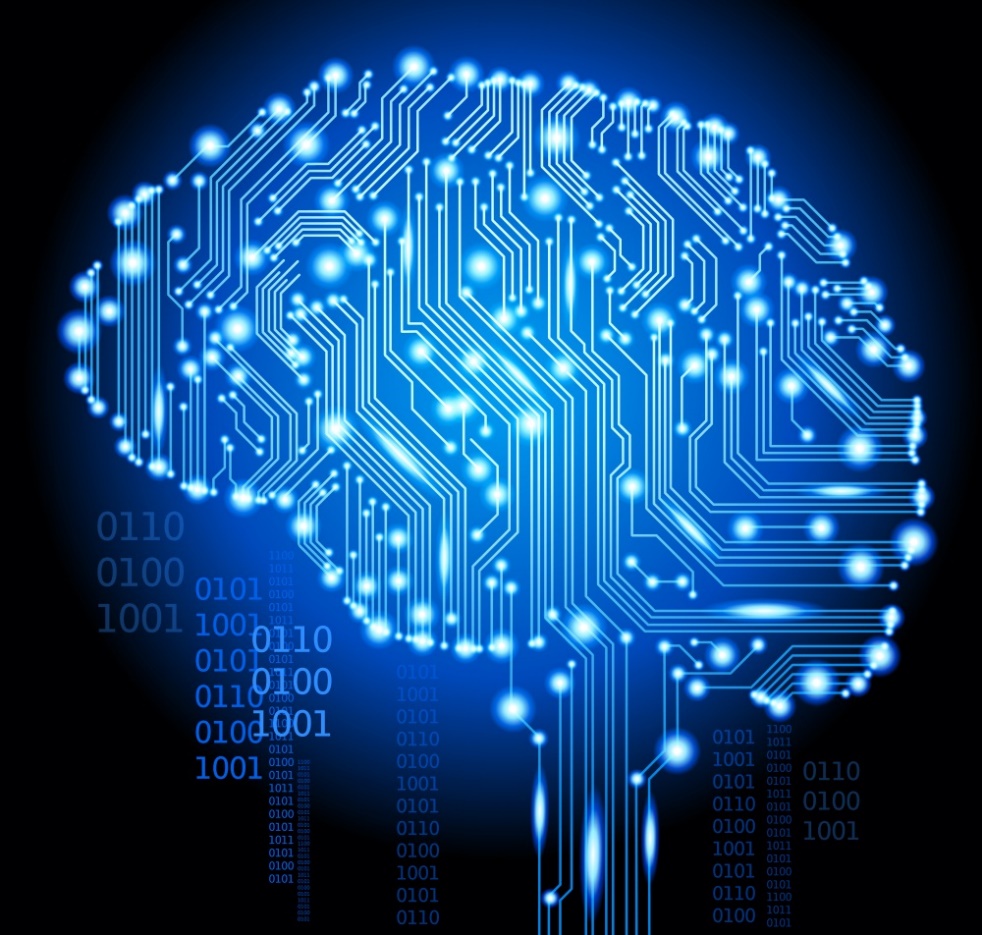


COS30018 – Intelligent Systems

**REPORT**

TASK B3 – DATA PROCESSING 2



**TABLE OF CONTENTS**

*Table of contents…………………………………………………………………………………………………………………2*

*I. Visualizing data solution…………………………………………………………………………………………………..3*

*1. Candletick Chart Display…………………………………………………………………………………………3*

*2. Boxplot Chart Display…………………………………………………………………………………………….6*

*II. Testing…………………………………………………………………………………………………………………………….8*

*1. Test candlestick chart…………………………………………………………………………………………….8*

*2. Test boxplot chart…………………………………………………………………………………………………10*

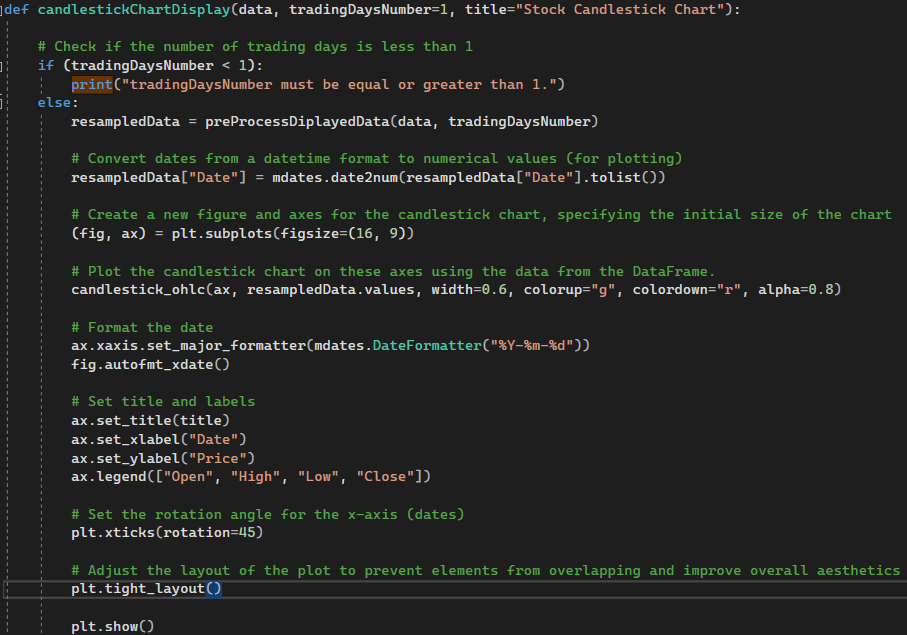
*III. References……………………………………………………………………………………………………………………12*

