```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression


df = pd.read_csv('/gdrive/My Drive/klimat.csv', sep=';', decimal=',')
df.columns = ['City', 'Lat', 'Long', 'Prec', 'Temp', 'Alt']
df['Temp'] = pd.to_numeric(df['Temp'])
df.describe
```

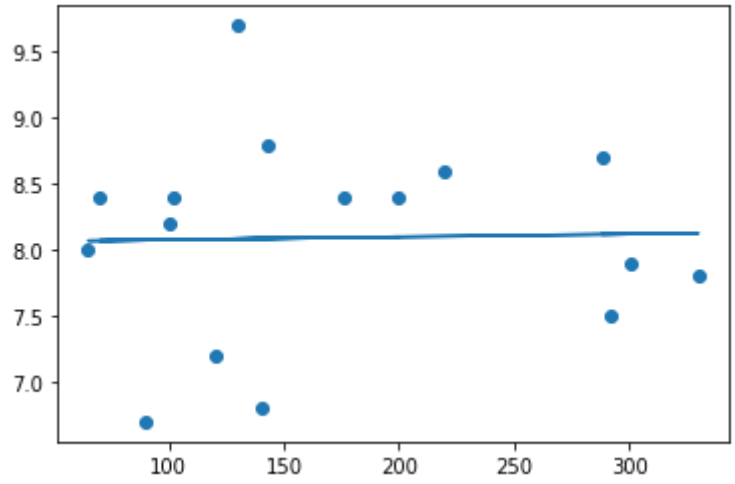
```
<bound method NDFrame.describe of             City              Lat
    0        Wrocław             51.11  17.0222222222222   548   9.7  130
    1          Opole  50.6647222222222  17.9269444444444   611   8.4  176
    2       Katowice  50.2641666666667  19.0236111111111   721   7.9  301
    3         Kraków  50.0613888888889  19.9383333333333   671   8.7  288
    4        Rzeszów  50.0336111111111  22.0047222222222   615   7.5  292
    5         Kielce  50.8741666666667  20.6333333333333   629   7.8  330
    6           Łódź  51.7766666666667  19.4547222222222   582   8.6  220
    7         Poznań  52.4083333333333  16.9341666666667   520   8.4  102
    8   Zielona Góra  51.9397222222222            15.505   600   8.8  143
    9       Szczecin  53.4380555555556  14.5422222222222   539   8.0   65
    10        Gdańsk           54.3475  18.6452777777778   541   6.7   90
    11     Bydgoszcz            53.125  18.0011111111111   533   8.4   70
    12      Warszawa  52.2322222222222  21.0083333333333   501   8.2  100
    13     Białystok  53.1352777777778  23.1455555555556   574   6.8  140
    14        Lublin  51.2480555555556  22.5702777777778   570   8.4  200
    15       Olsztyn  53.7730555555556  20.4761111111111   654   7.2  120>
```

We have to calculate β_0 and β_1 so that linear function $y = \beta_0 + \beta_1 x$ is closest to the data points X is altitude and Y is temperature.

```
X = df['Alt']
Y = df['Temp']
X_average = np.average(X)
Y_average = np.average(Y)
B_1 = np.sum((X-X_average)*(Y-Y_average))/np.sum((X-X_average)*(X-X_average))
B_0 = Y_average - B_1*X_average


def f(B_1, B_0, X):
  return B_0 + B_1*X

plt.plot(X, f(B_1, B_0, X))
plt.scatter(X, Y)
plt.show()
```




```
np.sum((f(B_1, B_0, X) - Y)**2)
```

```
8.983758690160517
```