

TANGLATEC SCHOOLSTRAAT 8 9883, BELLEM, BELGIUM +32486257049 info@tanglatec.com www.tanglatec.com VAT: BE0644.780.972

DETAILED WEEKLY REPORT

MACHINE LEARNING AND DATA SCIENCE WEEK 3

Submitted by Nyuysemo Calglain

Under the supervision of

TANGLATEC

01/08/2024

**Task**

**Develop a python function that returns the peak position of the gaussian plot for a given data set irrespective of its format.**

Table of Contents

[**1** **Introduction** 3](#_Toc173399175)

[**2** **Problem statement** 3](#_Toc173399176)

[**3** **Procedure (plan of work)** 3](#_Toc173399177)

[**4** **Implementation procedure** 4](#_Toc173399178)

[**4.1** **Module importation** 4](#_Toc173399179)

[**4.2** **Data Importation** 4](#_Toc173399180)

[4.2.1 **Importation test** 5](#_Toc173399181)

[**4.3** **Data visualization** 5](#_Toc173399182)

[**4.4** **Determining the peak position** 6](#_Toc173399183)

[**4.5** **Direct maximum search:** 6](#_Toc173399184)

[**4.6** **Gaussian Fit method:** 7](#_Toc173399185)

[**4.7** **Peak position analysis** 8](#_Toc173399186)

[4.7.1 **Discoveries:** 8](#_Toc173399187)

[4.7.2 **To improve consistency:** 8](#_Toc173399188)

[**5** **Conclusion** 9](#_Toc173399189)

[**6** **Difficulties** 9](#_Toc173399190)

[**7** **References** 9](#_Toc173399191)

**Table of Figures**

[**Figure 1 importation and Read data 4**](file:///C:\Users\CarlStorm\Desktop\internships2024\Tangla%20Tech%20internship\weekly%20report%20week%203.docx#_Toc173399606)

[**Figure 1 importation and Read data 4**](file:///C:\Users\CarlStorm\Desktop\internships2024\Tangla%20Tech%20internship\weekly%20report%20week%203.docx#_Toc173399607)

[**Figure 2 data display 5**](file:///C:\Users\CarlStorm\Desktop\internships2024\Tangla%20Tech%20internship\weekly%20report%20week%203.docx#_Toc173399608)

[**Figure 2 data display 5**](file:///C:\Users\CarlStorm\Desktop\internships2024\Tangla%20Tech%20internship\weekly%20report%20week%203.docx#_Toc173399609)

[**Figure 3 data visualization 5**](file:///C:\Users\CarlStorm\Desktop\internships2024\Tangla%20Tech%20internship\weekly%20report%20week%203.docx#_Toc173399610)

[**Figure 3 data visualization 5**](file:///C:\Users\CarlStorm\Desktop\internships2024\Tangla%20Tech%20internship\weekly%20report%20week%203.docx#_Toc173399611)

[**Figure 4 DirectMax implementation 6**](#_Toc173399612)

[**Figure 5 gaussian fit method 7**](file:///C:\Users\CarlStorm\Desktop\internships2024\Tangla%20Tech%20internship\weekly%20report%20week%203.docx#_Toc173399613)

[**Figure 5 gaussian fit method 7**](file:///C:\Users\CarlStorm\Desktop\internships2024\Tangla%20Tech%20internship\weekly%20report%20week%203.docx#_Toc173399614)

[**Figure 6 peak position analysis 9**](file:///C:\Users\CarlStorm\Desktop\internships2024\Tangla%20Tech%20internship\weekly%20report%20week%203.docx#_Toc173399615)

[**Figure 6 peak position analysis 9**](file:///C:\Users\CarlStorm\Desktop\internships2024\Tangla%20Tech%20internship\weekly%20report%20week%203.docx#_Toc173399616)

# **Introduction**

This report contains a detailed description of how I was able to build python modules that are able to detect the peak position of a gaussian plot.

The report also consists of the various difficulties faced, screenshots of various instances of the implementation, lessons learnt and comparisons of the values gotten from the different functions.

