PROJECT REPORT

Subject: Data Structures & Algorithms

Instructor: Maaz Ahmed

By:

Muhammad Abdullah (2206-2021)

Muhammad Raza (2207-2021)

BSCS-IV (Sec A)/Sp-23

Faculty of Engineering Science and Technology

Hamdard University, Karachi

1. Introduction

1.1 Title

DS TOOL: A Data Science Tool for Visualizing and Analyzing Stock Market Data

1.2 Project Scope

DS Tool is a comprehensive data science tool designed, by using data structures, to assist in thevisualization and analysis of stock market data. The tool provides functionalities for visualizingdata through sorting algorithms, searching for specific elements using linear search, and calculating essential statistical measures such as mean and median. Its primary objective is to empower the user in extracting valuable insights and making informed decisions based on the analysis of stock market trends and key statistical measures.

2. System Features

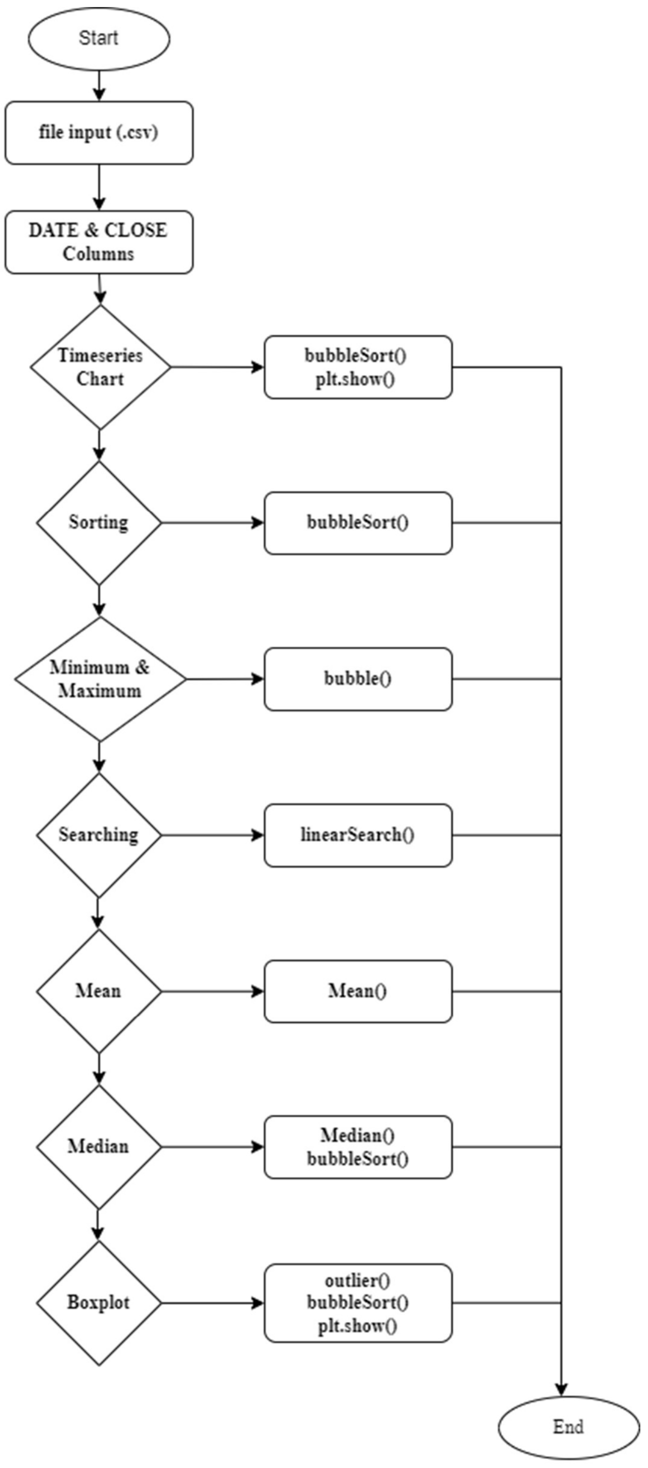
2.1 Description and Priority

DS Tool is a powerful data science tool specifically tailored for professionals in the finance industry. The tool offers a wide range of features to facilitate the visualization and analyzing stock market data, with a particular focus on sorting, searching and statistical calculations. The sorting feature enable users to explore the stock market dataset by sorting the data of closing. By organizing the data in sorted order, users can identify the trends, patterns and outliers within the dataset. The searching feature provides a quick and efficient way to find relevant information within the dataset for further analysis and decision-making. Furthermore, essential statistical calculations, including mean and median, provide insights into the central tendencies of the stock market data. These statistical measures provide a deeper understanding of overall behavior and distribution of the stock market data. By utilizing the functionalities, users can gain valuable insights, make data driven decisions, and enhance their understandingof the dynamics of stock market.

2.2 Functional Requirements

The tool expects a .csv file as an input which will be placed in the same directory where the solution file is present. The solution reads the columns labelled as "DATE" and "CLOSE", which will be used to carried out the calculations. The solution is limited foronly a few values of both columns.

