S01 - Import and Plot data for one stock

November 11, 2023

1 #001 Import and Plot data for one stock:

1.1 Define and import data, save as CSV

Import libraries

- Link to Pandas Documentation: https://pandas.pydata.org/docs/index.html
- Link to datareader Documentation: https://pandas-datareader.readthedocs.io/en/latest/index.html
- Link to Yf Documentation: https://pandas-datareader.readthedocs.io/en/latest/readers/yahoo.html
- Link to Plotly Documentation: https://plotly.com/python/

[1]:
```
#    !pip install pandas
#    !pip install pandas-datareader
#    !pip install yfinance
#    !pip install datetime
#    !pip install plotly_express
```

[2]:
```
import pandas as pd
from pandas_datareader import data as pdr
import numpy as np
import yfinance as yf
import datetime as dt
import plotly.express as px
import plotly.graph_objects as go
```

[3]:
```
# Define the start and end dates, last 5 years
end = dt.datetime.now()
start = end - dt.timedelta(days = 365*5)
```

[4]:
```
# define Tickers
tk = input('Enter the ticker code: ')
tickers = [tk]
file_name = tk + ".csv"

yf.pdr_override()
```

```
Enter the ticker code: MSFT
```

[5]:
```
#obtain data and save as CSV
df = pdr.get_data_yahoo(tickers, start = start, end = end)
df.to_csv(file_name)
df.head(2)
```

```
[*********************100%***********************]  1 of 1 completed
```

[5]:
```
                  Open        High         Low       Close   Adj Close  \
Date
2018-11-12  109.419998  109.959999  106.099998  106.870003  101.251991
2018-11-13  107.550003  108.739998  106.639999  106.940002  101.318344

              Volume
Date
2018-11-12  33621800
2018-11-13  35374600
```

1.2 Import CSV and test values

[6]:
```
#read CSV file
stock_df = pd.read_csv(file_name)
```

[7]:
```
# Test for null values, and show info
stock_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1258 entries, 0 to 1257
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       1258 non-null   object
 1   Open       1258 non-null   float64
 2   High       1258 non-null   float64
 3   Low        1258 non-null   float64
 4   Close      1258 non-null   float64
 5   Adj Close  1258 non-null   float64
 6   Volume     1258 non-null   int64
dtypes: float64(5), int64(1), object(1)
memory usage: 68.9+ KB
```

1.2.1 Calculate Daily Returns from one stock, and insert on DataFrame

[8]:
```
#insert 'Daily Return' column
stock_df['Daily Return'] = stock_df['Adj Close'].pct_change(1) * 100

# replace the first row, changes null for 0
stock_df['Daily Return'].replace(np.nan, 0, inplace = True)
stock_df
```

[8]:
```
            Date        Open        High         Low       Close   Adj Close  \
0     2018-11-12  109.419998  109.959999  106.099998  106.870003  101.251991
1     2018-11-13  107.550003  108.739998  106.639999  106.940002  101.318344
2     2018-11-14  108.099998  108.260002  104.470001  104.970001   99.881516
3     2018-11-15  104.989998  107.800003  103.910004  107.279999  102.079544
4     2018-11-16  107.080002  108.879997  106.800003  108.290001  103.040596
…            …           …           …           …           …           …
1253  2023-11-06  353.450012  357.540009  353.350006  356.529999  356.529999
1254  2023-11-07  359.399994  362.459991  357.630005  360.529999  360.529999
1255  2023-11-08  361.679993  363.869995  360.549988  363.200012  363.200012
1256  2023-11-09  362.299988  364.790009  360.359985  360.690002  360.690002
1257  2023-11-10  361.489990  370.100006  361.070007  369.670013  369.670013

        Volume  Daily Return
0     33621800      0.000000
1     35374600      0.065532
2     39495100     -1.418133
3     38505200      2.200636
4     33502100      0.941474
…          …             …
1253  23828300      1.057259
1254  25833900      1.121925
1255  26767800      0.740580
1256  24847300     -0.691082
1257  28042100      2.489676

[1258 rows x 8 columns]
```

[9]:
```
stock_df.describe().round(2)
```

[9]:
```
          Open     High      Low    Close  Adj Close        Volume  \
count  1258.00  1258.00  1258.00  1258.00    1258.00  1.258000e+03
mean    229.14   231.57   226.66   229.22     225.05  2.988689e+07
std      71.73    72.40    71.04    71.73      72.76  1.269096e+07
min      95.14    97.97    93.96    94.13      89.57  8.989200e+06
25%     162.03   163.63   160.41   162.10     156.59  2.185582e+07
50%     238.38   242.36   235.74   239.61     236.64  2.669715e+07
75%     287.82   289.95   284.04   287.69     283.81  3.387148e+07
max     362.30   370.10   361.07   369.67     369.67  1.112421e+08

       Daily Return
count       1258.00
mean           0.12
std            1.95
min          -14.74
25%           -0.85
50%            0.12
```

```
75%              1.12
max             14.22
```

1.3 Ploting Results

Used Plotly Express for data visualization

[10]:
```
# Define a function that performs interactive data visualization using Plotly␣
↪Express
def plotly_data(df, title):

    # Create figure
    fig = go.Figure()

    # Set title
    fig.update_layout(title_text = title)

    # For loop that plots all stock prices in the pandas dataframe df
    # starts with 1, to skip the date column

    for i in df.columns[1:]:
        # Add range slider
        fig.update_layout(xaxis=dict(rangeselector =␣
↪dict(buttons=list([dict(count=1, label="1m", step="month",␣
↪stepmode="backward"), dict(count=6, label="6m", step="month",␣
↪stepmode="backward"), dict(count=1, label="YTD", step="year",␣
↪stepmode="todate"), dict(count=1, label="1y", step="year",␣
↪stepmode="backward"), dict(step="all")])), rangeslider=dict( visible=True),␣
↪type="date"))
        # Add line graph
        fig.add_scatter(x = df['Date'], y = df[i], name = i)
        # Update Layout
        fig.update_layout({'plot_bgcolor': "white"})
    fig.show()
```

[11]:
```
# Ploting AdjClose
plotly_data(stock_df.iloc[:, [0 , 5]], (tk + ' - Ajusted Closing Price[$]'))

# Ploting Trade Volume
plotly_data(stock_df.iloc[:, [0 , 6]], (tk +' - Trading Volume'))

# Ploting Percentage Daily Return
plotly_data(stock_df.iloc[:, [0 , 7]], (tk +' - Percentage Daily Return [%]'))
```

[ ]:
```

```