Stock Toolbox

Django Based Website for Stock Information Lookup and Buying Selling Point Recommendation

https://github.com/JasoDing/WebEng21_zd75

Zihao Ding
Rutgers University
New Brunswick. NJ. USA
zd75@scarletmail.rutgers.edu

May,2021, New Brunswick, New Jersey USA

Table of Contents

Abstract

Stock Toolbox is a website aim to provide service to people who have little or none experience to the stock market but still want to participate in stock investing. The website as its name, is a simple tool for user to learn about the stock market. The website provides stock information lookup, a watchlist for user to track the stock they are interested, and a buying selling point suggestion system based on stock history performance. The onsite suggestion system is evaluated by using past years stock history performance. Based on the evaluation, investments followed the site's suggestion system can be profitable.

1. Introduction

The service this website provide are stock performance display system, and the stock buying and selling point suggestion system. To be clearer, this website allows user to save the stock they are interested in and check the stock's performance and the stock related news in the dashboard. This website also can suggest buying and selling point of a specific stock based on the stock's history data. The user can evaluate the performance of the suggestion system by checking the portfolio's value the system returned.

The value of the service is to provide a basic tool for beginners to analysis and track different stocks. Most of the stock related finance websites have different screeners and filters to help user sort the stock based on someway, however, few of them actual provide buying and selling point suggestion to user. Among those websites which actually have buying and selling point suggestion, only very few can let user to directly evaluate the accuracy of their suggestion and how the suggestion actually performs based on the stock's history data. The Stock Toolbox website provides very clear buying and selling suggestion and enable user to directly evaluate the performance of the algorithm using any stock's history data.

The rest of this paper first discusses related work in Section 2, the background in section 3, and then describes our implementation in Section 4, 5, 6. Section 7 describes how we evaluated our system and presents the results.

2. Cross-reference to related work

Among most stock related websites, stock performance checking is only a basic function. Then different sites have different focus as addons. For example, Yahoo Finance has Yahoo News and very wide range of data (stock data for all stock markets, funds around the word, even the cryptocurrencies) as its addons which allows user to get as much stock related information they need. Trading View is another website that provides powerful mathematical analysis tools and forum for users to share their view and discoveries. Other sites provided by financial companies which provide very broad market evaluations for free with "asking a finance expert", which require consultation fee based on time or "visit". Users may get more reliable and professional result or answer with the cost of money. All of these sites are good tools for anyone to do analysis on stocks based on different perspectives however none of these sites allows user to directly evaluate how effective their service is.

3. Background of the service

The stock buying selling point suggestion service on stock toolbox site is inspired by the presentation made by Abhishek Chaudhuri on April.23th 2021. Where he demonstrated his stock prediction system based on short term and long-term historical Data. In his system, he uses linear regression to get the best fit trend line, his system gives very general buying and suggestions based on the slope of the trend lines. The buying and selling suggestion system in the stock toolbox is also based on short- and long-term stock historical data but with an easier and simpler approach using the idea of rolling average to determine the critical points and actions.

The background of the unfinished stock performance comparison service is based on Dynamic Time Warping (DTW) algorithm from [1][2]. DTW is one of the algorithms for measuring similarity between two temporal sequences in time series analysis. In general, DTW is a method that calculates an optimal match between two given sequences. More detail about DTW is in part 6.2.

4. Brief summary of the service

Stock Toolbox website mainly provide 2 service, stock information lookup with tracking, and stock buying and selling point suggestion.

For the stock information lookup and tracking, user can use the search bar in the top of the site to enter the ticker symbol to search for the stock they want. Once a ticker symbol is entered, the website will take user to the stock page which displays detailed stock related information. The user also has the option to save the stock they are interested in to a watchlist, they can then view all the stocks they stored with their performance in the Dashboard page.

For the stock buying and selling point suggestion service, on the suggestion page, user can enter the ticker symbol, start date, end date, and the amount they would like to invest to see the buying and selling point the service recommended with the simulation result. They can also choose to refresh this page daily to get the in time buying and selling suggestions.

5. Brief description of the several vies of the drawing

The Stock Toolbox website is a site based on Django. Initially this is a website which provide multi-user stock performance tracking and stock suggestion. The ideal site page structure is as Figure 1. However, due to the limitation of production time, the account authorization function is not finished. Thus, the current website is now single user only and its page structure is as Figure 2.

