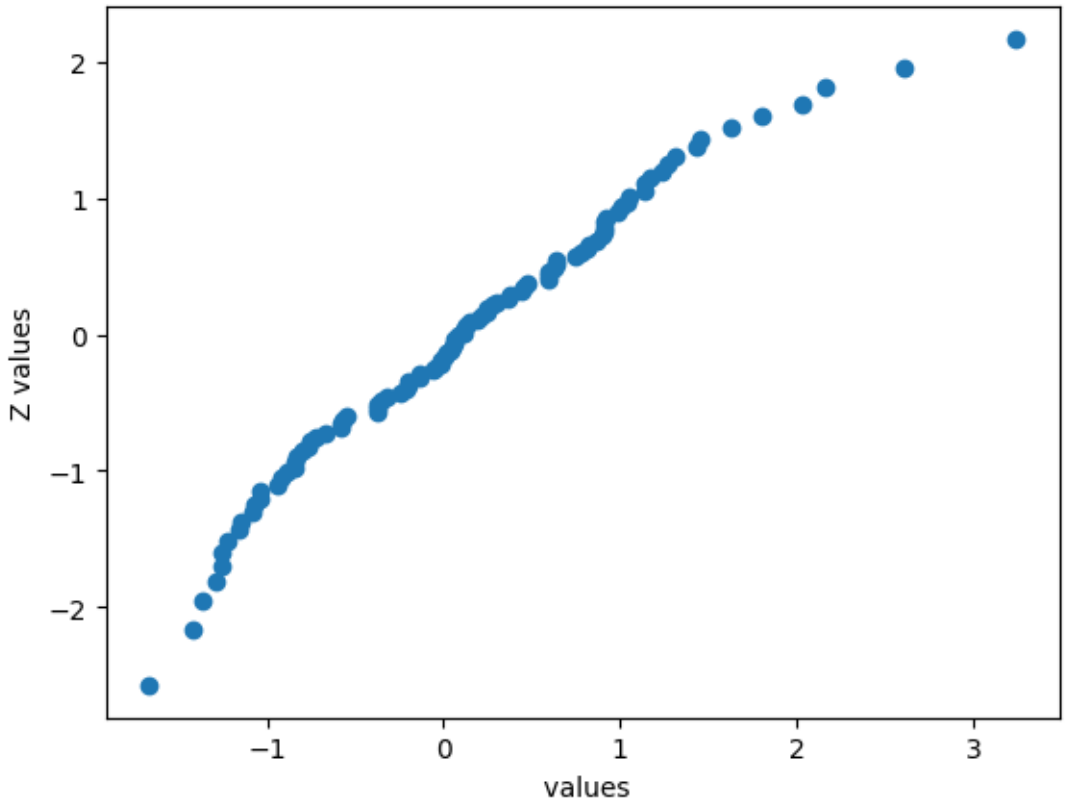
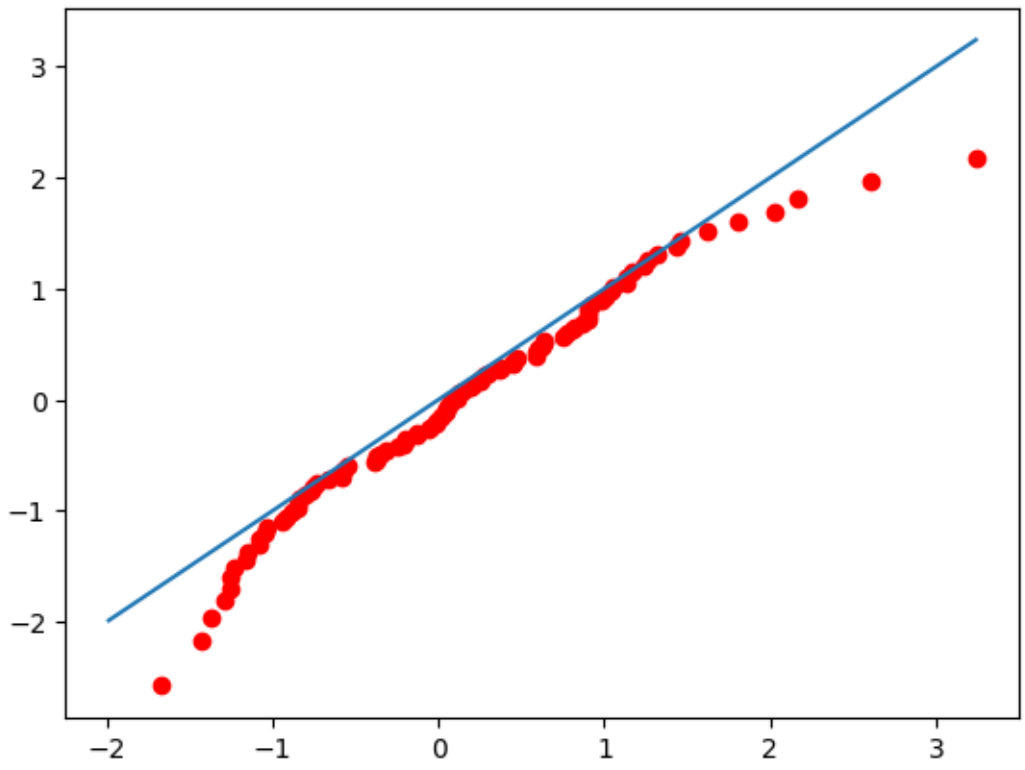
normal-prob-plot