# **Problem statement**

* Develop Python and C++ functions to determine the peak position of a Gaussian distribution from given x, y data.
* Functions should be versatile to handle data from various sources (CSV, TXT, etc.).

# **Procedure (plan of work)**

* Implement a function that will handle reading of data in any format
* Implement two methods: direct maximum search and Gaussian fitting.
* Apply functions to multiple datasets provided

# **Implementation procedure**

## **Module importation**

All the Modules needed to solve the problem above were imported and they are;

* **Numpy** for numerical calculations and analysis
* **SciPy** for scientific calculations like the Gaussian fitting
* **Pandas** for data description
* **Matplotlib** for data visualization

## **Data Importation**

The challenge here is to be able to import a dataset in any format not just a particular format and for this I targeted the problem by creating a function called ReadData (fileLocation) that will read the data in the specified fileLocation and return a pandas dataFrame. To implement this function, I use the try except procedure where I do all the implementation in the try and then except the case where the data format Is not supported. If else statements were also used to verify the data format and read the data based on its format. The python function endswith (“string”) to check the end of the file so we can determine its format. This implementation can be seen below

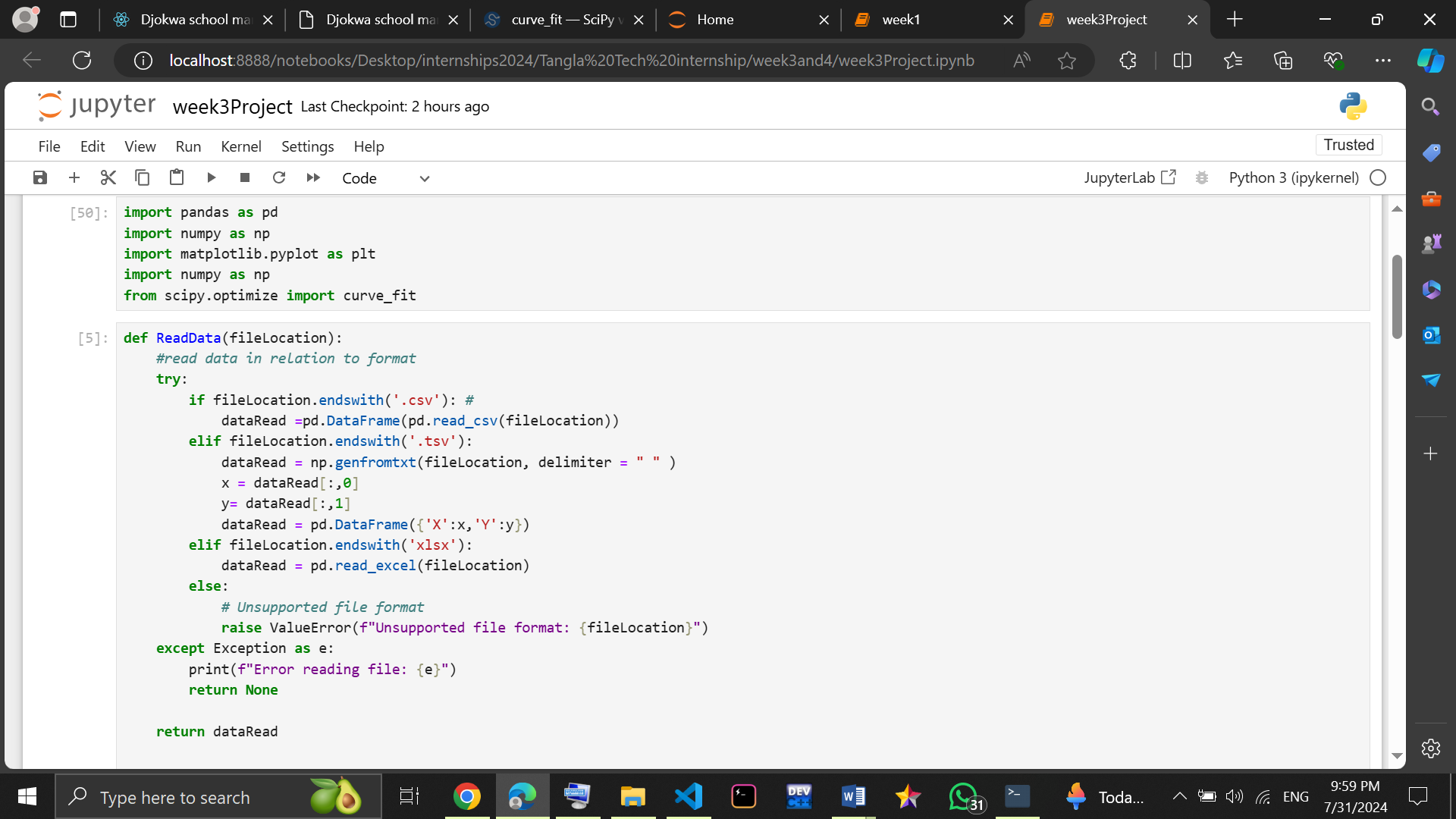
**Screenshot:**

Figure 1 importation and Read data

Figure 1 importation and Read data

### **Importation test**

After implementing the ReadData function I test it by parsing the file location of some datasets and storing what the function will return in data and then displaying this data as seen below;

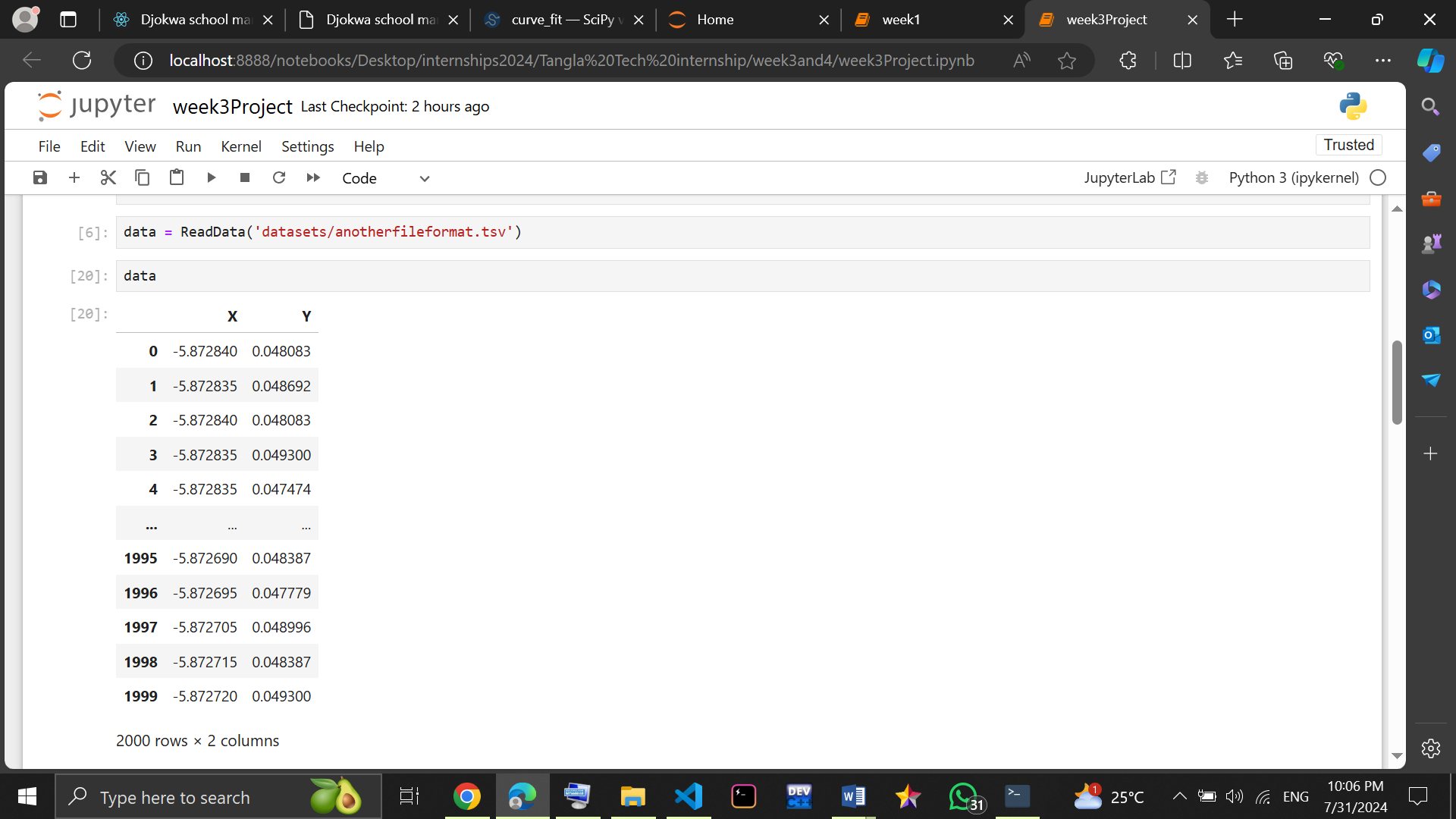
Screenshot:

Figure 2 data display

Figure 2 data display

## **Data visualization**

Here I separate the dataFrame into components x\_value and voltage and then use matplotlib to plot this data so we can see the data display and a picture image of the range where our peak value is located. As seen below;

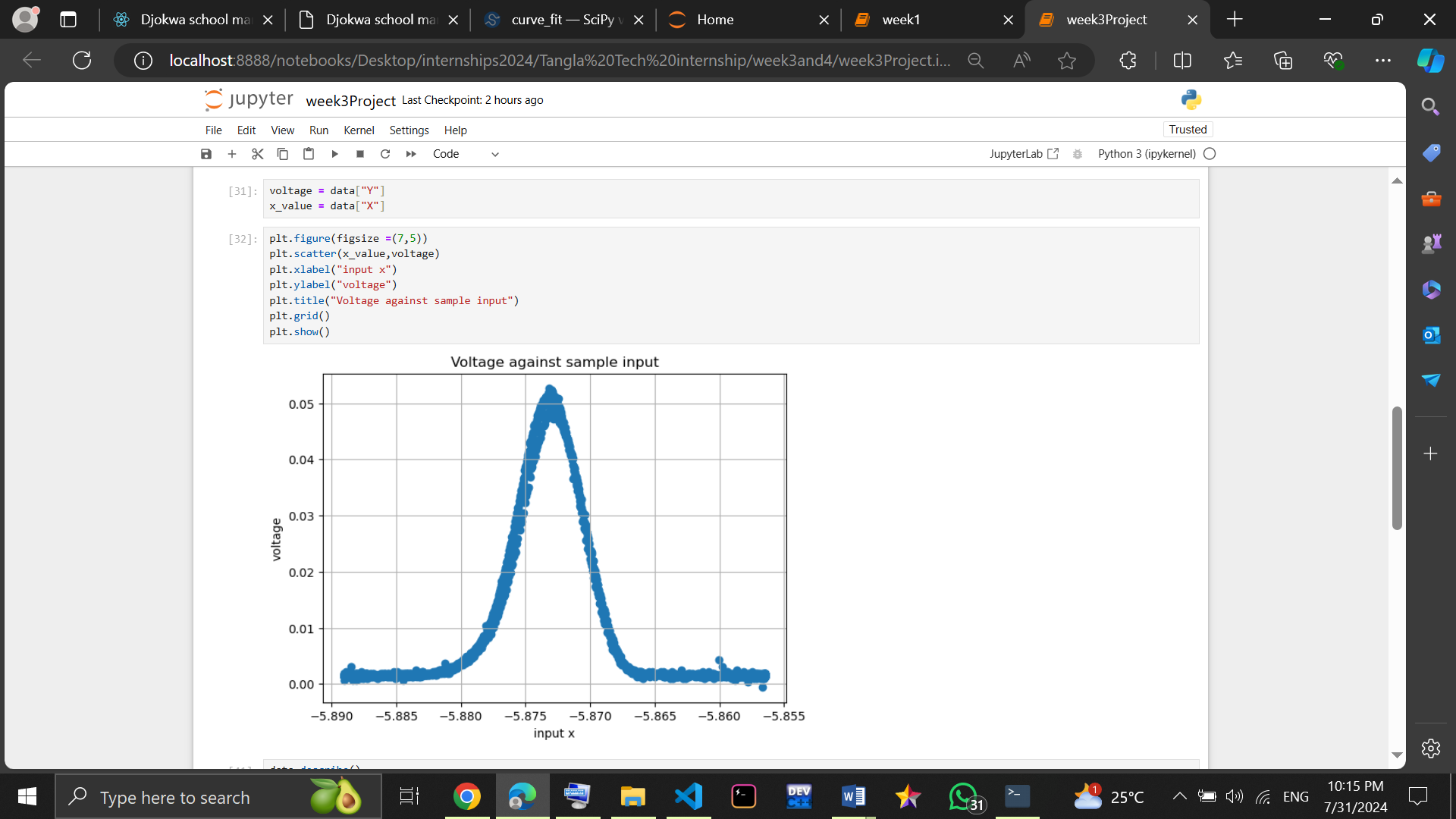
**Screenshot:**

Figure 3 data visualization

Figure 3 data visualization

**From the graph we can already guess that the x value for which the voltage is maximum should be between -5.875 and -5.870. This is our main aim of trying to visualize the data.**

we can further use the pandas describe () function to see the mean, and actual max value of the data.

## **Determining the peak position**

Here I’ll implement the functions that will determine the peak position.

We will do this using two methods and at the end we see what the results will look like.

**First**

## **Direct maximum search:**

Here I created a function called DirectMaxSearch (voltage\_component) that will take the y value as input. The numpy function argmax that finds the index of the largest number is then used with this y input to find the index of the largest voltage value and then our function will return the value of x at this position.

We can test this function by calling it and saving it’s value in a variable called directMax which we can the print out it’s content as seen below.