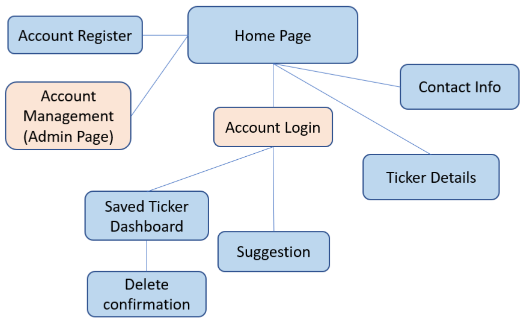


Figure 1: The ideal structure of Stock Toolbox website (pages in orange aren't finished due to limitation of time

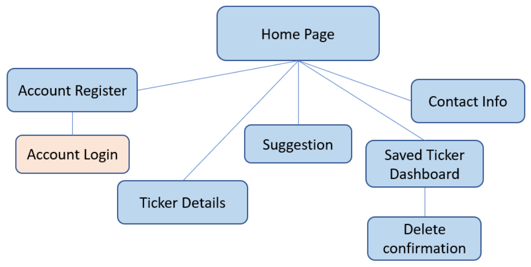


Figure 2: The current structure of Stock Toolbox website (pages in orange was removed due to unfixed issues with account authorization)

To achieve the presented structure, serve the site's main functions, and also save the amount of data that required to transfer every time user send a request, the Database for the website was designed as the following **Figure 3**.

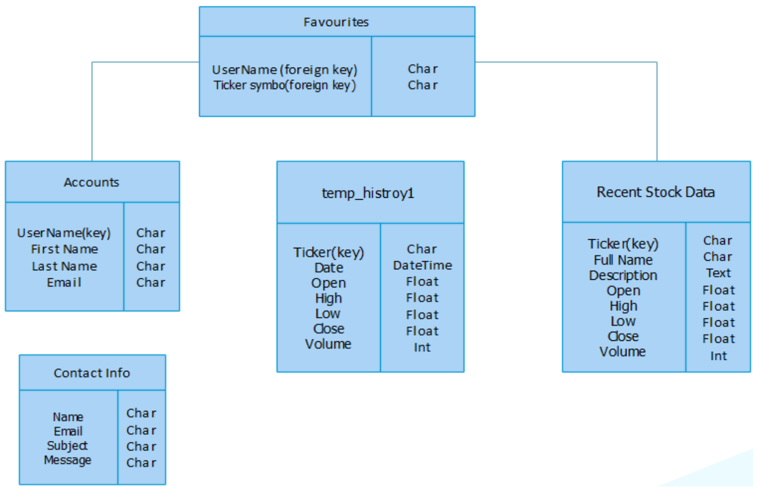


Figure 3: The Database structure of the Stock Toolbox Website

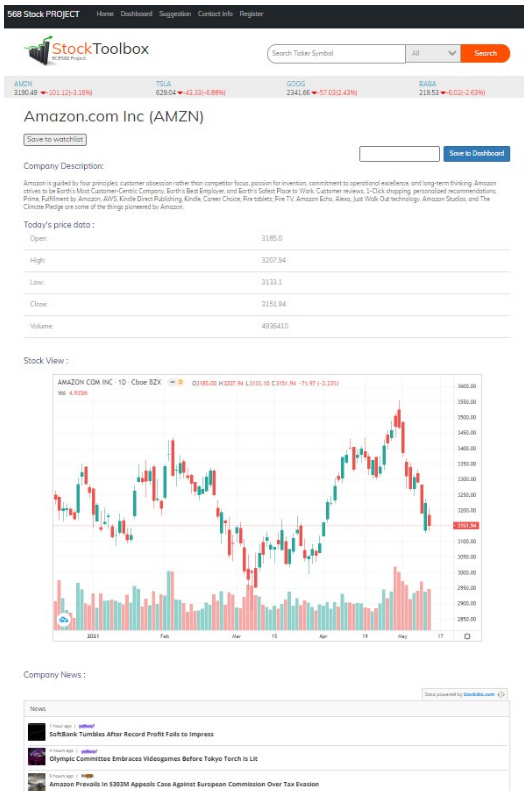
6. Detailed description of the web service

As mentioned in the previous part of the paper, the core of this website is trying to be the starting tool for stock trading beginners by providing stock price and related news with stock tracking and basic buying and selling suggestion. The first part of the web service is information collection and display, which tries to minimized the effort for beginners to gather stock related data and news. The rest part of the webservice is an attempt for giving some useful stock trading points to help beginner to learn about the market. This section also includes a stock performance comparison attempt with DTW, which was coded and tested locally but has not yet been deployed to the Stock Toolbox site.

