365")
```

Out[36]:
```
<AxesSubplot:xlabel='price', ylabel='availability_365'>
```



In [41]:
```
plt.scatter(data["reviews_per_month"],data["price"])
plt.plot(data["reviews_per_month"],data["price"],color="r",lw=0.1)
```

Out[41]:
```
[<matplotlib.lines.Line2D at 0x20d12f2cf40>]
```



Linear Regression

In [42]:
```
data.columns
```

Out[42]:
```
Index(['id', 'name', 'host_id', 'host_name', 'neighbourhood_group',
       'neighbourhood', 'latitude', 'longitude', 'room_type', 'price',
       'minimum_nights', 'number_of_reviews', 'last_review',
       'reviews_per_month', 'calculated_host_listings_count',
       'availability_365'],
      dtype='object')
```

In [119]:
```
from sklearn.model_selection import train_test_split
```

In [120]:
```
data
```

Out[120]:

	id	host_id	latitude	longitude	room_type	price	minimum_nights	number_of_reviews	last_review	reviews_per_month	calculated_host_listin
0	2539	2787	40.64749	-73.97237	Private room	149	1	9	19-10-2018	0.21	
1	2595	2845	40.75362	-73.98377	Entire home/apt	225	1	45	21-05-2019	0.38	
3	3831	4869	40.68514	-73.95976	Entire home/apt	89	1	270	05-07-2019	4.64	
4	5022	7192	40.79851	-73.94399	Entire home/apt	80	10	9	19-11-2018	0.10	
5	5099	7322	40.74767	-73.97500	Entire home/apt	200	3	74	22-06-2019	0.59	
...	...	...	...	...	...	...	...	...	...	...	
48782	36425863	83554966	40.78099	-73.95366	Private room	129	1	1	07-07-2019	1.00	
48790	36427429	257683179	40.75104	-73.81459	Private room	45	1	1	07-07-2019	1.00	
48799	36438336	211644523	40.54179	-74.14275	Private room	235	1	1	07-07-2019	1.00	
48805	36442252	273841667	40.80787	-73.92400	Entire home/apt	100	1	2	07-07-2019	2.00	
48852	36455809	74162901	40.69805	-73.92801	Private room	30	1	1	08-07-2019	1.00	

38821 rows × 12 columns

In [122]:
```
data.columns
```

Out[122]:
```
Index(['id', 'host_id', 'latitude', 'longitude', 'room_type', 'price',
       'minimum_nights', 'number_of_reviews', 'last_review',
       'reviews_per_month', 'calculated_host_listings_count',
       'availability_365'],
      dtype='object')
```

In [123]:
```
x=data[['id', 'host_id', 'latitude', 'longitude', 'room_type','minimum_nights', 'number_of_reviews', 'last_review',
       'reviews_per_month', 'calculated_host_listings_count',
       'availability_365']]
y=data["price"]
```

In [124]:
```
x.shape
```

Out[124]:
```
(38821, 11)
```

In [125]:
```
y.shape
```

Out[125]:
```
(38821,)
```

In [126]:
```
from sklearn.linear_model import LinearRegression
```

In [127]:
```
reg_model=LinearRegression()
```

In [128]:
```
X=x.values.reshape(len(x),-1)
```

In [129]:
```
X
```

Out[129]:
```
array([[2539, 2787, 40.64749, ..., 0.21, 6, 365],
       [2595, 2845, 40.75362, ..., 0.38, 2, 355],