**VISUALIZING DATA SOLUTION**

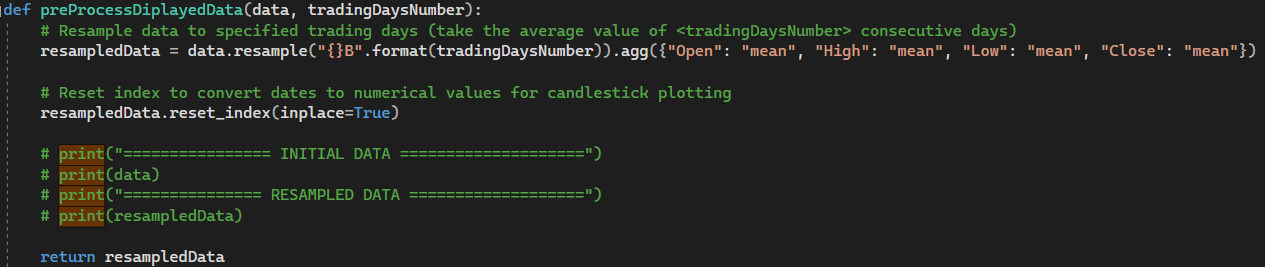
For this processing task, I have created a new file of “dataVisualizing.py”, with the methods “candlestickChartDisplay()” and “boxplotChartDisplay()”, creating the two different ways of data illustration.

**1. Candlestick Chart Display**

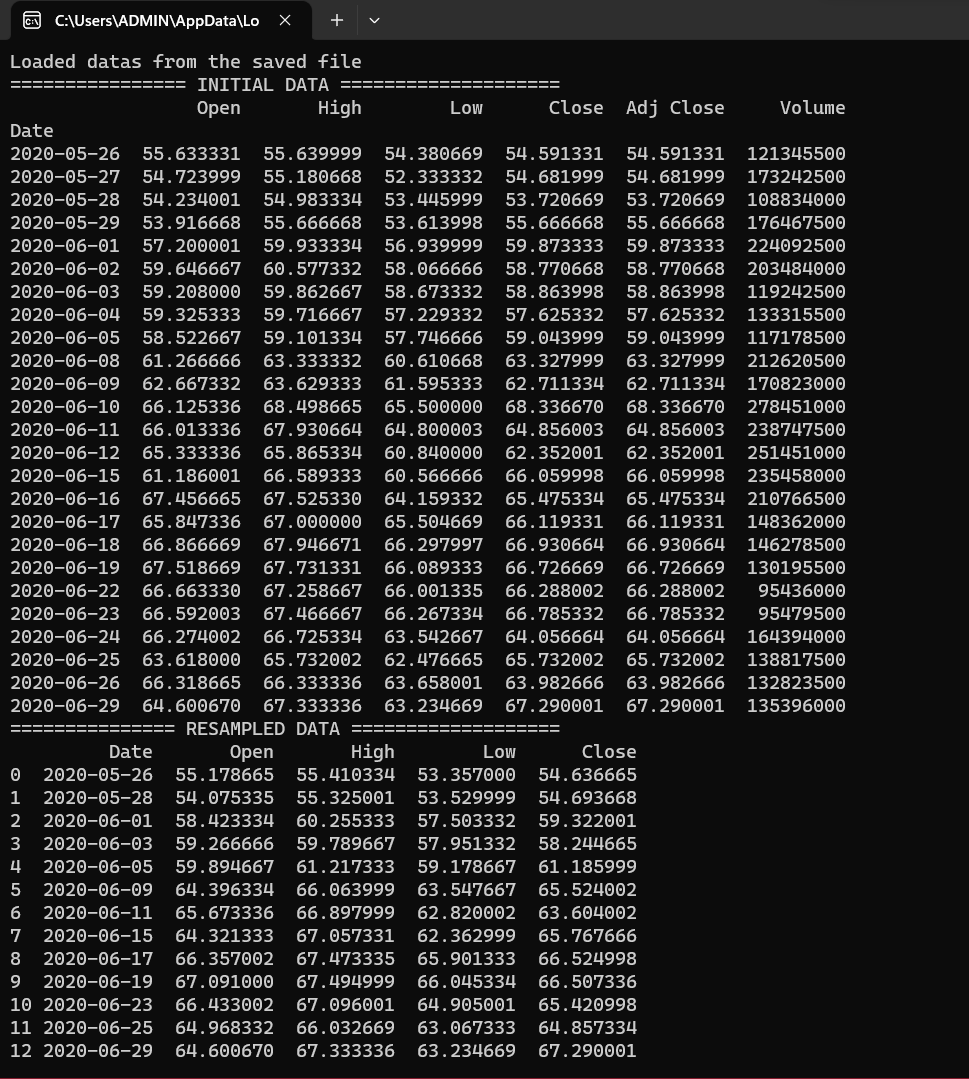
As I have mentioned, the method “candlestickChartDisplay()” is responsible for visualizing the data as a candlestick chart. It takes three parameters, “data” is the DataFrame, “tradingDaysNumber” is the number of consecutive days that a candle represents, and “title” is the tile of the chart:



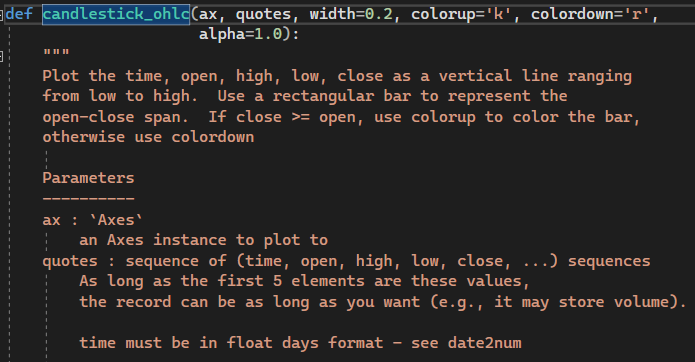
After making sure “tradingDaysNumber” (n) is greater than or equal to 1, I have resampled the data by another method of “preProcessDisplayedData()”:



For this resampled data, only Open, High, Low and Close stock prices are considered. The data will be resampled in the following way: The total number of days of the DataFrame will be reduced by n times, and the average value of 5 consecutive days will be assigned to a column, with the index being the first of those 5 days, as below:



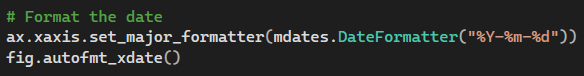
After that, the “Dates” column in the DataFrame is converted to numerical values for plotting, as the requirements of the parameter “quotes” in the method “candlestick\_ohlc()”:



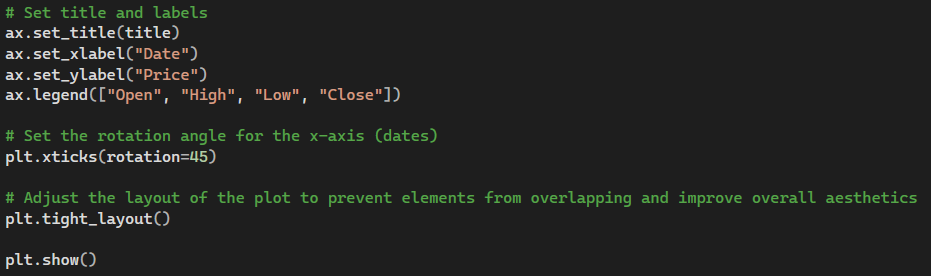
With the method “subplots()” of the module “matplotlib.pyplot”, the new figure (fig) and axes (ax) are created, and the “figsize” parameter specifies the initial dimensions of the figure (16 inches width and 9 inches height).

Also, using the method “candlestick\_ohlc()” creates a candlestick chart on the specified axes using the resampled and processed stock data, setting the bar colors for upward and downward movements of each candle (green: the close price is higher than open, red: the open price is higher) and adjusting the width and transparency of the bars.

To ensure that the dates on the x-axis of the plot are displayed in the "YYYY-MM-DD" format, I have used:

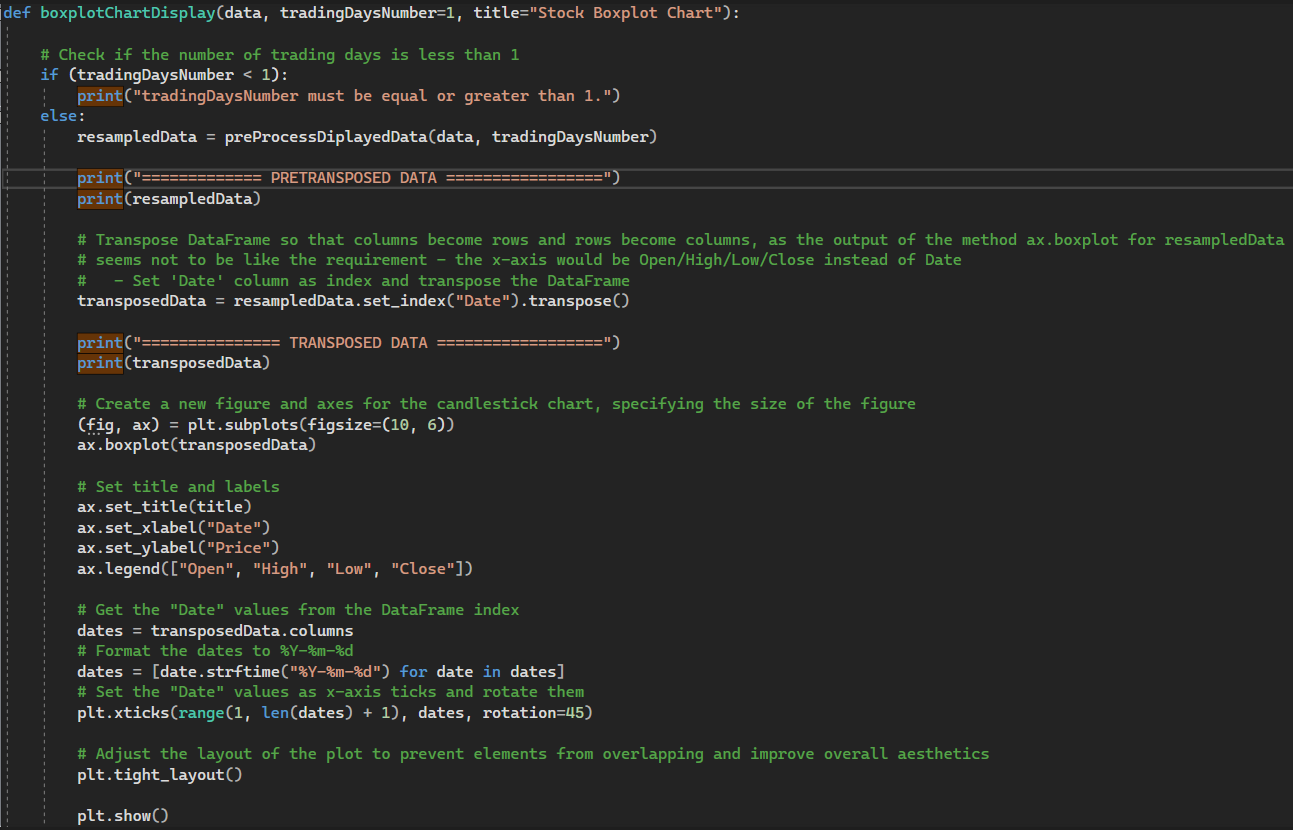


Finally, the method sets the annotation for the chart, including the title, and labels, and adjusts the layout to prevent elements from overlapping, before launching the plot:



**2. Boxplot Chart Display**

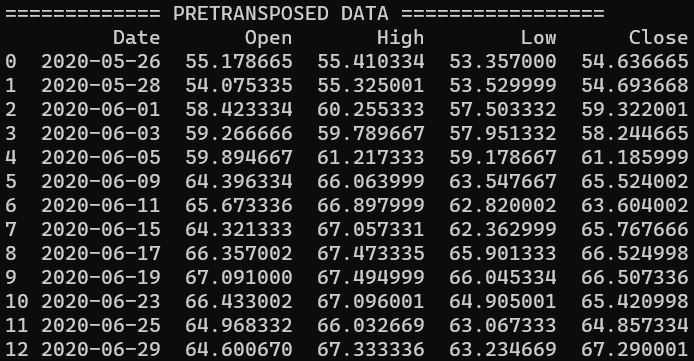
The method “boxplotChartDisplay()” takes the responsibility for plotting the boxplot chart. There are three parameters in this method with similar functionality to the “candlestickChartDisplay()” one.



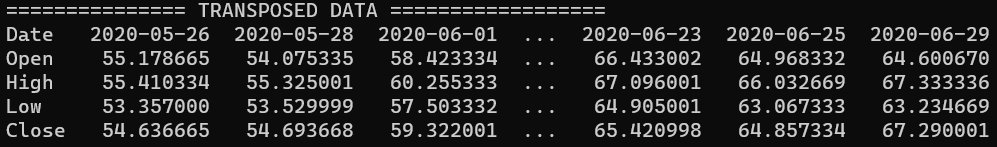
Initially, to express the data of n consecutive trading days, I used the same “preProcessDisplayedData()” method as mentioned above.

As using the “matplotlib.axes.Axes.boxplot” method directly for the DataFrame will give the false result (there would only be 4 boxes representing Open, High, Low and Close stock prices in the range of time), I have to transpose the data (Columns becomes rows and vice versa):

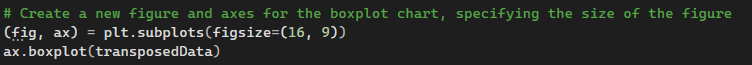
* Before transposing:



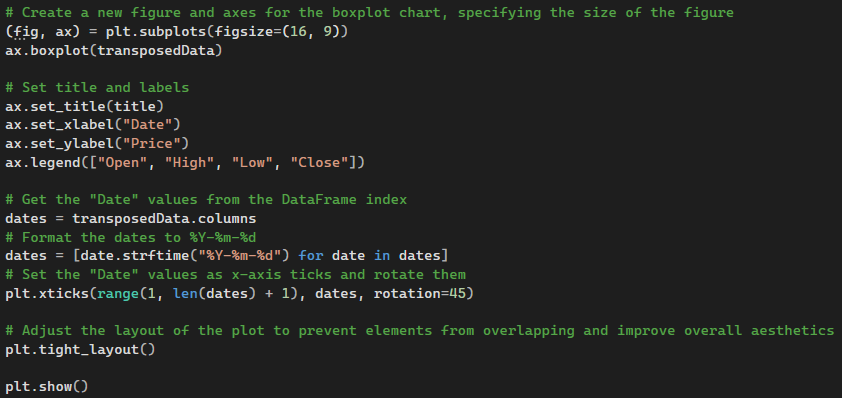
* After transposing:



Next, the new figure and axes for the boxplot chart are created, specifying the size of the figure of 16x9 (inches), before applying the mentioned“boxplot()” method for the “transposedData”:

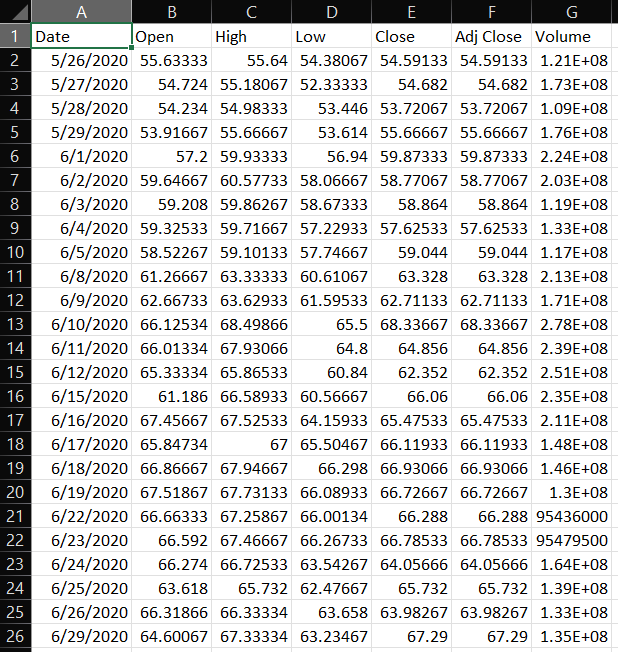


Lastly, the prelaunching and launching processes are basically similar to the “candlestickChartDisplay()” method, except the stage of setting the “Dates” values as x-axis ticks, as if not, it would be integer indices (0, 1, 2, 3, …) by default:



**TESTING**

For testing the chart, I have used the test data of Tesla (TSLA) from 5/26/2020 to 6/29/2020 (26 days)