6.1 Stock Related Information Collection

Stock related Information collection is not only the starting point for beginners to learn about the stock market but it is also a base for all other webservice. This information collection service provides every detail about the company the stock present, the history and current stock performance, and also some recent news related to the stock that the user is looking for.

Since this service need to provide in time stock data to user, this page is heavily based on the financial research API called Tiingo. The reason why I use Tiingo is because of the following reasons: as in 2021, yahoo API is free but is also "dead" for 9 years, only maintained by some volunteers. Some APIs which as good as Yahoo charges around 10 - 50 USD per month or by traffic. There are some "Free" APIs, I have looked the following: *Tiingo*, *Quandl*, *Alpha Vantage*, and *WorldTradingData*. I decided to use Tiingo because it gives more resource to free users. For detailed discussion about different APIs please refer to 568 Apr.4th.2021 group 5 report. By requesting live stock data using Tiingo API, the current price of the stock with the company description and history data are displayed in this page.



Amazon.com Inc (AMZN)

Company Description:

Amazon is guided by four principles: customer obsession rather than competitor focus, passion for invention, commitment to operational excellence, and long-term thinking. Amazon strives to be Earth's Most Customer-Centric Company, Earth's Best Employer, and Earth's Safest Place to Work. Customer reviews, 1-Click shopping, personalized recommendations, Prime, Fulfillment by Amazon, AWS, Kindle Direct Publishing, Kindle, Career Choice, Fire tablets, Fire TV, Amazon Echo, Alexa, Just Walk Out technology, Amazon Studios, and The Climate Pledge are some of the things pioneered by Amazon.

Today's price data :

Open:	3185.0
High:	3207.94
Low:	3133.1
Close:	3151.94
Volume:	4936410

Company News :

News

1 hour ago | yahoo!
SoftBank Tumbles After Record Profit Fails to Impress

4 hours ago | yahoo!
Olympic Committee Embraces Videogames Before Tokyo Torch Is Lit

9 hours ago |
Amazon Prevails In $303M Appeals Case Against European Commission Over Tax Evasion

Figure 4: A general view of the Collected Stock Information page. Started with basic stock information with an option to save the current stock to watchlist, an interactable stock performance graph, and the Company related News at the End.

The company related news feed is collected from the *stockdio*, a financial related service provider. News related to listed company is very important since they can affect the company's stock price in both ways. Some of the critical points that found in the next section (6.2) can all be explained by the news on the same day. For example, the critical point found by the (6.2) algorithm in Walmart's stock data on Feb.18th.2021, a huge drop (from 147.2 to 137.66) is due because on that day, Walmart hosted the virtual 2021 Investment Community Meeting where they released the quarterly financial report, it did not look good which reflects a huge drop in the next day's stock price. The company related news can explain more than 90% of the "shocks" in company's stock price. In order to provide a wider, accurate view of the stock market to the user, the news feed is a must.

6.2 Rolling Average Based Suggestion System

From Abhishek Chaudhuri's presentation on April.23th 2021, he presented a stock buying suggestion algorithm based on short- and long-term trend calculated by linear regression. Inspirited from him, I came up with the current rolling average based stock buying and selling algorithm.

The main idea of the suggestion algorithm is simple. The algorithm calculates the rolling average based on two different time period, every time when the short-term rolling average comes across the long-term rolling average, it is a "critical point" to make a buying or selling decision.

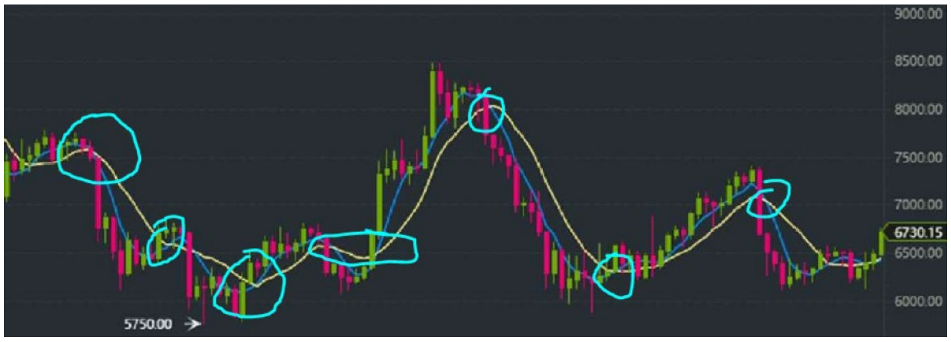


Figure 5: An example of critical points in rolling average-based suggestion algorithm