       [3831, 4869, 40.68514, ..., 4.64, 1, 194],
       ...,
       [36438336, 211644523, 40.54179, ..., 1.0, 1, 87],
       [36442252, 273841667, 40.80787, ..., 2.0, 1, 40],
       [36455809, 74162901, 40.69805, ..., 1.0, 1, 1]], dtype=object)
```

In [130]:
```
Y=y.values.reshape(len(y),-1)
```

In [131]:
```
X_train, X_test, y_train, y_test = train_test_split( x, y, test_size=0.2, random_state=42)
```

In [132]:
```
X_train.drop(["last_review"],inplace=True,axis=1)
```

In [133]:
```
y_train
```

Out[133]:
```
34510    232
45484    250
12973    200
33682     69
35097    119
        ... 
7032      65
13389     98
46704     50
897      219
19039    169
Name: price, Length: 31056, dtype: int64
```

In [134]:
```
X_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 31056 entries, 34510 to 19039
Data columns (total 10 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   id                              31056 non-null  int64  
 1   host_id                         31056 non-null  int64  
 2   latitude                        31056 non-null  float64
 3   longitude                       31056 non-null  float64
 4   room_type                       31056 non-null  object 
 5   minimum_nights                  31056 non-null  int64  
 6   number_of_reviews               31056 non-null  int64  
 7   reviews_per_month               31056 non-null  float64
 8   calculated_host_listings_count  31056 non-null  int64  
 9   availability_365                31056 non-null  int64  
dtypes: float64(3), int64(6), object(1)
memory usage: 2.6+ MB
```

In [136]:
```
X_train.drop("room_type",inplace=True,axis=1)
```

In [137]:
```
reg_model.fit(X_train,y_train)
```

Out[137]:
```
LinearRegression()
```

In [138]:
```
reg_model.coef_
```

Out[138]:
```
array([-2.56682883e-07,  3.49513003e-08,  1.53362558e+02, -6.90199123e+02,
        4.93304519e-02, -2.03017498e-01,  3.60772024e-01,  1.04762865e-01,
        1.55203571e-01])
```

In [139]:
```
X_train
```

Out[139]:

	id	host_id	latitude	longitude	minimum_nights	number_of_reviews	reviews_per_month	calculated_host_listings_count	availability_365
34510	27369200	12243051	40.74290	-73.99428	29	2	0.29	96	311
45484	34784681	262287464	40.68713	-73.98592	1	1	0.59	1	365
12973	9854420	33889947	40.72767	-74.00344	2	20	0.48	1	0
33682	26685314	200621725	40.71215	-73.94082	2	70	5.79	2	30
35097	27832008	104607422	40.83005	-73.93992	1	29	3.02	1	103
...	...	...	...	...	...	...	...	...	...
7032	5054397	8873293	40.68951	-73.95524	2	14	0.27	2	0
13389	10041376	51542781	40.76660	-73.99256	7	6	0.14	1	0
46704	35386912	261462340	40.73979	-73.77752	1	1	1.00	1	178
897	325429	92788	40.77610	-73.95265	4	102	1.15	2	280
19039	15125599	3191545	40.76100	-73.98522	30	5	0.15	23	365