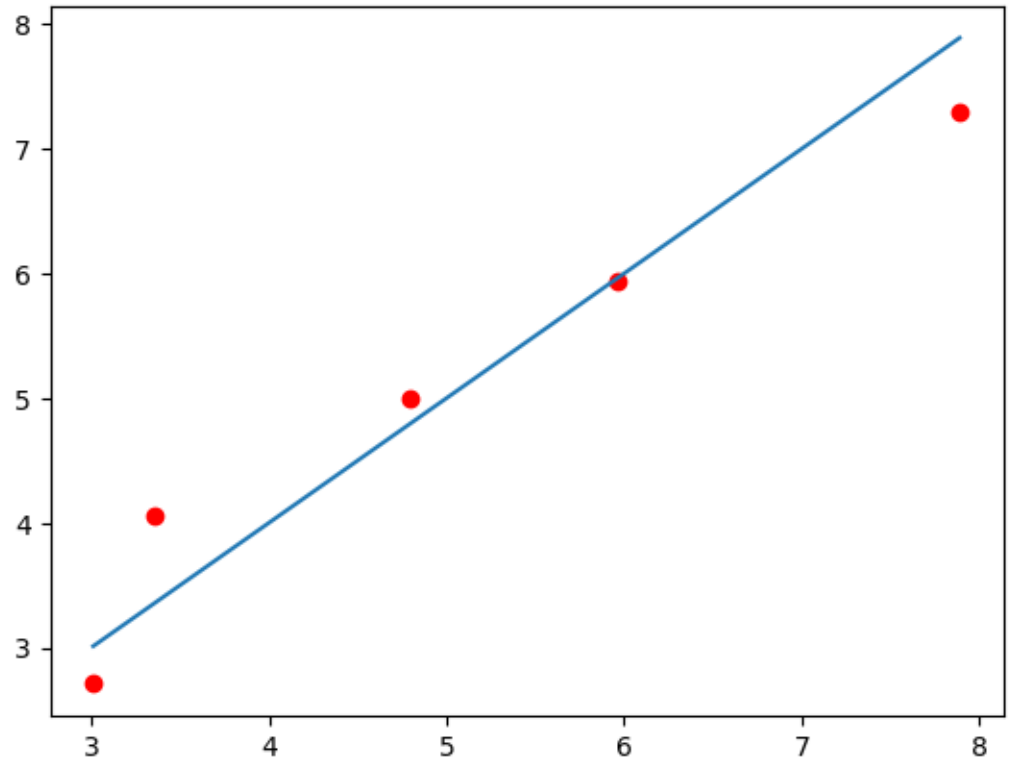
November 1, 2024

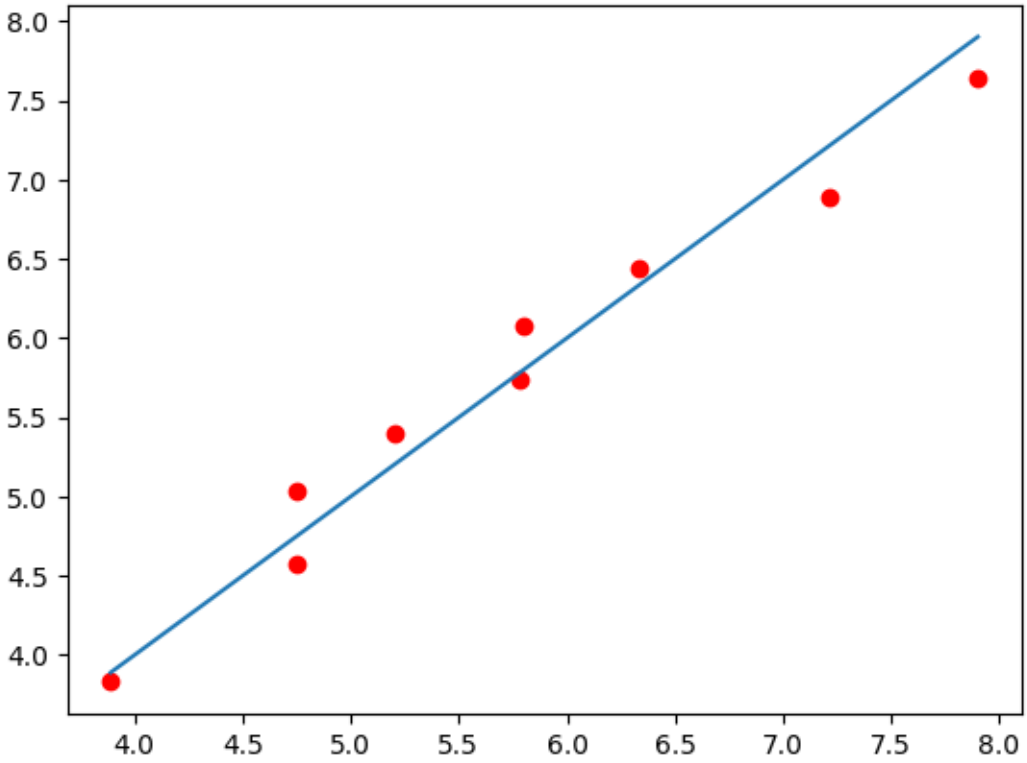
[1]:
```
from scipy.stats import zscore
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
#its just an example npp of original values v/s theortical values(z scores)
def npp(data):
    data = sorted(data)
    p = [(data.index(i)-0.5)/len(data) for i in data]
    z = zscore(p)

    # t = [norm.ppf(i, np.mean(data), np.std(data)) for i in p]
    #xi=[np.std(data)*zi+np.mean(data) for zi in z]

    xi=norm.ppf(p)

    plt.scatter(data, xi)
    plt.ylabel('Z values')
    plt.xlabel(' values')
    plt.show()
    plt.plot(data,  xi,'ro',data,  data)
    plt.show()

#n datapoints
n = 100
data = np.random.randn(n)
npp(data)
```