**Screenshot:**

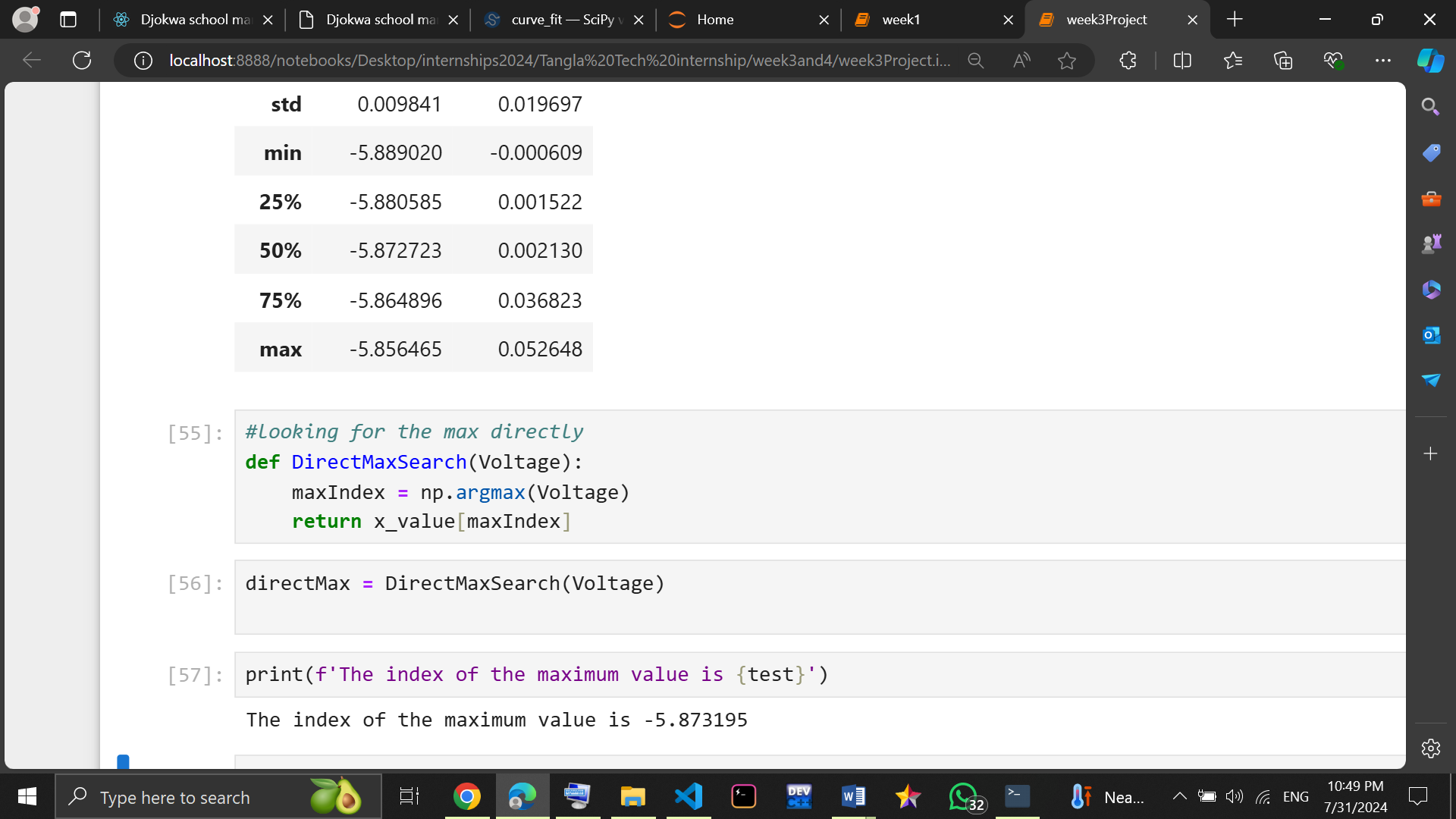


Figure 4 DirectMax implementation

**Lastly**

## **Gaussian Fit method:**

Procedure:

I followed the algorithm below to implement the Gaussian Fit method.

I started by defining the function gaussian that takes 4 parameters which are the x\_value, the amplitude, the mean and the standard deviation and calculate the peak position based on the equation below.

#### **Gaussian Equation**

**f(x) = amplitude \* exp (-(x - mean) ^2 / (2 \* std^2))**

I then proceed by defining another function **find\_peak (x,y)** that takes as parameters x and y and that will be responsible for fitting this data into our gaussian function above following the procedure below

* The function takes the values x and y gotten from our data as input.
* Provides initial guesses for the Gaussian parameters (amplitude, mean, standard deviation).
* Uses curve\_fit **(a Scipy.optimize library)** to fit the Gaussian to the data.
* Extracts the peak position (mean) from the fitted parameters.
* Returns the peak position or None if fitting fails.

The code implementation can be seen on the screenshot below.

