**PRACTICAL:3**

**AIM:**

Implement the linear regression and calculate the different evaluation measure (MAE, RMSE etc.). for the same. Also implement gradient descent and observe the cost with linear regression using gradient descent. Do not use any Python library for linear regression. (Hint: Linear Regression Formula is Y= mX +b where Y is target variable and X is independent variable)

**CODE:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

%matplotlib inline

df=pd.read\_csv('data.csv')

df

df['Release Clause']=df['Release Clause'].str.replace(r"€","")

df['Release Clause']=df['Release Clause'].str.replace(r"M","")

df['Release Clause']=df['Release Clause'].str.replace(r"K","")

df['Release Clause']=df['Release Clause'].astype(float)

x=df.Potential[:1000]

df['Release Clause']=df['Release Clause'].fillna(0)

y=df['Release Clause'][:1000]

y

plt.scatter(x,y,color='green')

plt.show()

mean\_x=x.mean()

mean\_y=y.mean()

tempx=[(i-mean\_x)\*\*2 for i in x]

var\_x=sum(tempx,0)/(len(tempx)-1)

tempy=[(i-mean\_y)\*\*2 for i in y]

var\_y=sum(tempy,0)/(len(tempy)-1)

print(var\_x,var\_y)

temp1x=[(i-mean\_x) for i in x]

temp1y=[(i-mean\_y) for i in y]

total=0

for i in range(len(temp1x)):

  for j in range(len(temp1y)):

    if i==j:

      total=total+temp1x[i]\*temp1y[j]

print(total)

cov=total/(len(temp1x)-1)

print(cov)

r=cov/((var\_x\*var\_y)\*\*0.5)

print(r)

slope=r\*((var\_y\*\*0.5)/(var\_x\*\*0.5))

c=mean\_y-slope\*mean\_x

print(slope,c)

generated\_val=[]

for i in range(len(x)):

  new\_y=slope\*x[i] + c

  generated\_val.append(round(new\_y,2))

  print(round(new\_y,2),y[i])

new\_meanY=sum(generated\_val) / len(generated\_val)

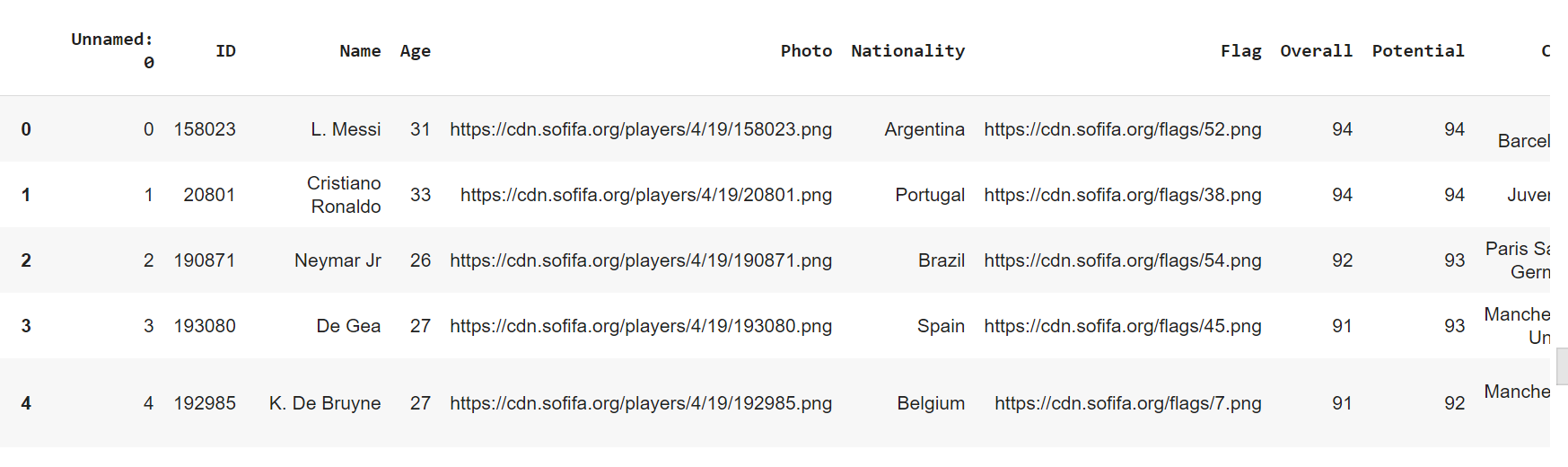
new\_diff=[(i-j)\*\*2 for i,j in zip(y,generated\_val)]

new\_sum=sum(new\_diff)

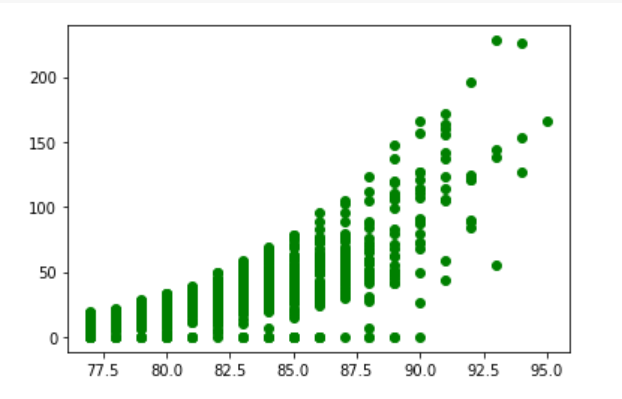
rmse=mse\*\*0.5

print(mse,rmse)

**OUTPUT:**



*Glimpses of dataset*



*Scatter plot*