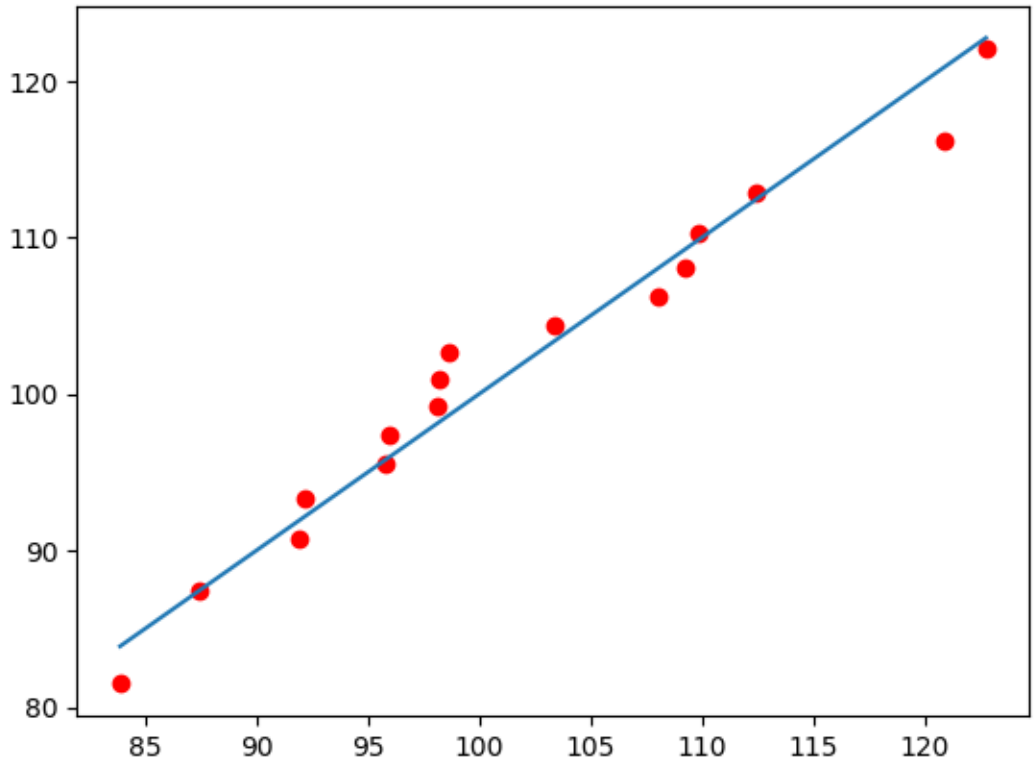

Z values
values
2
1
0
−1
−2
−1
0
1
2
3



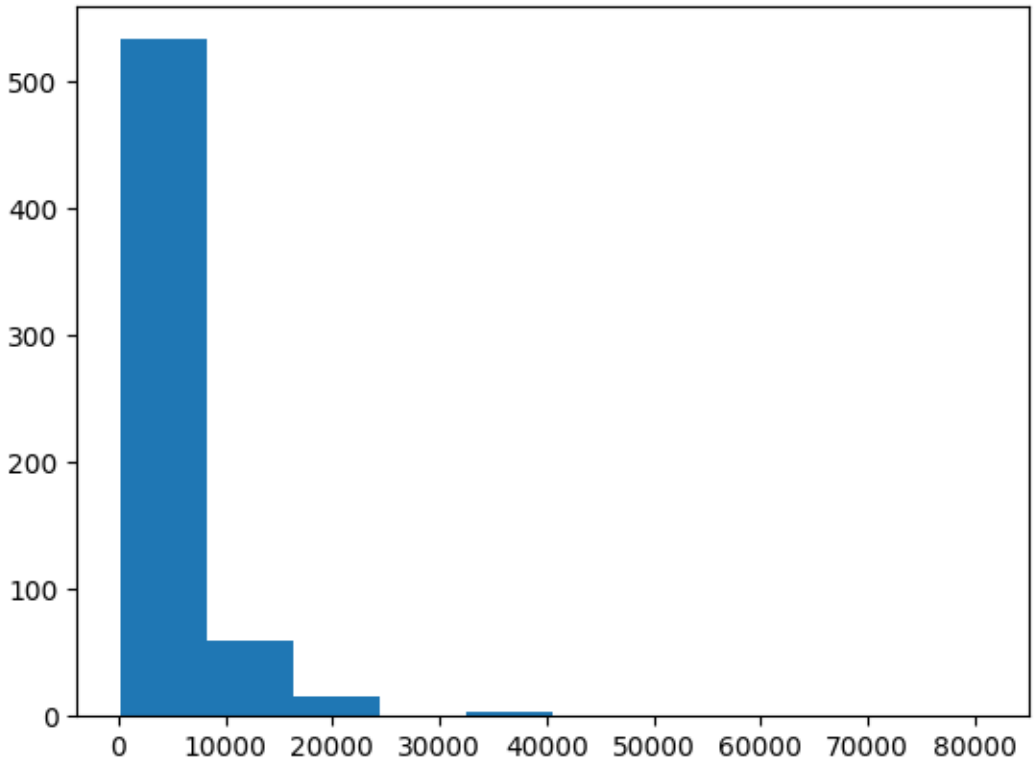
[2]:
```
X1 = [3.01, 3.35, 4.79, 5.96, 7.89]

X2 = [3.89, 4.75, 4.75, 5.20, 5.78, 5.80, 6.33, 7.21, 7.90]

X3 = [108.047, 109.249, 103.385, 112.454, 95.780,  122.734, 109.842 , 120.858,
      98.604, 98.122,  95.971, 98.173,   87.437 , 91.884, 92.193,  83.882]
def npp1(data):
    p =[]
    t =[]
    data = np.sort(np.array(data))
    p = [(i - 0.5)/len(data) for i in range(1, len(data)+1)]
    t = [norm.ppf(i, np.mean(data), np.std(data)) for i in p]
    plt.plot(data, t, 'ro', data, data)
    plt.show()

npp1(X1)
npp1(X2)
npp1(X3)
```