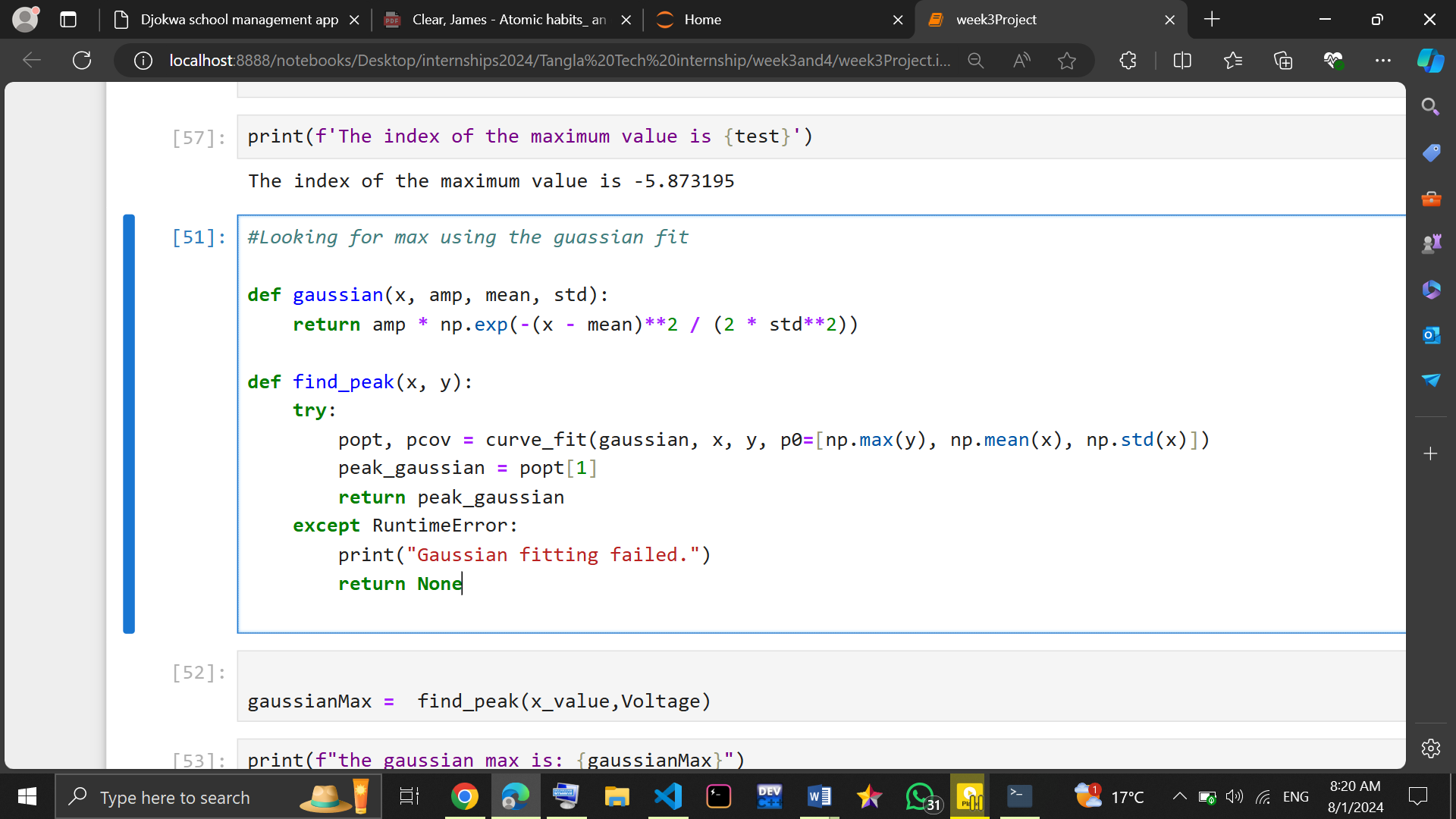
 **Screenshot:**

Figure 5 gaussian fit method

Figure 5 gaussian fit method

This code fits a Gaussian curve to your data, extracts the parameters of the fitted Gaussian, and then assigns the fitted mean (which is the peak position) to the peak\_gaussian variable.

## **Peak position analysis**

I decided to compare the values gotten from both the direct method and the gaussian fit method and discovered they have a slight difference between them I then went to do research on what could possibly the problem.

### **Discoveries:**

#### **Noise and Data Quality:**

Real-world data often contains noise, which can affect the maximum value and the Gaussian fit.

Outliers can significantly influence the direct maximum search but might have less impact on the Gaussian fit.

#### **Gaussian Assumption:**

The Gaussian fit assumes the data perfectly follows a Gaussian distribution.

Deviations from this assumption can lead to discrepancies between the fitted peak and the actual peak.

#### **Definition of Peak:**

The definition of the peak might differ slightly between methods. For example, the Gaussian fit might give the mean of the distribution, while the direct maximum might give the exact location of the highest data point.

### **To improve consistency:**

Multiple fit attempts: Try different initial parameter estimates for the Gaussian fit.

Peak refinement: Combine the results from both methods to estimate a more accurate peak position.

After calling the **find\_peak (x,y)** I store its value in gaussianMax variable and then compare the results of direct max and find their mean to get a better approximation of the max position as seen below .

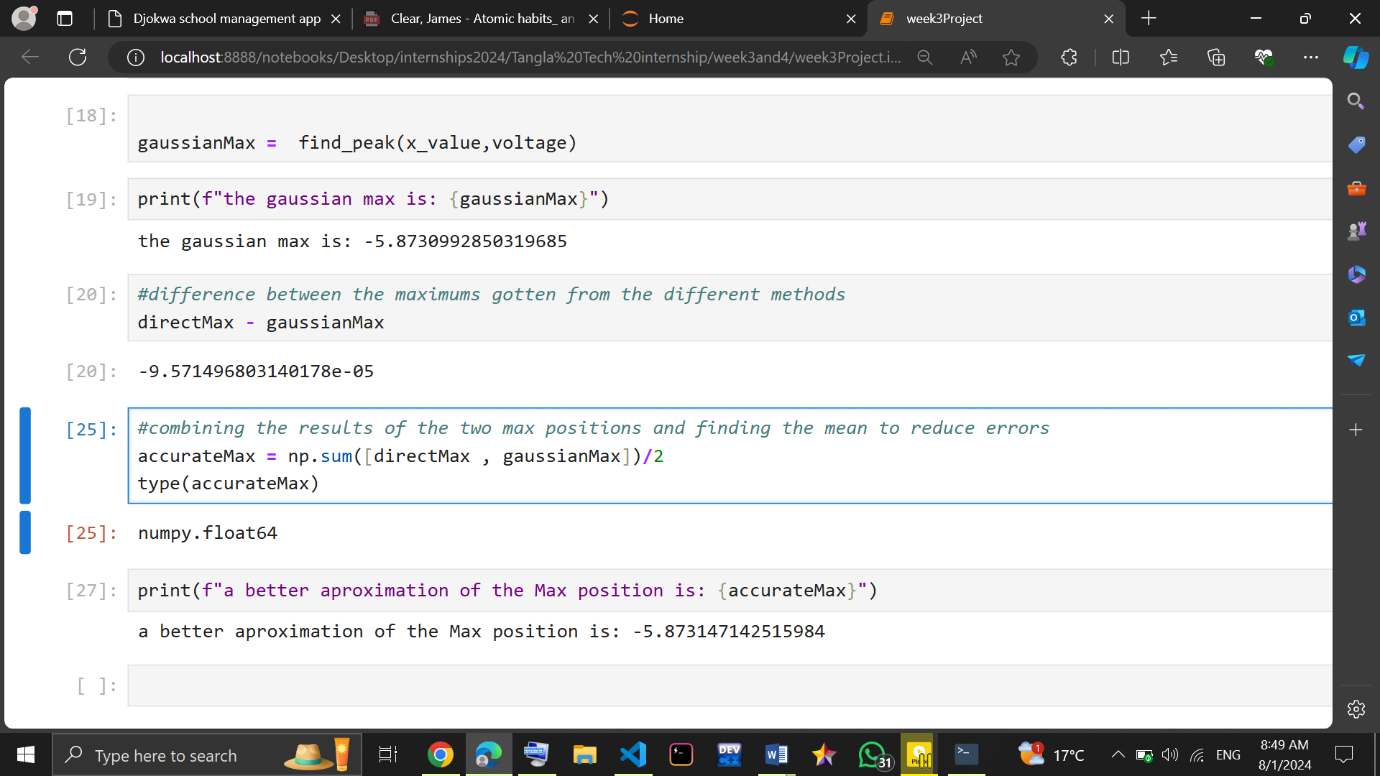
**Screenshot:**

Figure 6 peak position analysis

Figure 6 peak position analysis

# **Conclusion**

We have worked through the library importation, reading data in any format, creating a dataFrame from the data and separating it to x and y components, describing and visualizing our data and finally determining the maximum position or peak value using a direct method and using the gaussian fit approach. We have seen some sources of error in the values gotten from the different methods and how to get a more accurate approximation.

# **Difficulties**

I was not able to implement all the possible data formats that exist

Building the module that calculate the max position using the gaussian fit proved difficult but after research and numerous attempts I was able to implement something.

# **References**

* Google
* Artificial intelligence
* Friends
* YouTube