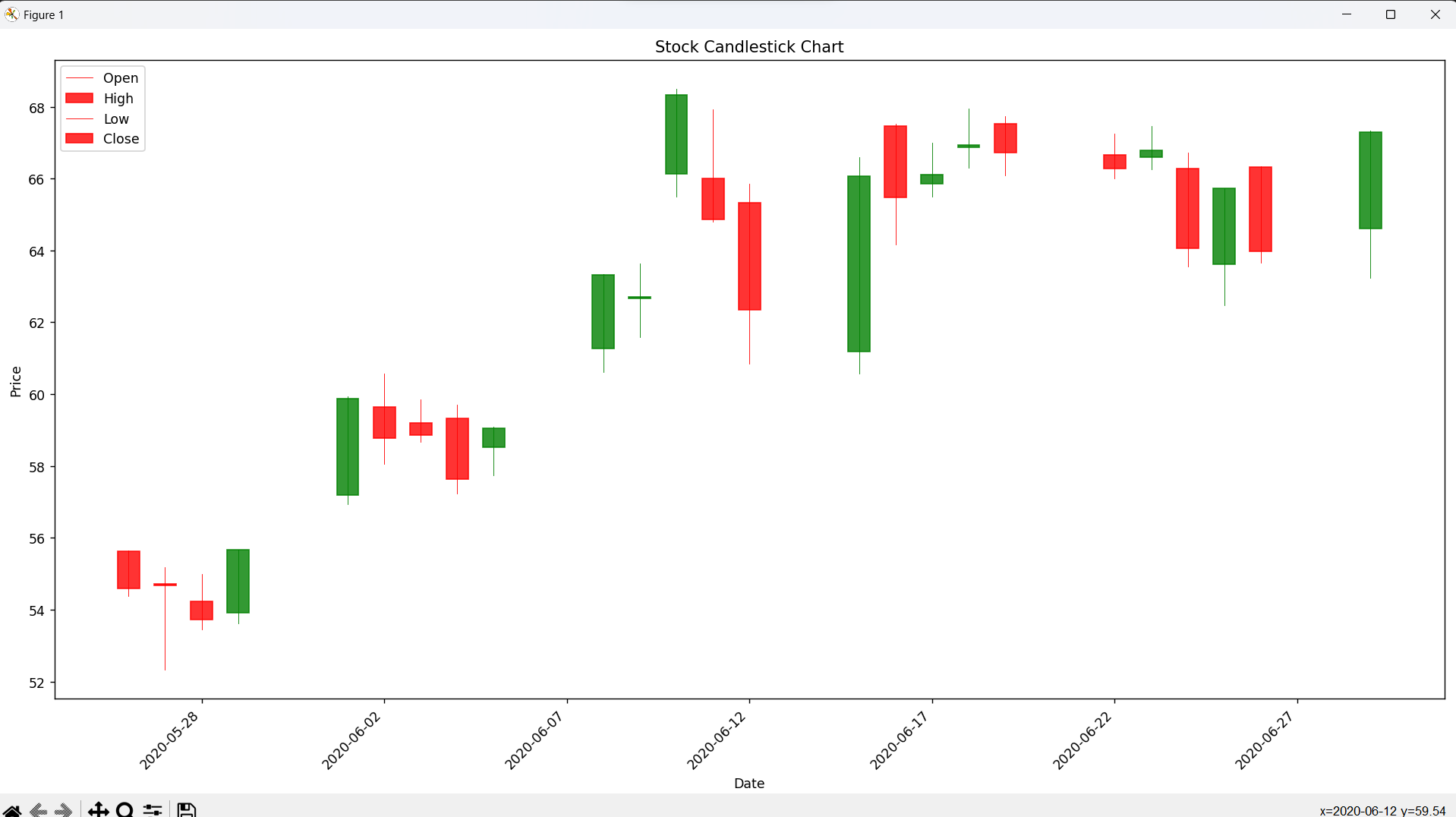


**1. Test candlestick chart**

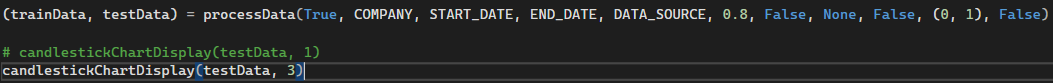
* First, I have tested the method with the parameter “tradingDaysNumber” (n) of 1:



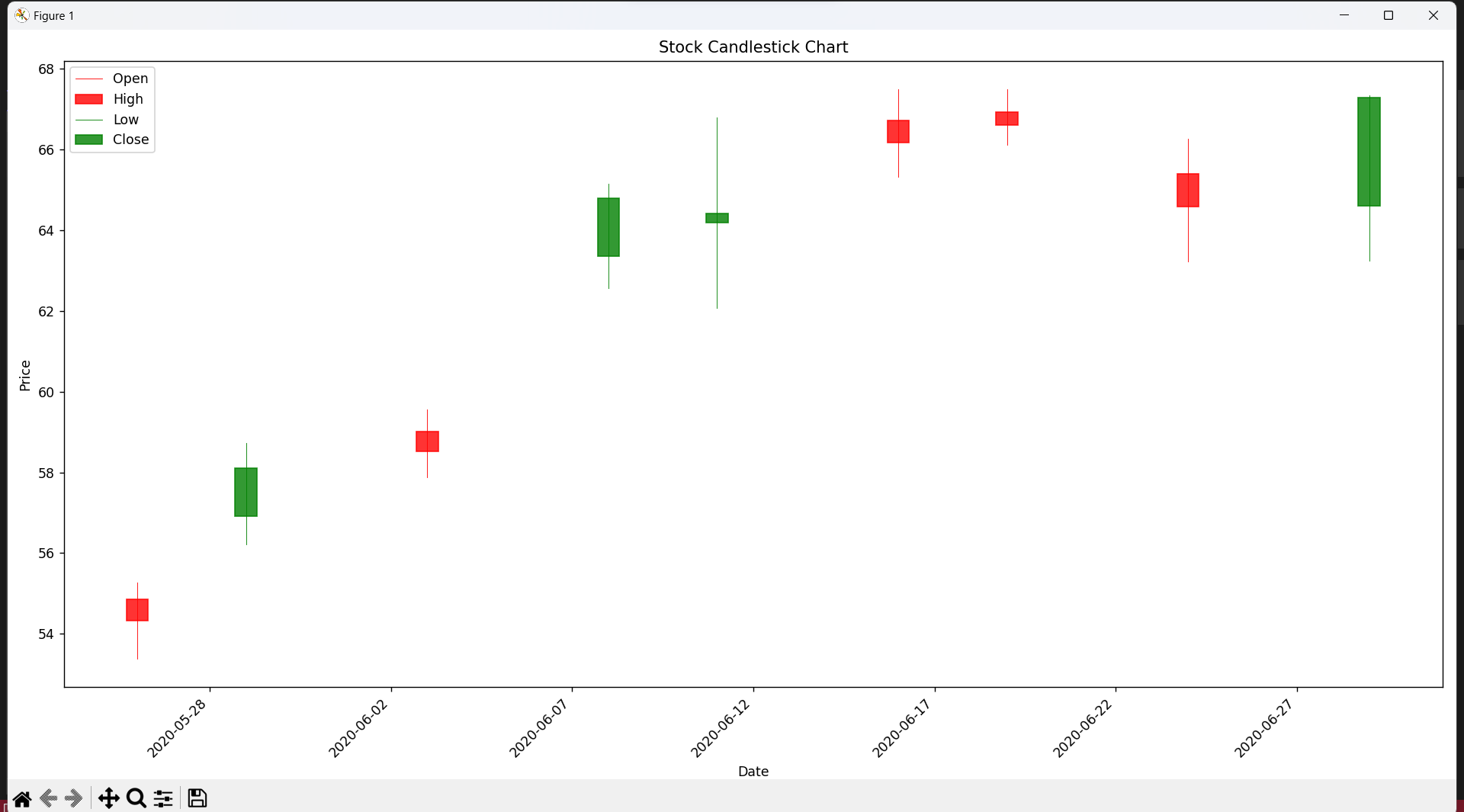
Result:



* Second, n = 3:

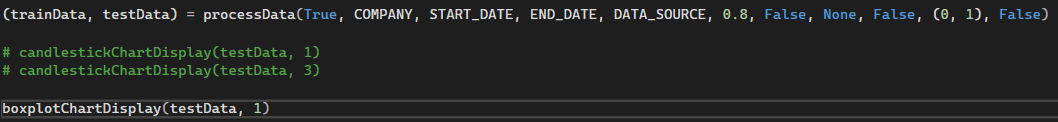


Result:

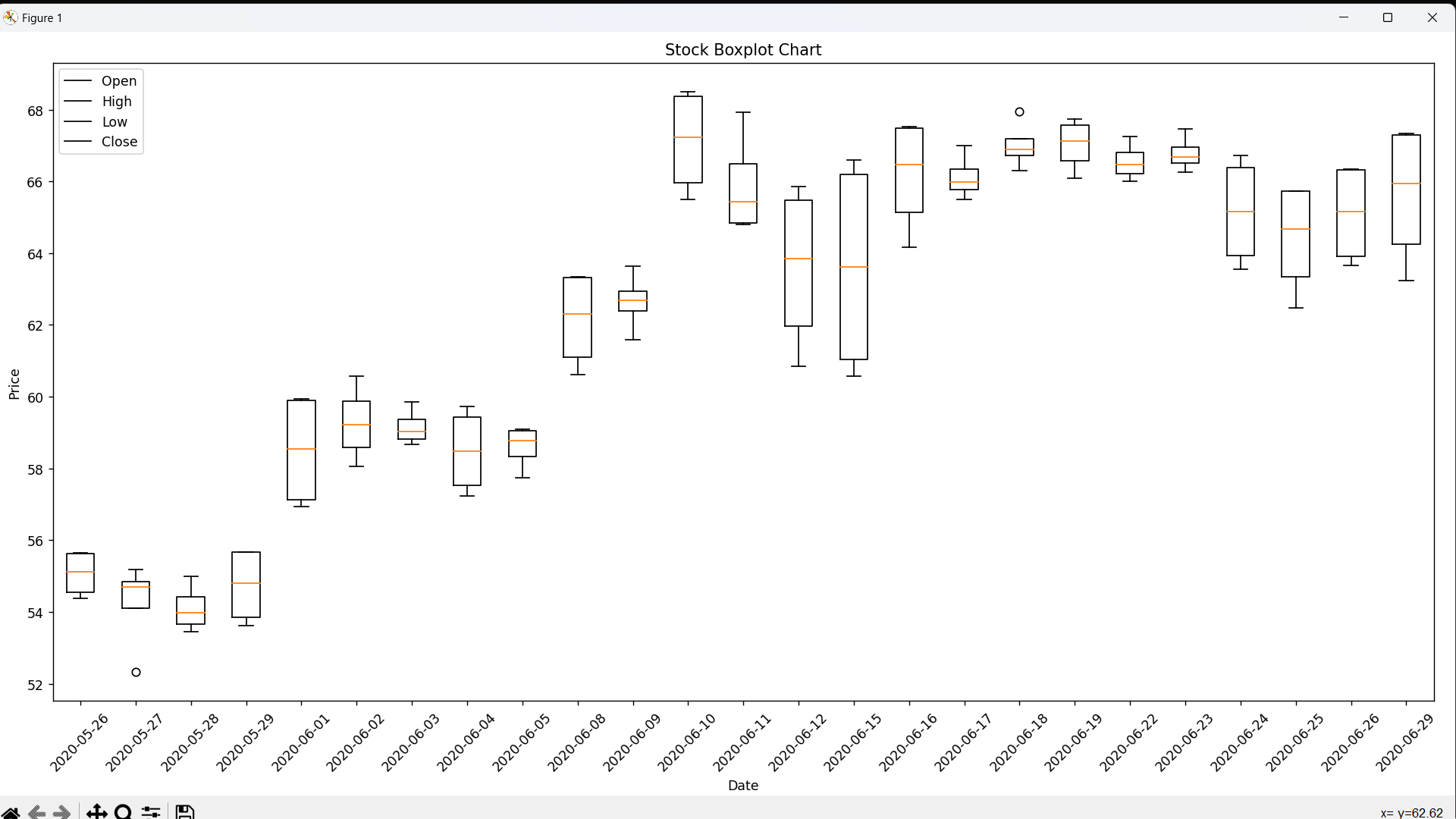


**2. Test boxplot chart**

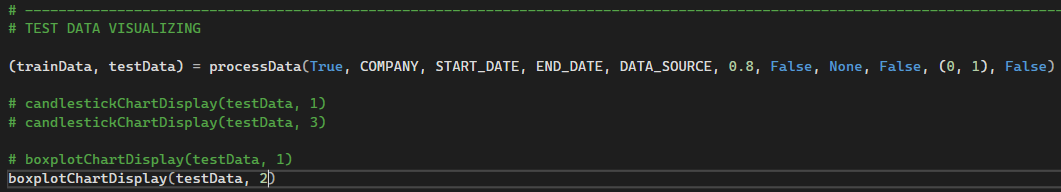
* Running “boxplotChartDisplay()” with “tradingDaysNumber” = 1:



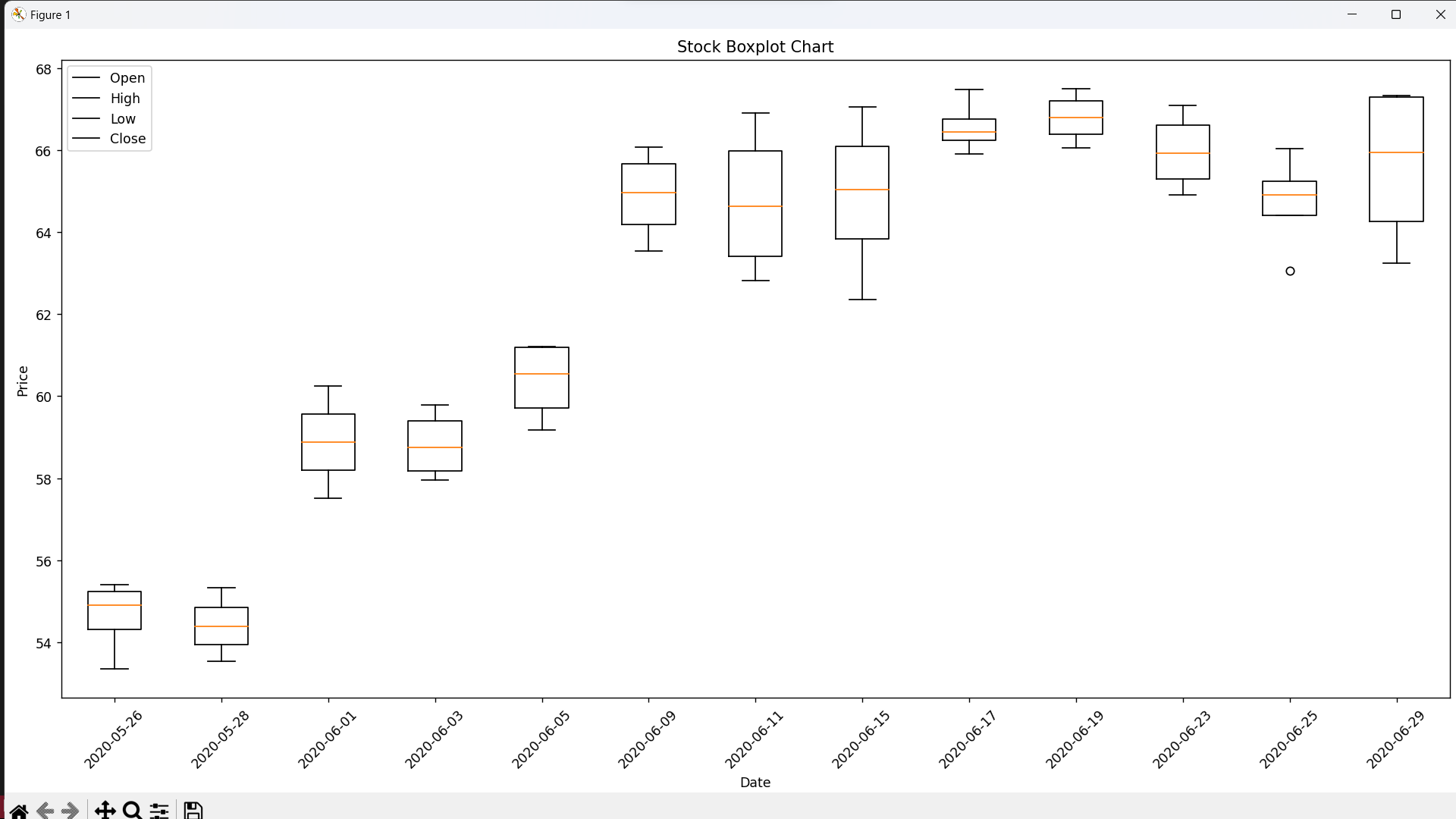
Result:



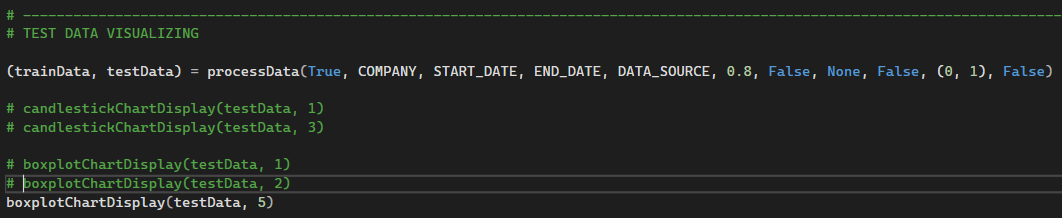
* With n = 2:



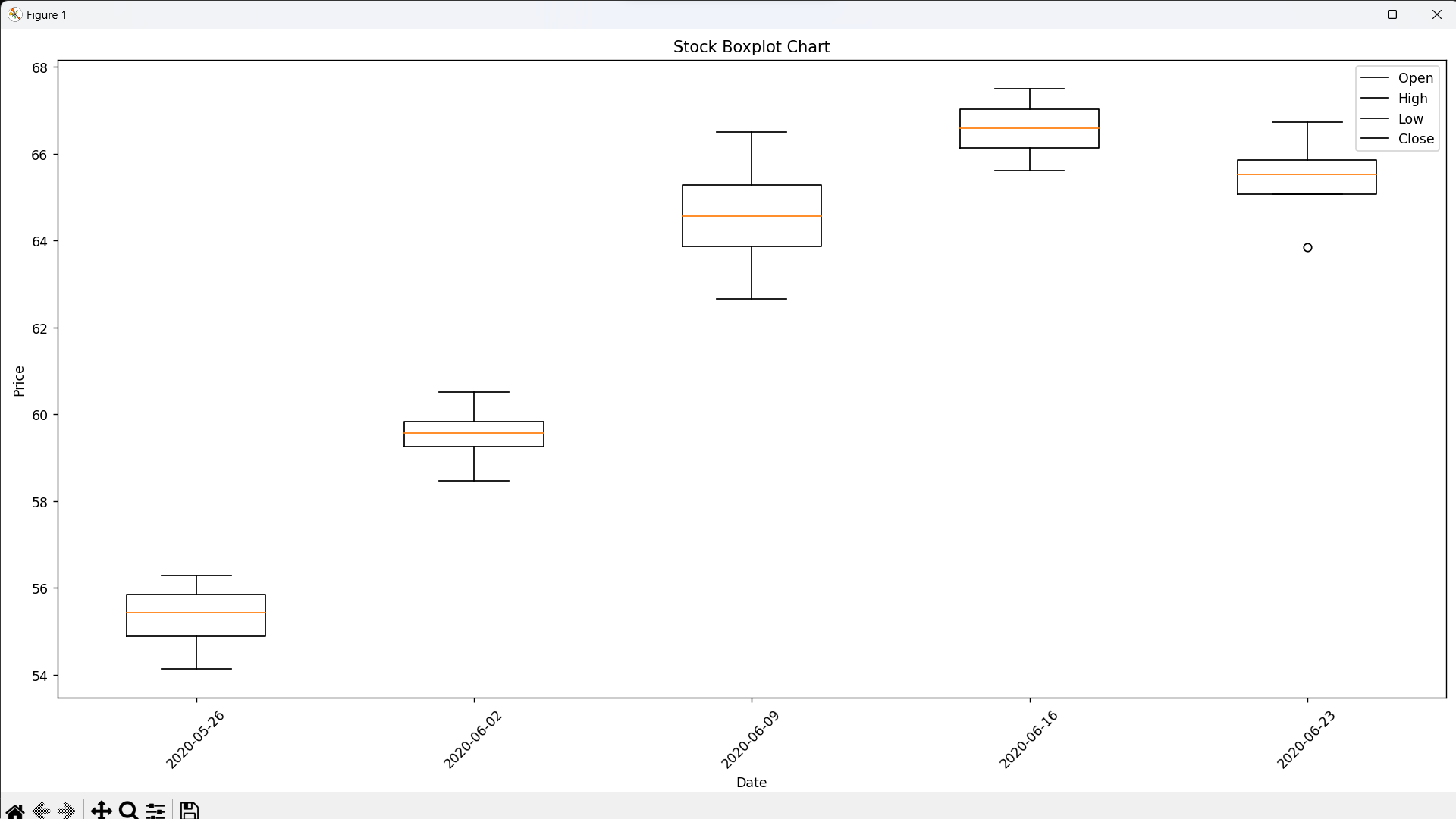
Result:



* With n = 5:



Result:



**REFERENCES**

[1] “Candlestick Chart in Python (mplfinance, plotly, bokeh, bqplot & cufflinks) by Sunny Solanki,” coderzcolumn.com. <https://coderzcolumn.com/tutorials/data-science/candlestick-chart-in-python-mplfinance-plotly-bokeh>

[2] “Python Programming Tutorials - Candlestick OHLC graphs with Matplotlib” pythonprogramming.net. <https://pythonprogramming.net/candlestick-ohlc-graph-matplotlib-tutorial/>

[3] “pandas.DataFrame.resample — pandas 1.5.1 documentation,” pandas.pydata.org. <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.resample.html>

[4] “Box plot chart (stock),” DataClarity, Mar. 23, 2022. <https://support.dataclaritycorp.com/hc/en-us/articles/4408961673364-Box-plot-chart-stock-#:~:text=A%20box%20plot%20chart%20shows>

[5] “matplotlib.axes.Axes.boxplot — Matplotlib 3.1.2 documentation,” matplotlib.org. <https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.axes.Axes.boxplot.html>

[6] “pandas.DataFrame.transpose — pandas 1.4.2 documentation,” pandas.pydata.org. <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.transpose.html>