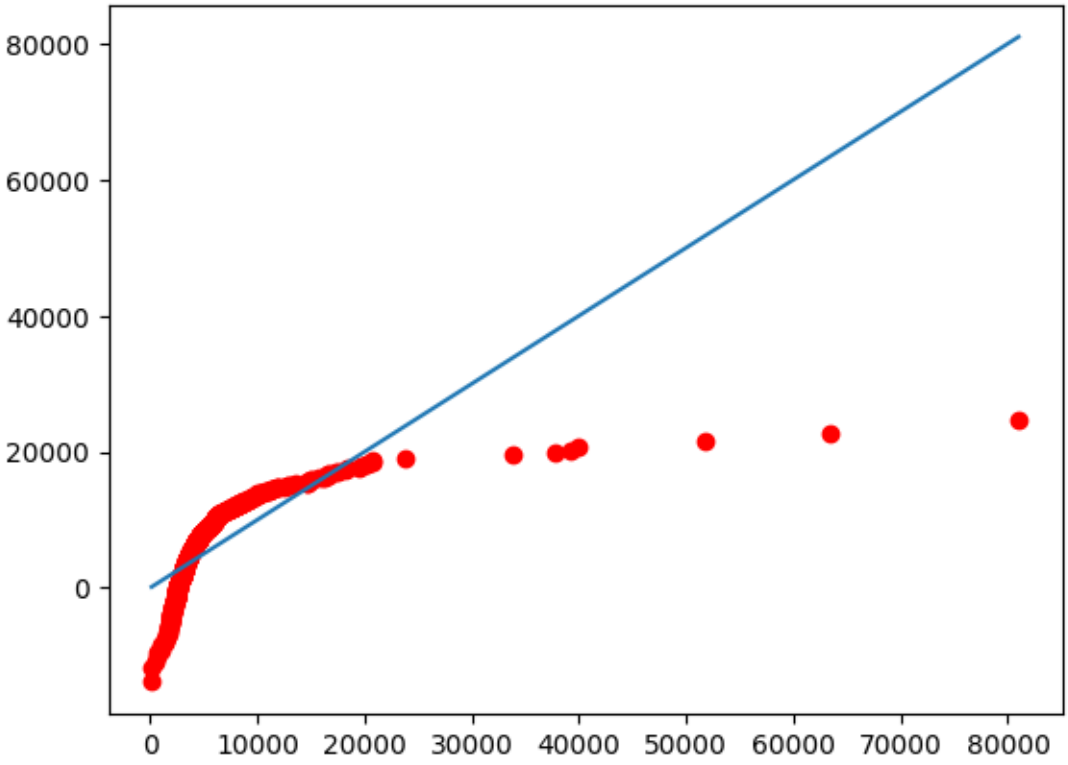

8
7
6
5
4
3
3
4
5
6
7
8

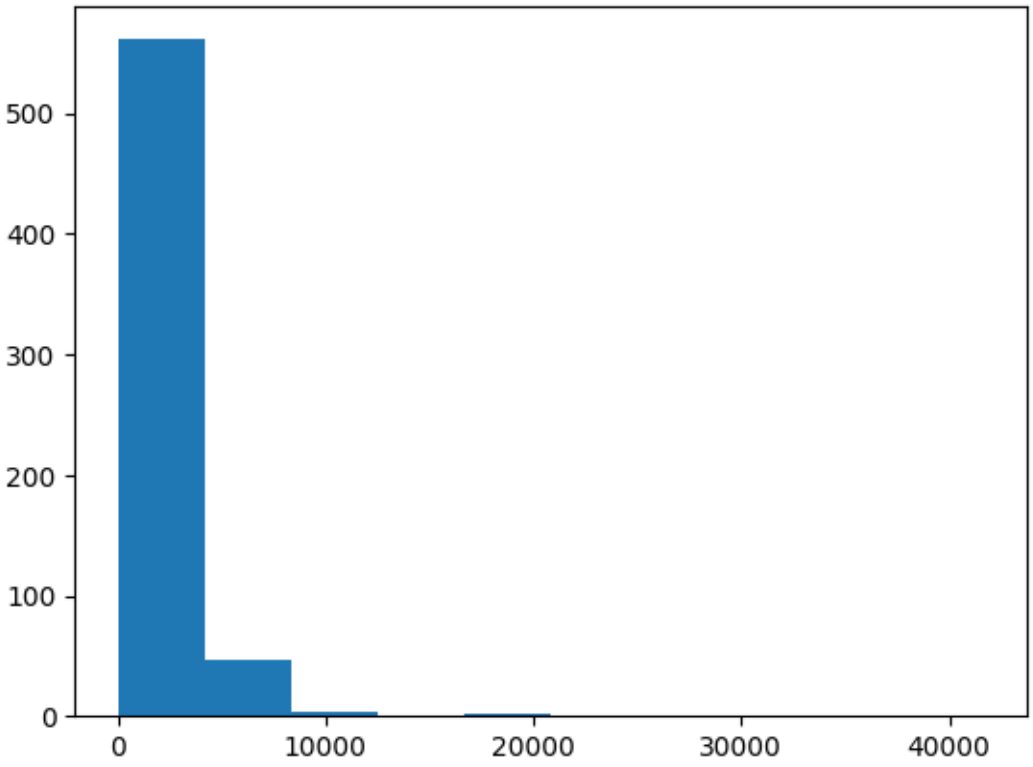

8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
4.0
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0

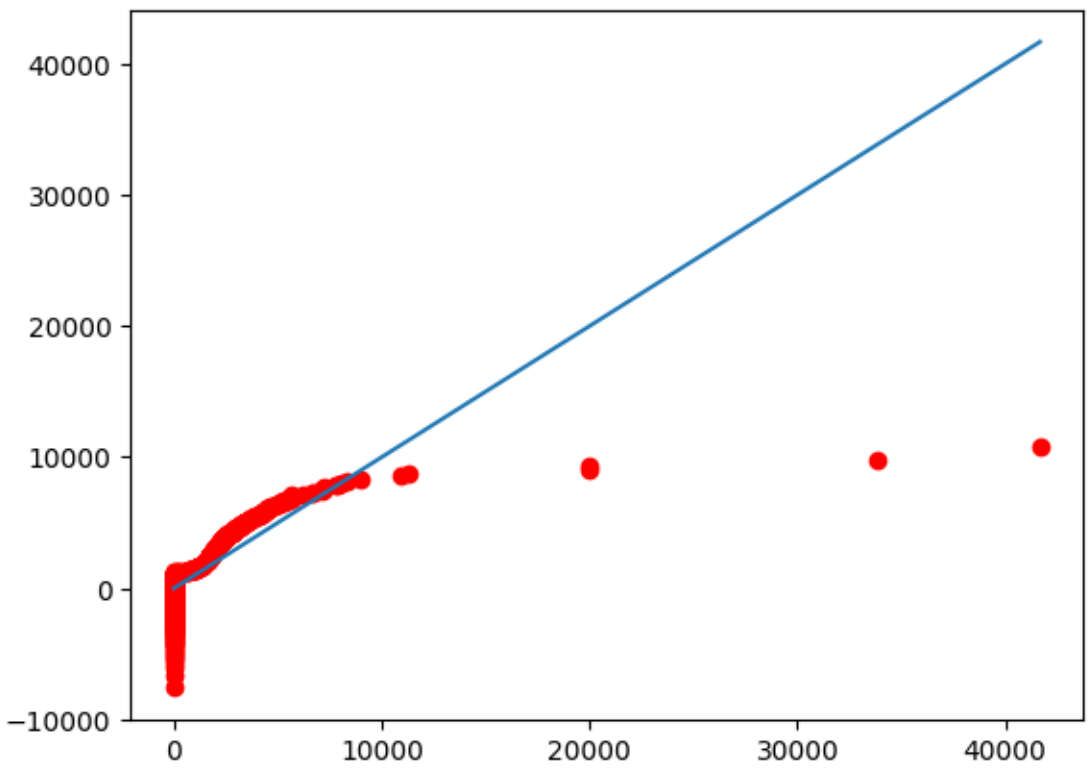


[3]:
```
import pandas as pd

df = pd.read_csv('train.csv')
plt.hist(df.ApplicantIncome)
plt.show()
npp1(df.ApplicantIncome)
df1 = pd.read_csv('train.csv')
plt.hist(df1.CoapplicantIncome)
plt.show()
npp1(df1.CoapplicantIncome)
```


500
400
300
200
100
0
0
10000
20000
30000
40000
50000
60000
70000
80000


80000
60000
40000
20000
0
0
10000
20000
30000
40000
50000
60000
70000
80000


500
400
300
200
100
0
0
10000
20000
30000
40000


40000
30000
20000
10000
0
−10000
0
10000
20000
30000
40000

[ ]: