TAQ Write-up

Sihan Qi, Sihan Liu, Ruihan Zhuang

Feb 2022

1 Preparing the TAQ Data

1.1 Part (b)

In this question, we are supposed to filter out the companies that are not in the SP500 list.

First, we call pd.read_excel to read the "s&p500.xlsx" file, put all values in the column "ticker symbol" into a list, and then apply a set function to remove duplicates in this list. In this way, we obtain a set that contains all ticker symbols of companies that are or have been a component of s&p500 index.

Then, we use os.walk function to check every file in our data folder. If the the ticker shown in the filename is not in our s&p500 set, we delete such file from our data folder.

1.2 Part (c)

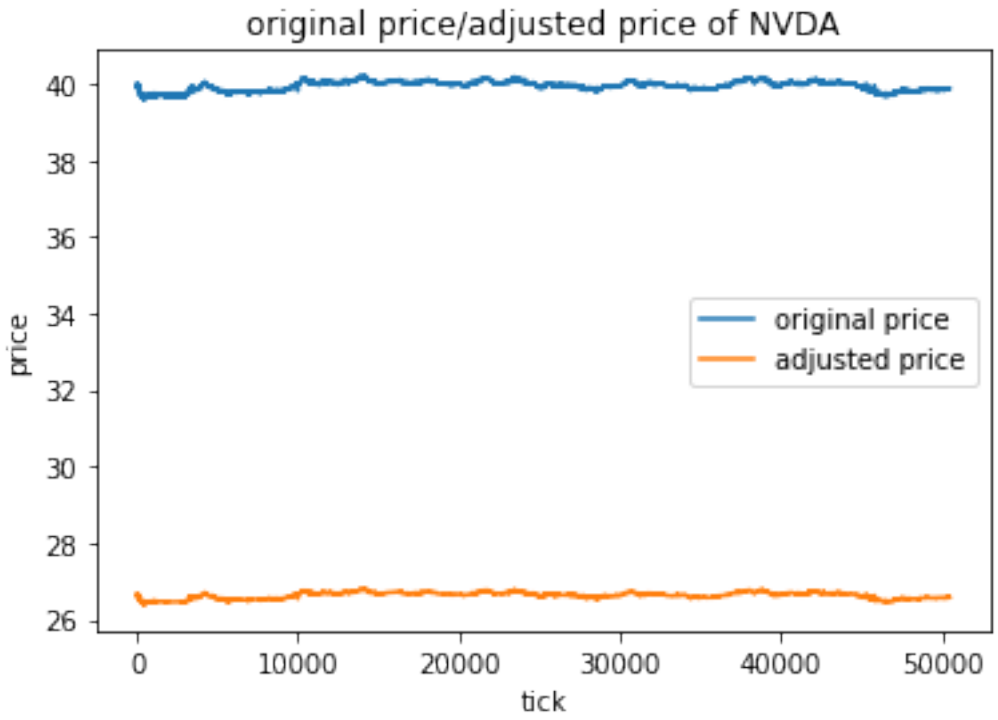
In this question, we are supposed to adjust trades, quotes and number of shares using the columns "Cumulative Factor to Adjust Prices" and "Cumulative Factor to Adjust Shares/Vol" in the s&p500.xlsx file.

Given that the last day of out data is "2007-09-20", the formula we used to adjust the price of a stock at date t is:

$$\text{adjusted price} = \text{price} * \frac{\text{Cumulative Factor to Adjust Prices at date t}}{\text{Cumulative Factor to Adjust Prices at "2007-09-20"}}$$

In this way, if the "Cumulative Factor to Adjust Prices" has not changed during the period, there is no adjustment of the price.

Shares/Vol is adjusted in the same way.



1.3 Part (d)

We use the suggested procedure to clean the the data. In order to justify what is the best parameter (k, γ), we clean the data, fit the data into ARMA(1,1) model and then compute the p value of Ljung-Box test for the residual of the data. We think the parameter performs better as the p value get larger since the large p value suggests that we are more confidence to say that we cannot reject the null hypothesis that the residual is not the white noise, meaning that data fit ARMA(1,1) model well as we desire.

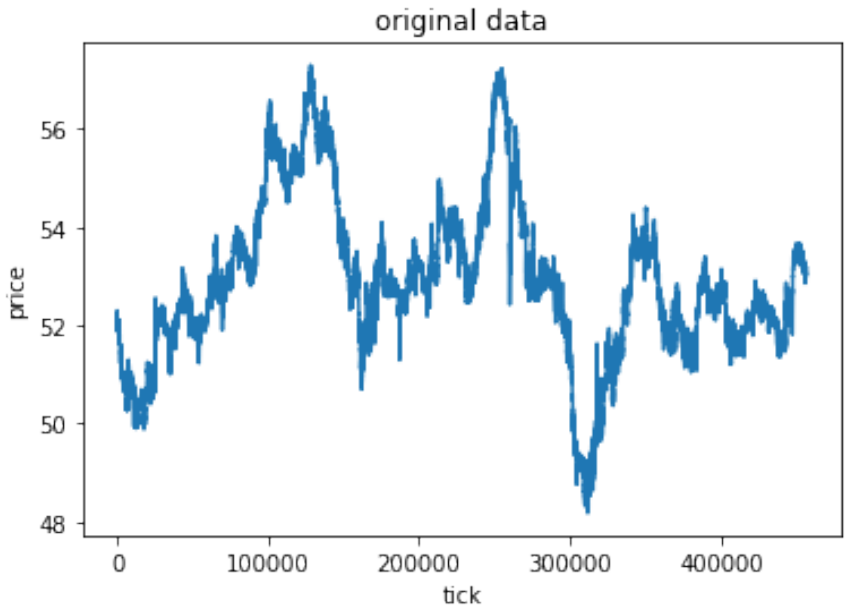
The optimized parameter is (k=100000, γ=0.001).

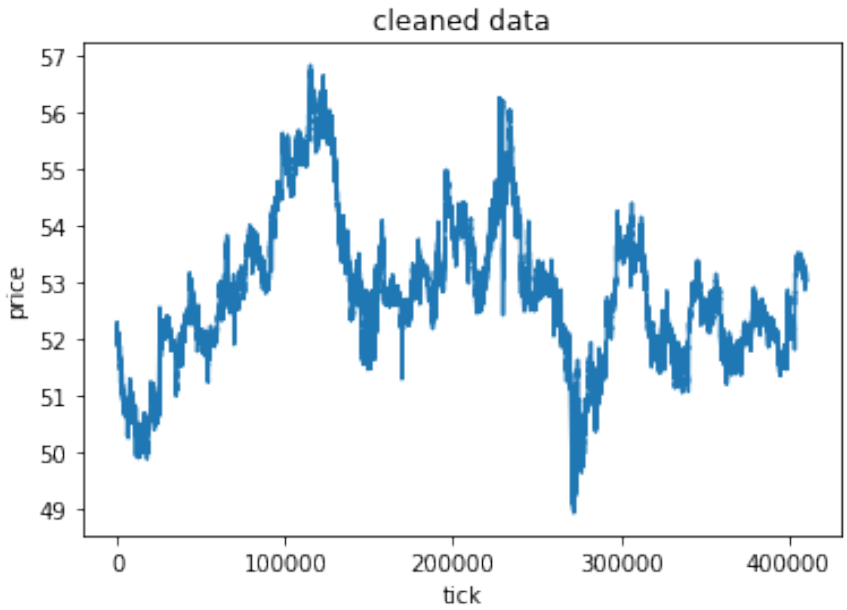
```python
#output the k and gamma with the largest p-value
def optimize(df_trade_large):
    df=df_trade_large[0:200000]
    mean=df["adjusted price"].mean()
    gamma=0.0005
    import statsmodels.api as sm
    k_list=[10000,100000]
    gamma_list=[0.0002,0.0005,0.001]
    df_cleaned=[]
    k_gamma_list=[]
    p_value_list=[]
    for k in k_list:
        print("k= ",k)
        for gamma in gamma_list:
            print("gamma=  ",gamma)
            data = clean_trade(df,k,gamma*mean)
            df_cleaned.append(data)
            data = np.array(data["adjusted price"].dropna())
            res = sm.tsa.ARIMA(data, order=(1,0,1)).fit()
            p_value=sm.stats.acorr_ljungbox(res.resid, lags=[10], return_df=True)["lb_pvalue"].to_list()[0]
            p_value_list.append(p_value)
            k_gamma_list.append((k,gamma))
    largest_index = p_value_list.index(max(p_value_list))
    print(p_value_list)
    print(k_gamma_list)
    return k_gamma_list[largest_index]
```

```python
1 optimize(df_trade_large)
```

```
k=  10000
gamma=   0.0002
         gamma=   0.0005
         gamma=   0.001
         k=  100000
         gamma=   0.0002
         gamma=   0.0005
         gamma=   0.001
         [6.122794173475554e-31, 7.408684952020 29e-35, 4.038576602689635e-23, 9.641912933412083e-21, 1.0874939670605314e-21,
         1.0815312619554706e-20]
         [(10000, 0.0002), (10000, 0.0005), (10000, 0.001), (100000, 0.0002), (100000, 0.0005), (100000, 0.001)]
```

Out[377]: (100000, 0.001)





We can see from the graph that there are less extreme values in the cleaned data.

2 Compute Summary Statistics of TAQ Data

2.1 Part (a)

Return of trades:

We noted that for each second, there are several trade records that have the same time stamp.

Thus, for each second, we produce a single price using the volume weighted average of the prices of the trades that happened in the same second.

