Name: Manas Kapoor

Email address: manaskapoor3000@gmail.com

Phone No. +91-9810300253

University: Delhi Technological University

Scalar Field-Assignment Submission

Question 1 Overview:

Write a Python function to calculate the trade performance of a given set of trades executed by a financial firm. The function should take a pandas DataFrame as input with the following columns:

1. Date (datetime64[ns]): The date and time of the trade
2. Symbol (string): The ticker symbol of the traded security
3. Side (string): Either 'buy' or 'sell'
4. Size (float, optional): The number of shares traded (default to 1 if not provided)
5. Price (float): The price at which the trade was executed

Your task is to:

1. Choose 10 financial metrics that you believe are the most important for tracking trade performance of a strategy.
2. Write a function that takes in this DataFrame and computes these 10 metrics.
3. Account for both long and short strategies in your calculations.
4. Explain how you compute returns in the case of long-short strategies.

Solution:

Data-preprocessing:

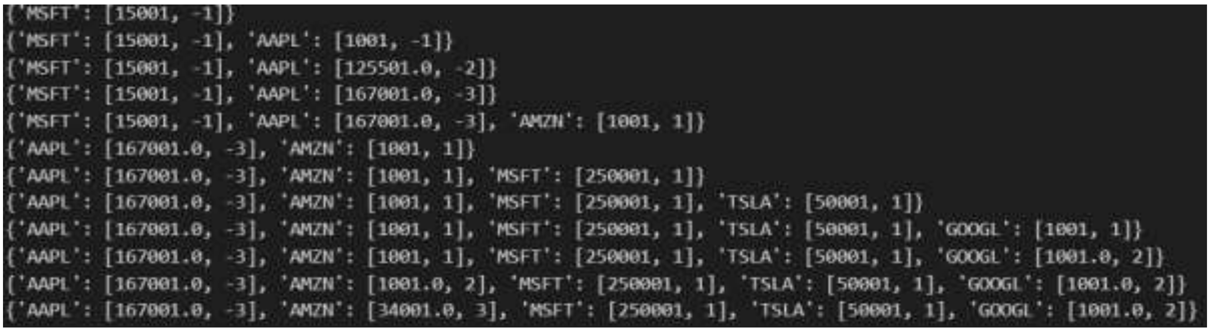
1. **Column removal:**
   Irrelevant columns present in the DataFrame like: 'disclosureYear','disclosureDate','owner','district','representative','capitalGainsOver200USD','option_symbol','assetDescription' were removed from the dataset
2. **Column Rename:**

The relevant columns were then renamed to the Titles provided in the question

3. **Conversion to Date-time:**
   The Date column was converted to Datetime format. The DataFrame was sorted according to date
4. **Missing value fill:**
   The value of ‘Size’ column was filled as 1, since no data about number of shares was given

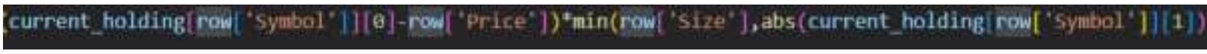
Metric Selection:

1. **Total profit:**
   Time Complexity (O(N))
   - The code maintains a dictionary of current_holding to keep in account of ongoing trades



```
{'MSFT': [15001, -1]}
{'MSFT': [15001, -1], 'AAPL': [1001, -1]}
{'MSFT': [15001, -1], 'AAPL': [125501.0, -2]}
{'MSFT': [15001, -1], 'AAPL': [167001.0, -3]}
{'MSFT': [15001, -1], 'AAPL': [167001.0, -3], 'AMZN': [1001, 1]}
{'AAPL': [167001.0, -3], 'AMZN': [1001, 1]}
{'AAPL': [167001.0, -3], 'AMZN': [1001, 1], 'MSFT': [250001, 1]}
{'AAPL': [167001.0, -3], 'AMZN': [1001, 1], 'MSFT': [250001, 1], 'TSLA': [50001, 1]}
{'AAPL': [167001.0, -3], 'AMZN': [1001, 1], 'MSFT': [250001, 1], 'TSLA': [50001, 1], 'GOOGL': [1001, 1]}
{'AAPL': [167001.0, -3], 'AMZN': [1001, 1], 'MSFT': [250001, 1], 'TSLA': [50001, 1], 'GOOGL': [1001.0, 2]}
{'AAPL': [167001.0, -3], 'AMZN': [1001.0, 2], 'MSFT': [250001, 1], 'TSLA': [50001, 1], 'GOOGL': [1001.0, 2]}
{'AAPL': [167001.0, -3], 'AMZN': [34001.0, 3], 'MSFT': [250001, 1], 'TSLA': [50001, 1], 'GOOGL': [1001.0, 2]}
```