As the example provides in the Figure 5, yellow line presents the long-term average, the blue line presents the short-term average, Aqua circles presents the "critical points" where the short-term average and long-term average cross each other. In the figure short term average is 5-day average and the long-term average is 10-day average. The suggestion the algorithm is as simple as: if short term average is higher than the long-term average before the critical point, sell all the stock holds; if short term average is less than long term average before the critical point, buy the stock with the amount of money user provided. This algorithm simple but effective as long as using the correct window size for long and short term. It would be ideal if the window size is adjust differently based on different stock's history performance, however, due to the limitation of time for this project, I did not further explore this feature.

6.2.1 Rolling Average Based Suggestion System - Pseudocode

```
def avg(data, portfolio, hold, cycle = 0):
    # Wait until enough data collected
    if cycle < long term window:
        cycle += 1
        return (data, portfolio, hold, cycle = 0)

    # Main function:
    short_avgs = data.rolling(window=SHORT).mean()
    long_avgs = data.rolling(window=LONG).mean()

    #Average of 3 days before:
    Three = cycle - 3
    Short average 3 = short avgs[three]
    Long average 3 = long avgs[three]
    #Average of 2 days before:
    Two = cycle - 2
    Short average 2 = short avgs[Two]
    Long average 2 = long avgs[Two]

    # buy stock if:
    If (Short average 3 < long average 3) and (short average 2 >= long average 2) and hold == 0:
        # Buy stock with all money in portfolio
        SIGNAL _ BUY

    # Sell stock if:
    If (Short average 3 > long average 3) and (short average 2 <= long average 2) and hold > 0:
        #Sell all stock holds
        SIGNAL_SELL

    return (data, portfolio, hold, cycle)

end
```

6.3 Dynamic Time Warping (DTW)

The **unfinished** stock similarity comparison service used the idea of Dynamic Time Warping. The dynamic time warping, also called DTW, is an algorithm that using one-to-many method to apply Euclidean distance, which means for one point, it will not just directly map to the point at the same position in the other series, but find its correspondence in a certain range of points in the other series. This method is to find the optimal legal alignment between the two time-series, by finding the most similar point pairs defined by Euclidean distance, as Figure 5 shows, which will eliminate the influence of different phases or speed between the two series. Since all stock history performance data are time series data, DTW algorithm can be directly applies to compare the performance of different stocks.

During the process of finding such optimal alignments, there are three rules that we must follow: The Boundary conditions, monotonicity, and continuity. The boundary condition means the first index of one series must be matched with the first index of the other series, although they can be matched to multiple points. The same rule should also be followed by the two last indices. The boundary condition ensures that the DTW calculation runs cover the full range of both series. Monotonicity means the mapping of the indices from the first sequence to indices from the other sequence must be monotonically increasing, and vice versa. In other words, the matching lines in the right picture must not be crossed with each other in the middle. Continuity means Every index from the first sequence must be matched with one or more indices from the other sequence and vice versa.

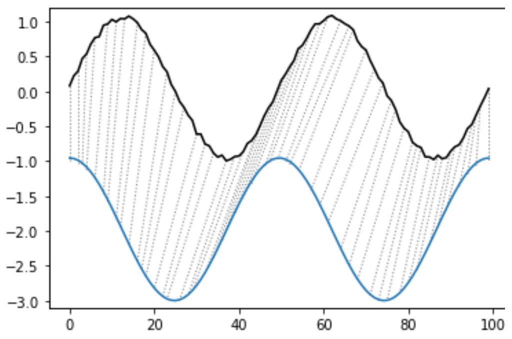


Figure 5: Illustration of DTW

Considering these rules together, finding optimal alignments is just like moving two pointers along the two time-series. Pointers can move forward one index at a time, or stop, but cannot move backward. At the beginning, both pointers are at the first index of each series. For each of the following step, there are three options: pointer A moves forward and pointer B stays, B moves forward and A stays, or both A and B move forward. Such decision is made by comparing the Euclidian distance of the three pairs of indices, and choose the minimum distance to move. The effect of such step is when the patterns of two series are matched, both pointers will move forward, and when the patterns are not matched, one pointer will stay until the other pointer find the next matching index, which is shown by the horizontal and vertical path in the left image and the colored matching line in the as Figure 6.

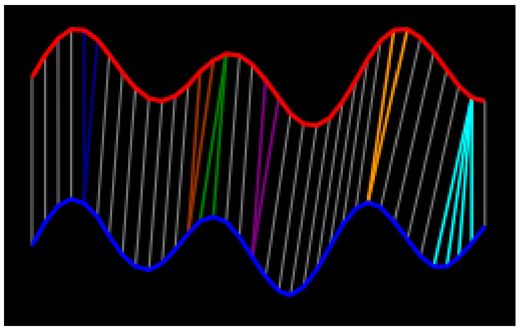
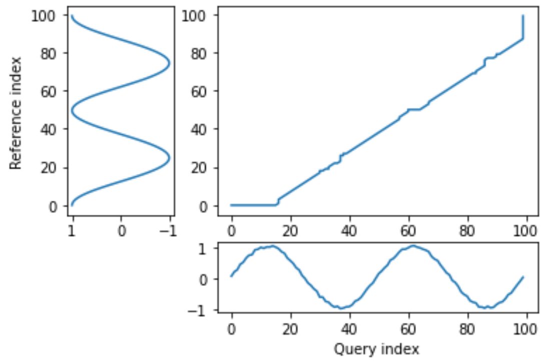


Figure 6: matching point pairs in DTW

Using mathematical equations to express the process mentioned above. First, we need to create an M by N matrix A, where M and N are the length of each series. Then we define a cost function as the picture shows to fill in matrix A. Each cost value is equal to the Euclidean distance between two indices at that entry plus the local minimum of its possible predecessors. So we can trace a path from A(0,0) to A(m,n), with all of its values are local minimums. The cost at A(m,n) is defined as the DTW distance between the two series.



$$D(i,j) = Dist(i,j) + \min[D(i-1,j), D(i,j-1), D(i-1,j-1)]$$

Figure 7: Calculation of distance matrix

DTW has following restrictions and rules [1]:
I) Every index from the first sequence must match with one or more indices from the other, vice versa
II) The First index from first sequence must matched with the first index from the other sequence. (but it does not have to be its only match)
III) The last index from the first sequence must be matched with the last index from the other sequence (but it does not have to be its only match)

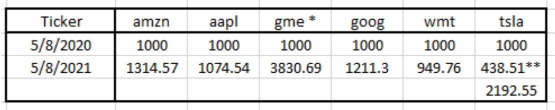
IV)The mapping of the indices from the first sequence to indices from the other sequence must be monotonically increasing, and vice versa

6.3.1 DTW Pseudocode (DTW)

```
def dtw(stock history x, stock history y):
    # Initialization of DTW martrix
    for i = 1..n
        for j = 1..m
            C[i, j] = inf

    C[0, 0] = 0.

   # Main loop of DTW matrix
   for i = 1..n                    # For Each Row
       for j = 1..m                # For Each Column
           dist = d(x_i, y_j) ** 2   # ED distance
           C[i, j] = dist + min(C[i-1, j], C[i, j-1],
                                C[i-1, j-1])

   return sqrt(C[n, m])
end
```

7. Evaluation

To evaluate the suggestion system in 6.2, several tests had been run using some well-known listed company's history stock price. The evaluation is to simulate if using $1000 to start on May.8th.2020, find out the final return after a year. The first row presents the ticker symbol of the listed companies, the second row presents the starting fund, the 3rd row presents the total amount after a year if buy and sell stock totally based on the 6.2 algorithm. Overall, among most of the large listed companies, the result can be said it is profitable. However, there are some special cases in the table presented.

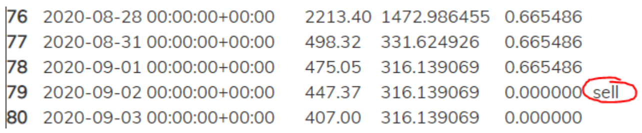


Ticker	amzn	aapl	gme *	goog	wmt	tsla
5/8/2020	1000	1000	1000	1000	1000	1000
5/8/2021	1314.57	1074.54	3830.69	1211.3	949.76	438.51**
						2192.55

Table 1: The year Return of $1000 investment at May.8th.2020 using the 6.2 algorithm

The first special case is the GME, as for the past several months, the "wallstreetbets" subreddit have been pushing the price of GME, the algorithm caught the trend which result in 3.8 times return of the initial fund.