2.3 Flowchart



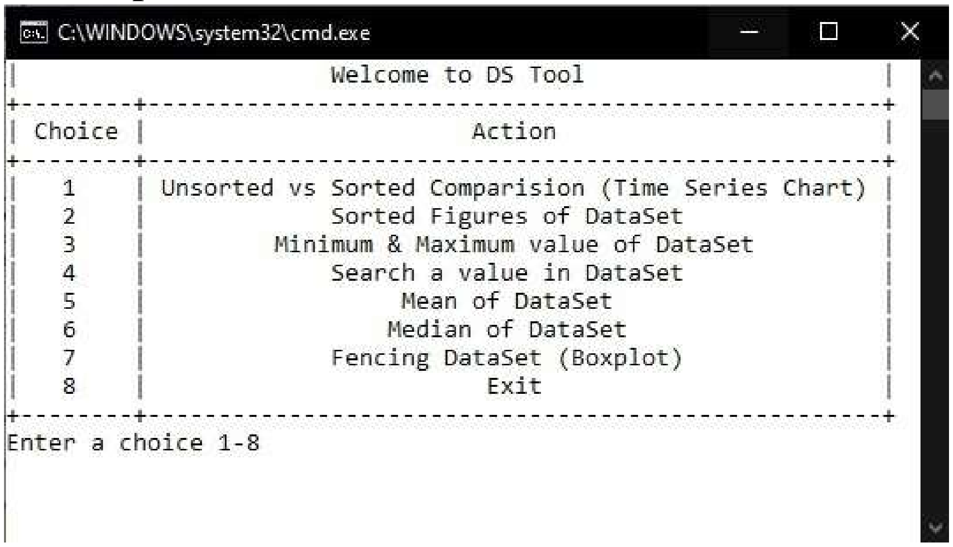
DS TOOL FLOWCHART

3. Source Code

```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
from statistics import mean
import nuampy as np
from datetime import datetime
from prettytable import PrettyTable


data = pd.read_csv('data1.csv')
price_date = data['Date']
price_close = data['Close']

def bubbleSort(array):
    for j in range(0, len(array) - i - 1):
      if array[j] > array[j + 1]:
        temp = array[j]
        array[j] = array[j+1]
        array[j+1] = temp

def bubble(array):
    bubbleSort(array)
    print('Minimum=',array[0])
    print('Maximum=',array[len(array)-1])


def linearSearch(array):
    key=float(input("Enter a Precised Value to Search"))
    for x in range(len(array)):
        if array[x]==key:
            place=x+1
            print(place)
            break
    else:
        print("Not Found")

def Mean(array):
    a=mean(array)
    print(round(a,1))

def outlier(arry):
   bubbleSort(arry)
   q1, q3= np.percentile(arry,[25,75])
   iqr = q3 - q1
   lower_bound = q1 -(1.5 * iqr)
   upper_bound = q3 +(1.5 * iqr)
   # Create a larger figure and axis
   fig, ax = plt.subplots(figsize=(6, 5))

   # Create the boxplot horizontally
   ax.boxplot(arry, vert=False)

   # Set the x-axis limits using the lower_bound and upper_bound
   ax.set_xlim(lower_bound - 1, upper_bound + 1)

   # Plot lower_bound and upper_bound as dots
   ax.plot(lower_bound, 1, 'ro', label=f'Lower Bound: {lower_bound}')
   ax.plot(upper_bound, 1, 'go', label=f'Upper Bound: {upper_bound}')
   ax.plot(q1, 1, 'bo', label=f'First Quartile: {q1}')
   ax.plot(q3, 1, 'yo', label=f'Third Quartile: {q3}')
   ax.set_title('Anything before Lower and after Upper Bounds is Outlier')

   # Display legend
   ax.legend()

   plt.show()


def Median(array):
    bubbleSort(array)
    n=len(array)
    print(array)
    if n%2==0:
        f=n//2
        s=(n+2)//2
        a=(array[f]+array[s])/2
        print('Median=',a)
    else:
        place=(n+1)/2
        a=array[place]
        print('Median=',a)
    return a


def comparision(price_date,price_close):
    fig = plt.figure()

    # Subplot 1
    plt.subplot(2, 2, 3)
    plt.plot_date(mdates.date2num(price_date), price_close, linestyle='solid')
    plt.gca().xaxis.set_major_locator(mdates.DayLocator(interval=1))
    plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%d'))

    bubbleSort(price_close)

    # Subplot 2
    plt.subplot(2, 2, 4)
    plt.plot_date(mdates.date2num(price_date), price_close, linestyle='solid')
    plt.gca().xaxis.set_major_locator(mdates.DayLocator(interval=1))
    plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%d'))

    # Adjust the spacing between subplots
    plt.tight_layout()

    # Add the month and year in the title
    title_date = datetime.strptime(price_date[0], '%Y-%m-%d').strftime('%B
%Y')
    fig.suptitle(f'Comparision between Unsorted and Sorted Figures for
({title_date})')


    plt.show()

columns = ["Choice", "Action"]

myTable = PrettyTable()
myTable.add_column(columns[0], ["1", "2", "3","4", "5", "6", "7", "8"])
myTable.add_column(columns[1], ["Unsorted vs Sorted Comparision (Time Series
Chart)", "Sorted Figures of DataSet", "Minimum & Maximum value of DataSet",
"Search a value in DataSet", "Mean of DataSet", "Median of DataSet", "Fencing
DataSet (Boxplot)","Exit"])
myTable.title = "Welcome to DS Tool"
print(myTable)
```

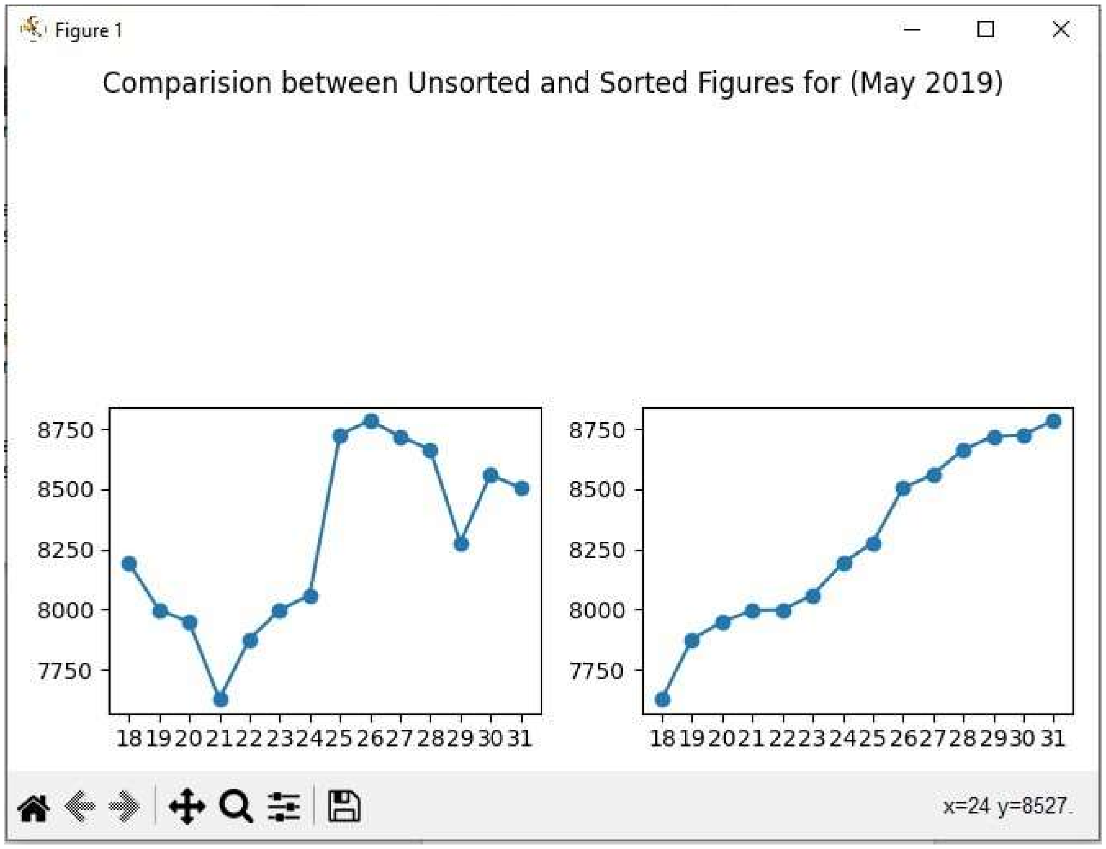
```
#Driver code
while True:
    choice=input("Enter a choice 1-8\n")
    match choice:
        case "1":
            comparision(price_date,price_close)
        case "2":
            bubbleSort(price_close)
            print(price_close)
        case "3":
            bubble(price_close)
        case "4":
            linearSearch(price_close)
        case "5":
            Mean(price_close)
        case "6":
            Median(price_close)
        case "7":
            outlier(price_close)
    if choice == "8":
        print("\nThank you for using DS Tool,\nKeep Analyzing!!!")
        break
```

4. Outputs

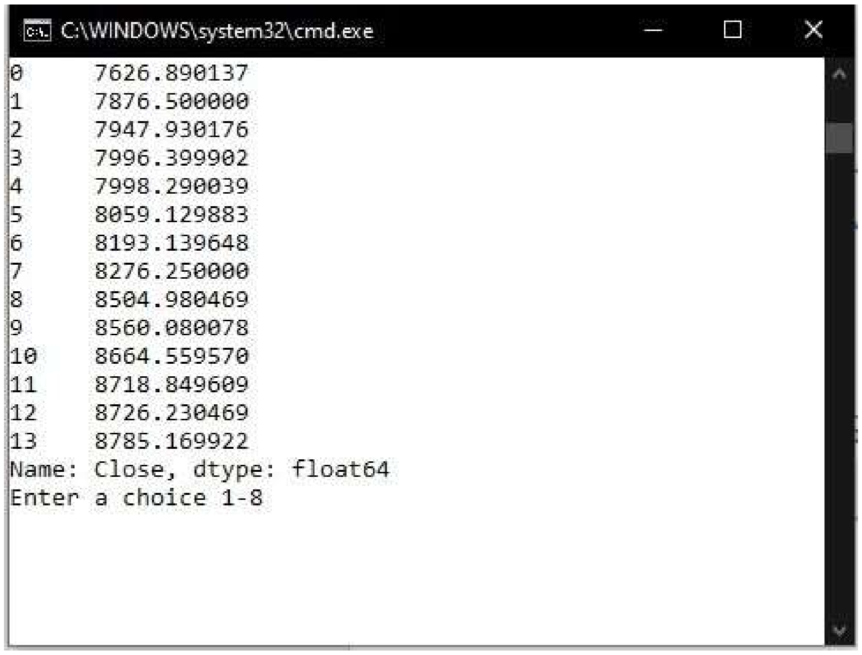


```
C:\WINDOWS\system32\cmd.exe
|                         Welcome to DS Tool                          |
+--------+------------------------------------------------------------+
| Choice |                           Action                           |
+--------+------------------------------------------------------------+
|   1    | Unsorted vs Sorted Comparision (Time Series Chart)         |
|   2    |            Sorted Figures of DataSet                       |
|   3    |        Minimum & Maximum value of DataSet                  |
|   4    |            Search a value in DataSet                       |
|   5    |                 Mean of DataSet                            |
|   6    |                Median of DataSet                           |
|   7    |            Fencing DataSet (Boxplot)                       |
|   8    |                      Exit                                  |
+--------+------------------------------------------------------------+
Enter a choice 1-8
```

CHOICE 1:

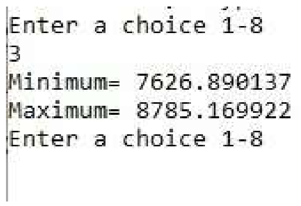


CHOICE 2:



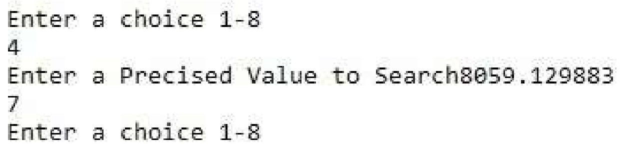
```
C:\WINDOWS\system32\cmd.exe
0      7626.890137
1      7876.500000
2      7947.930176
3      7996.399902
4      7998.290039
5      8059.129883
6      8193.139648
7      8276.250000
8      8504.980469
9      8560.080078
10     8664.559570
11     8718.849609
12     8726.230469
13     8785.169922
Name: Close, dtype: float64
Enter a choice 1-8
```

CHOICE 3:



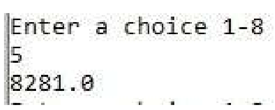
```
Enter a choice 1-8
3
Minimum= 7626.890137
Maximum= 8785.169922
Enter a choice 1-8
```

CHOICE 4:

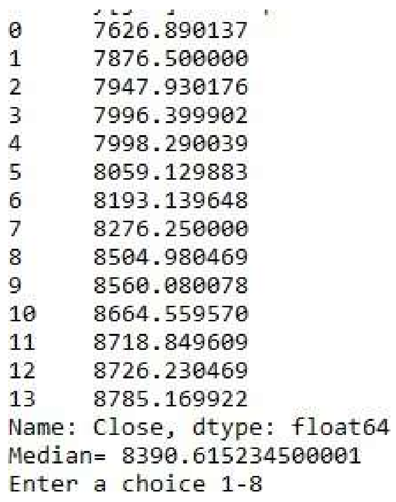


```
Enter a choice 1-8
4
Enter a Precised Value to Search8059.129883
7
Enter a choice 1-8
```

CHOICE 5 & 6:

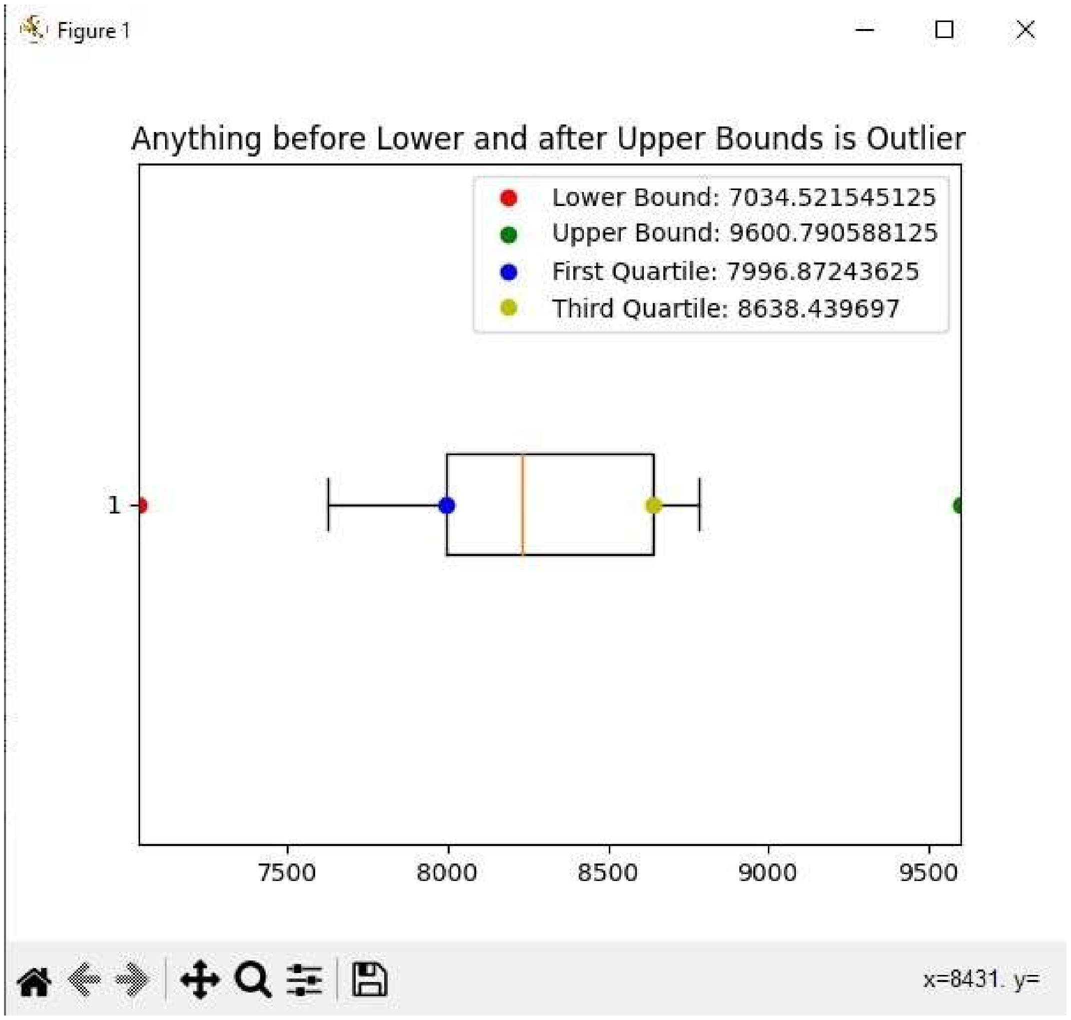


```
Enter a choice 1-8
5
8281.0
```

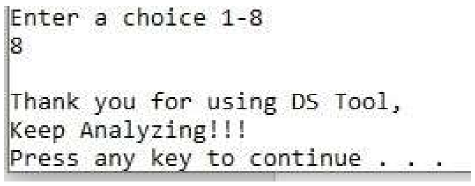


```
0     7626.890137
1     7876.500000
2     7947.930176
3     7996.399902
4     7998.290039
5     8059.129883
6     8193.139648
7     8276.250000
8     8504.980469
9     8560.080078
10    8664.559570
11    8718.849609
12    8726.230469
13    8785.169922
Name: Close, dtype: float64
Median= 8390.615234500001
Enter a choice 1-8
```

CHOICE 7:



CHOICE 8:



```
Enter a choice 1-8
8

Thank you for using DS Tool,
Keep Analyzing!!!
Press any key to continue . . .
```