31056 rows × 9 columns

In [143]:
```
X_test
```

Out[143]:

host_id	latitude	longitude	room_type	minimum_nights	number_of_reviews	last_review	reviews_per_month	calculated_host_listings_count	availability_365
3967335	40.72527	-73.95016	Entire home/apt	2	31	02-06-2019	0.54	2	12
9898029	40.65041	-73.92574	Entire home/apt	3	10	03-06-2019	0.65	5	156
4622027	40.68194	-73.92896	Entire home/apt	2	147	22-06-2019	1.89	1	27
13974214	40.68058	-73.93856	Entire home/apt	3	92	19-06-2019	1.45	1	248
39288710	40.68544	-73.93872	Entire home/apt	3	64	24-06-2019	1.38	1	296
...	...	...	...	...	...	...	...	...	...
5592622	40.75731	-73.91489	Private room	1	22	22-06-2019	4.93	5	119
4534893	40.72553	-73.98831	Private room	2	46	08-06-2019	1.17	4	307
19802029	40.76468	-73.98514	Entire home/apt	7	37	11-02-2019	0.65	1	97
44881523	40.82940	-73.94695	Entire home/apt	3	13	07-08-2017	0.28	1	0
43825799	40.75896	-73.96251	Entire home/apt	2	37	01-07-2019	4.19	1	135

olumns

In [144]:
```
X_test.drop(["room_type","last_review"],inplace=True,axis=1)
```

In [145]:
```
X_test
```

Out[145]:

	id	host_id	latitude	longitude	minimum_nights	number_of_reviews	reviews_per_month	calculated_host_listings_count	availability_365
5576	4053517	3967335	40.72527	-73.95016	2	31	0.54	2	12
7729	5849991	9898029	40.65041	-73.92574	3	10	0.65	5	156
2020	894015	4622027	40.68194	-73.92896	2	147	1.89	1	27
4195	2730497	13974214	40.68058	-73.93856	3	92	1.45	1	248
9758	7500571	39288710	40.68544	-73.93872	3	64	1.38	1	296
...	...	...	...	...	...	...	...	...	...
40292	31242053	5592622	40.75731	-73.91489	1	22	4.93	5	119
15300	12252205	4534893	40.72553	-73.98831	2	46	1.17	4	307
5659	4126676	19802029	40.76468	-73.98514	7	37	0.65	1	97
11036	8523742	44881523	40.82940	-73.94695	3	13	0.28	1	0
36300	28888147	43825799	40.75896	-73.96251	2	37	4.19	1	135

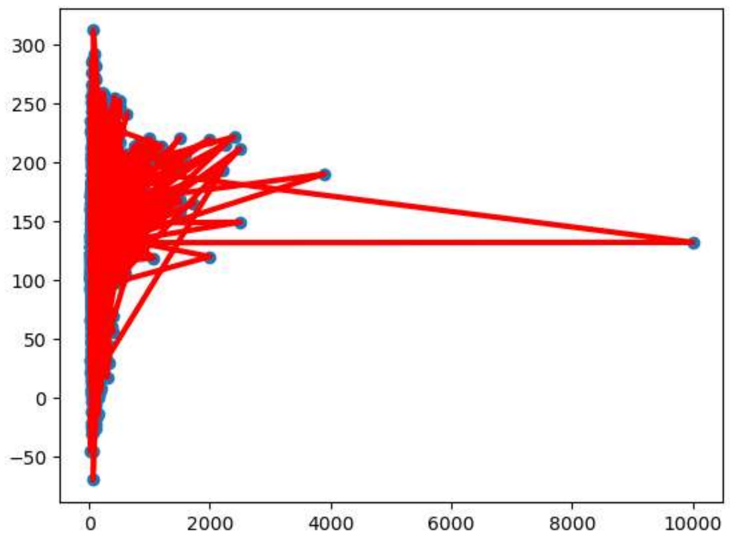
7765 rows × 9 columns

In [146]:
```
pred=reg_model.predict(X_test)
```

In [147]:
```
pred
```

Out[147]:
```
array([125.43944273, 123.86620611,  84.15619733, ..., 168.31706384,
       141.11875947, 153.23265652])
```

```
In [149]: plt.scatter(y_test,pred)
          plt.plot(y_test,pred,color="r",lw=3)

Out[149]: [<matplotlib.lines.Line2D at 0x20d22abb580>]
```



```
In [150]: from sklearn.metrics import accuracy_score,mean_absolute_error,mean_squared_error

In [151]: mae=mean_absolute_error(y_test,pred)

In [152]: mae

Out[152]: 75.53791670451125

In [153]: mse=mean_squared_error(y_test,pred)

In [154]: mse

Out[154]: 31853.704204613874

In [155]: mse1=np.sqrt(mse)

In [156]: mse1

Out[156]: 178.4760605924892

In [159]: from sklearn.metrics import r2_score

In [160]: sc=r2_score(y_test,pred)

In [161]: sc

Out[161]: 0.047755146782353264

In [ ]:
```