Then we calculated X-second returns as the percentage change between the first price record in each X second intervals.

For return of mid-quotes, we first generate the mid-quote for each bid-ask pair and we use the same approach to calculate the returns.

2.2 Part (c)

3 Analysis of Autocorrelation

3.1 Part (a)

We write the function buck_test, which applies the Ljung-Box test to the x-seconds return of trade, with default lag $k = 10$. Starting with buck size x equals to 1, with each step adding 1 to buck size x, we keep running the Ljung-Box test until we find a buck size x_0 such that the p value of the Ljung-Box test is greater than 0.05, which indicates that we cannot reject the hypothesis that the x_0-seconds return of trade are independently distributed. In this way we find the smallest buck size that statisfied our requirement.

3.2 Part (b)

We write the function adf_test, which applies the ADF test to the x_0-seconds return of trade, where we use the function buck_test to find the buck size x_0. Then if the p value is smaller than 0.05, we say that the x_0-seconds return of trade is stationary. Otherwise, we say that the x_0-seconds return of trade is not stationary.

result: running this code

```
from src.compute_summary_stats import Stats
import pandas as pd
from src import MyDirectories

OUTPUT=MyDirectories.getTempDir()+"/output"
#use smaller dataset to minimize runtime
csv_path=OUTPUT+"/NVDA_20070619-20070621_trades.csv"
df=pd.read_csv(csv_path)
five_min_returns=Stats.ret_trade(df,5*60)
print(five_min_returns)
return_stats=Stats.ret_stats(five_min_returns,5*60)
print(return_stats)
```

gives the result saying that the input price series is stationary

```
optimized size is   15
Stationary
```

4 Mean-Variance Optimization in Python

4.1 Part (b)

4.1.1 Convergence Tolerance

The convergence tolerance in CVXOPT can be controlled using the abstol and reltol parameters when invoking the solver. The abstol parameter sets the absolute tolerance for the primal and dual residuals, while the reltol parameter sets the relative tolerance for the primal and dual residuals. The solver will terminate when both the primal and dual residuals are smaller than the specified tolerances. The default values for abstol and reltol in CVXOPT are 1e-7 and 1e-6, respectively. However, these values can be adjusted to obtain a desired level of precision or speed.

4.2 Mathematical Description of the sample

The expected return of a portfolio is given by:

$$E(r) = \text{pbar}' * x$$

where pbar is a column vector of expected returns for each asset, x is a column vector of the allocation of each asset in the portfolio, and $'$ denotes the transpose operator.

The risk (standard deviation) of a portfolio is given by:

$$\text{sigma} = \text{sqrt}(\text{x}' * \text{S} * \text{x})$$

where S is the covariance matrix of returns for the assets in the portfolio.

The portfolio optimization problem can be formulated as a quadratic program:

$$\text{minimize } \text{x}' * (\text{mu} * \text{S}) * \text{x} - \text{pbar}' * \text{x}$$

subject to:

$$\text{x}' * 1 = 1 \text{ (the sum of the allocations must be 1)}$$

$$\text{G} * \text{x} \leq \text{h} \text{ (inequality constraints on the allocations)}$$

$$\text{A} * \text{x} = \text{b} \text{ (equality constraints on the allocations)}$$

where mu is a regularization parameter that controls the trade-off between risk and return, and G, h, A, and b are matrices and vectors that define any additional constraints on the allocations.

4.3 Part (c)

Construct market portfolio In this question, we were asked to construct a market portfolio assuming that we are in a CAPM world.

We considered two approaches. First, we thought about estimating the expected mean and covariance matrix of the assets using historical data. Then perform a mean variance optimization as the example in Question 2(b) did with CVXOPT package. However, since there is no historical data prior to "2007-06-20" and that we have more assets than the number of dates, we cannot get an estimation of the expected return and covariance.

We switched to the second apprach, where the construction of the market portfolio purely relies on the CAPM assumption. Assuming that we are in a CAPM world where there are only s&p500 stocks in the market, then, as shown in any finance class, the weight of an asset i in the market portfolio should be

$$w_i = \frac{\text{market capitalization of the asset i}}{\text{total market capitalization}}$$

Using this method, we can get the weights of the stocks in the market portfolio on the given dates by computing their market capitalization.

To compute the market capitalization of a stock on a given date, we extracted the "Price or Bid/Ask Average" and "Shares Outstanding" columns in "s&p500.xlsx" file. Then

Market capitalization of a stock = price * shares outstanding

In this way , we get the components and weights of the market portfolio.

Compute turnover The formula we used to compute the turnover of the market portfolios is

Turnover = (Sum of absolute changes in weight of assets) / 2

where the "sum of absolute changes in weight of assets" is calculated by summing the absolute values of the differences between the weight of each asset on 2007-09-20 and its weight on 2007-06-20.

The result is: turnover ratio = 0.04752050004753335.