The second special case is the TSLA, the algorithm resulted as only $438.51 after a year's investment, this is due to the 1 to 5 SPLIT happened on Aug.31st.2020, as presented in Figure 8. As for the 1 to 5 splits, the actual return of a year is 5 times of the result which is $2192.55 as presented in the box in table 1. The split resulted a huge drop in the market close price which falsely triggered the algorithm to sell all the holds. However, even with the mistake, the result is still profitable.



```
76  2020-08-28 00:00:00+00:00    2213.40  1472.986455  0.665486
77  2020-08-31 00:00:00+00:00    498.32   331.624926   0.665486
78  2020-09-01 00:00:00+00:00    475.05   316.139069   0.665486
79  2020-09-02 00:00:00+00:00    447.37   316.139069   0.000000 sell
80  2020-09-03 00:00:00+00:00    407.00   316.139069   0.000000
```

Figure 8: The 1-5 split in TSLA, which falsely triggered the algorithm to sell the stocks.

The third case is WMT, where result in negative return. The reason is that in 2020, covid-19 has impacted the company negatively. The cause of several huge drops with in a day that the suggestion algorithm cannot handle in time is mainly due to the publish of their Quarterly financial report. Because the algorithm in 6.2 can only detect the change in the second day after the drop, which means at the time it sells, it is already too late. Such thing can be prevented by reading the company news, this is also why the news feature was embedded in the main service.

Overall, the suggestion algorithm on site Stock Toolbox performs fairly based on the history stock data, but it cannot handle the split of shares, nor the drop in 1 day. If the stock price changed too fast in one day, the algorithm cannot catch it in time and can only perform buy or sell action on the second day.

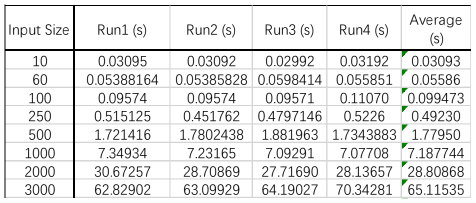
The suggestion algorithm cannot handle the split of shares because there is no signal in the stock close price to tell if there is a split or not. But this can be solved by calculating the total shares instead of holds as Prof. Liang suggested. For the second flaw, at current stage there is no solution since the algorithm only takes stock history close price into consideration. The best way to react to the unusual drop or rise in time is keep track to the company's news. Which may require natural language processing, which is also way beyond my knowledge.

8. Claims

No claims from author, all materials from this paper are free to use based on MIT license.

9. Other things worth to mention

As you may notice, the section 6.3, the DTW algorithm was not implemented in the Stock Toolbox website, this is due to the limited of time and the performance of the algorithm did not met my expectations. The reason why the DTW algorithm is still presented in the paper and also presented in the code is due to for the past months, I have been working on implementing several different algorithms to be the core webservice of my project. The DTW algorithm took around 11 minutes to do one similarity search between only a 100 stock's history data, with each stock has a year of history close price. It is way to slow as a live webservice and is very space consuming compared with the algorithm presented in 6.2. DTW based stock similarity search also does not provide as much useful information as the stock buying and selling algorithm currently in the website. Thus, I decided not to put it in the final site, but still included it in this paper and the final submission since I have already spent way too many times on it.



Input Size	Run1 (s)	Run2 (s)	Run3 (s)	Run4 (s)	Average (s)
10	0.03095	0.03092	0.02992	0.03192	0.03093
60	0.05388164	0.05385828	0.0598414	0.055851	0.05586
100	0.09574	0.09574	0.09571	0.11070	0.099473
250	0.515125	0.451762	0.4797146	0.5226	0.49230
500	1.721416	1.7802438	1.881963	1.7343883	1.77950
1000	7.34934	7.23165	7.09291	7.07708	7.187744
2000	30.67257	28.70869	27.71690	28.13657	28.80868
3000	62.82902	63.09929	64.19027	70.34281	65.11535

Table 2. The performance of DTW based similarity comparison algorithm

Thanks for reading.

References

[1] Salvador, Stan and Chan, Philip. 'Toward Accurate Dynamic Time Warping in Linear Time and Space'. 1 Jan. 2007 : 561 – 580.

[2] Django documentation

[3] Rakthanmanon, Thanawin, Bilson, Campana, Abdullah, Mueen, Gustavo, Batista, Brandon, Westover, Qiang, Zhu, Jesin, Zakaria, and Eamonn, Keogh. "Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping." . In Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 262–270). Association for Computing Machinery, 2012.