   - This is done in order to not include unsettled trades in profit calculations
   - The code can also provide the number of unsettled trades, with average price of purchase/sell and number of shares
   - If a company symbol is not present in dictionary, it first creates its entry with price and share no. If the trade is of short selling, it accounts the number of trades as negative
   - If the symbol is present in the dictionary and the new entry share same sign as shares present, it edits the price as the weighted average of price and shares.
   - Otherwise, the profit variable adds the value of :



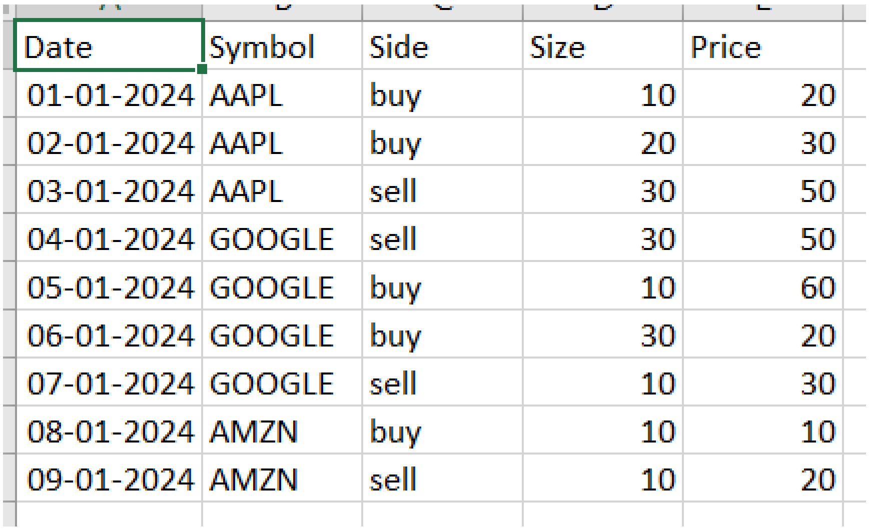
```
(current_holding[row['Symbol']][0]-row['Price'])*min(row['Size'],abs(current_holding[row['Symbol']][1])
```

2. **Drawdown:**
   Time Complexity O(N):
   - Drawdown is referred as the longest streak of loss ever occurred.
   - The function takes in a profit_array in which each element is the profit recorded at that row.
   - By simple logic we can calculate the maximum streak of loss making trades
3. **Total trades:**
   - Calculated by taking into account the total number of rows present in the DataFrame
4. **Win rate:**
   - For every profitable trade a +1 is stored in a variable and for every loss making trade is also recorded
   - Win rate=(win)/(win+loss)
5. **Average Return per trade:**
   - Calculated by taking the ratio of (total profit)/(win+loss)
6. **Average Hold Period:**
   - Average hold period becomes an important metric as it accounts for the duration for which out capital will be blocked
   - This function was implemented by creating a dictionary which stores a QUEUE data structure for each of the symbol.
   - The QUEUE data structure holds the value of (date, share_qty).
   - If the share of the first entry in queue shares same sign as that of the incoming entry, then the new entry is appended in the queue.
   - However, if the signs differ the shares present in queue are one by one squared of with the new entry.
   - For each square of the corresponding date-difference is stored in an array,
   - The average of this array is given as output
7. **Capital Invested:**
   - The maximum capital required to execute such portfolio is essential to calculate.
   - This is done using the same logic in the profit function, by maintaining a current_holding dictionary
   - At the end of processing of each row, the dictionary is traversed to account for the capital involved in holding
   - Maximum of this capital is regarded as the Capital invested.

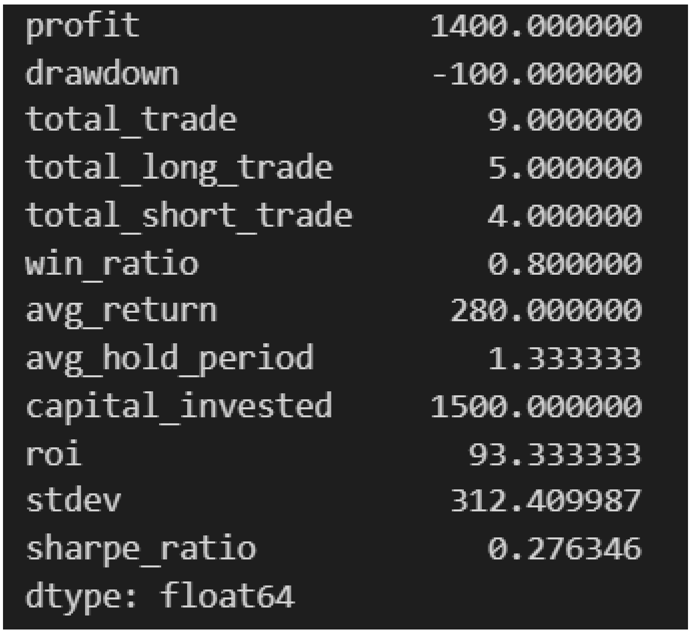
8. **ROI (Return on Investment):**
    - It is simply calculated as Profit/Capital Invested*100
9. **Standard Deviation:**
    - It is important metric to know the volatility of the portfolio.
    - Calculated using the profit_array, using the Numpy function of stdev
10. **Sharpe Ratio:**
    - Used to calculate how better is the portfolio performance with respect to the Risk free rate of interest.
    - Sharpe Ratio=(ROI-Risk Free Rate)/Standard deviation

Results:

For Question-1, I have made a dummy Dataframe to check if it is performing correctly:



Date	Symbol	Side	Size	Price
01-01-2024	AAPL	buy	10	20
02-01-2024	AAPL	buy	20	30
03-01-2024	AAPL	sell	30	50
04-01-2024	GOOGLE	sell	30	50
05-01-2024	GOOGLE	buy	10	60
06-01-2024	GOOGLE	buy	30	20
07-01-2024	GOOGLE	sell	10	30
08-01-2024	AMZN	buy	10	10
09-01-2024	AMZN	sell	10	20



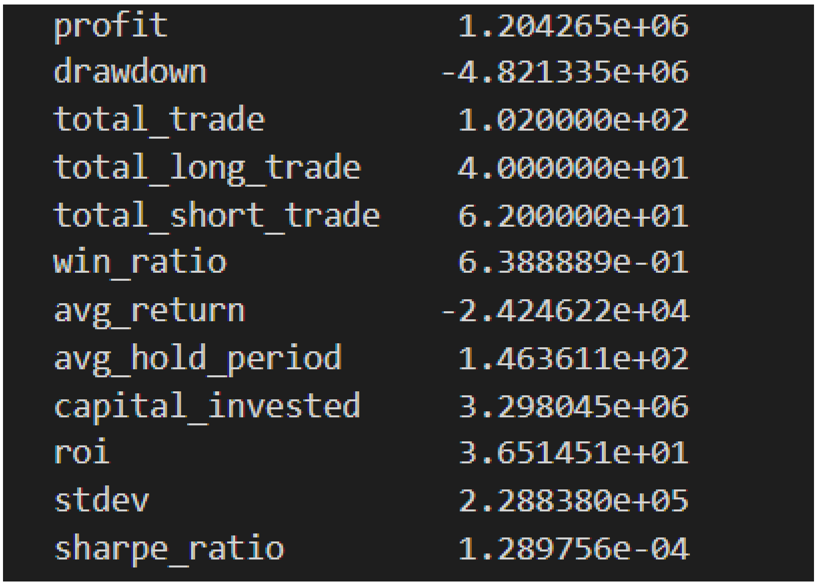
```
profit                1400.000000
drawdown              -100.000000
total_trade              9.000000
total_long_trade         5.000000
total_short_trade        4.000000
win_ratio                0.800000
avg_return             280.000000
avg_hold_period          1.333333
capital_invested      1500.000000
roi                     93.333333
stdev                  312.409987
sharpe_ratio             0.276346
dtype: float64
```

Question 2:

Assumptions:

- The Amounts columns had a range provided, the first entry of the range is taken as the price of the share
- Share Size is not given, it is taken as 1

Results:



```
profit                1.204265e+06
drawdown             -4.821335e+06
total_trade           1.020000e+02
total_long_trade      4.000000e+01
total_short_trade     6.200000e+01
win_ratio             6.388889e-01
avg_return           -2.424622e+04
avg_hold_period       1.463611e+02
capital_invested      3.298045e+06
roi                   3.651451e+01
stdev                 2.288380e+05
sharpe_ratio          1.289756e-04
```