Jinyang Ruan
011696096

CPT S 515 Advanced Algorithms Final

***Abstract*—This paper will answer two questions by using the knowledge of hashing, de Bruijn graph, Markov chain and matrices to predict stock market based on previous days' data and to measure the similarity between two stock charts.**

***Keywords*—stock market, hashing, de Bruijn graph, Markov chain, matrix**

I. Introduction

Many people believe that stock market is unpredictable. However, if we closely looking at a stock chart, we will find the data on the chart should be predictable since they are not randomly generated at all. As long as we have enough certain data of one stock, we could find a way to predict the trend of that stock and let algorithm help us to make decisions whether we should buy or sell the stock. In this paper, I will propose approaches to answer two questions: A. How to predict tomorrow's data for a stock based on previous chart history. B. Given two stock charts, how to determine the similarity between those two.

II. Approach to Predict One Stock Chart

Step1. Problem set up and analysis

We use $A(\alpha)$ to denote a stock predicting algorithm A which takes only one input α where α is a stock chart. A stock chart α has a very simple data structure: an array $\alpha[1 \dots k]$ for some large k. Each element $\alpha[i]$ is a market data for the stock of day i, which is represented by a tuple of five numbers:

- open price of the day
- close price of the day
- highest price of the day
- lowest price of the day
- number of shares traded in the day (volume)

Step2. Hash this array

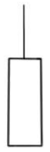
From the chart, I cannot find a clear relationship between whether we can earn money or lose money and the value of volume. Just in case volume do affect the price of the day, I roughly divide "volume" into the following 9 categories in million:

$$[0,50], (50,100], (100,150], (150,200], (200,250],$$

$$(250,300], (300,350], (350,400], (400,+\infty]$$

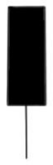
Our goal is to make money, which means we will buy this stock when we predict the "price" for the next day is higher than today. Based on every days' price trend, I start by dividing them into the following 5 categories:

1. *open price = the lowest price*



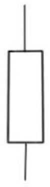
In this case, the "price" of next day will be absolutely higher than today's. You will absolutely earn money if you hold this stock today, say 100% win.

2. *open price = the highest price*



In this case, the "price" of next day will be absolutely lower than today's. You will absolutely lose money if you hold this stock today, say 0% win.

3. *close price > open price but none of them is either highest or lowest.*



In this case, the price of next day will be sometimes lower than today, but generally speaking, the price of next day will be higher than today. The probability we will earn money in this day can be computed as:

$$Prob = \frac{highest\ price - open\ price}{highest\ price - lowest\ price}$$

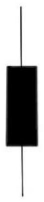
Then we set 5 probability domains to classify every probabilities under this case:

$$(0,0.2], (0.2,0.4], (0.4,0.6], (0.6,0.8], (0.8,1)$$

For each range, we use the middle number to denote the win rate which can be represented by:

$$0.1, 0.3, 0.5, 0.7, 0.9$$

4. *open price > close price, but none of them is either highest or lowest.*



Generally speaking, we will lose money at this day. The win rate can be computed by:

$$Prob = \frac{highest\ price - open\ price}{highest\ price - lowest\ price}$$

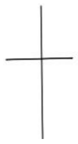
Similarly, Then we set 5 ranges to classify every probabilities under this case:

$$(0,0.2], (0.2,0.4], (0.4,0.6], (0.6,0.8], (0.8,1)$$

For each range, we use the middle number to denote the win rate which can be represented by:

$$0.1, 0.3, 0.5, 0.7, 0.9$$

5. *open price = close price*



The win rate can be computed by:

$$Prob = \frac{highest\ price - open\ price}{highest\ price - lowest\ price}$$

Similarly, Then we set 5 ranges to classify every probabilities under this case:

$$(0,0.2], (0.2,0.4], (0.4,0.6], (0.6,0.8], (0.8,1)$$

For each range, we use the middle number to denote the win rate which can be represented by:

$$0.1, 0.3, 0.5, 0.7, 0.9$$

According to the above classifications, I found that win rate of all situations can be calculated by

$$Prob = \frac{highest\ price - open\ price}{highest\ price - lowest\ price}$$

So, I reset categories in 7 based on the win probability:

- 0% when $prob = 0$
- 10% when $prob \in (0,0.2]$
- 30% when $prob \in (0.2,0.4]$
- 50% when $prob \in (0.4,0.6]$
- 70% when $prob \in (0.6,0.8]$
- 90% when $prob \in (0.8,1)$
- 100% when $prob = 1$

So far, we have totally $9 * 7 = 63$ categories, and we hash the chart $\alpha[1 \ldots k]$ into those 63 slots. The number of categories can be modified by change the range of probability domains.

Step3. Contruct the de Bruijn Graph

The hash table is finite, so assume that hash table is a finite symbol set. After hashing $\alpha[1 \ldots k]$, each element $\alpha[i]$ can be represented by a symbol. For each symbol, there is a probability can be computed which is the rate we can earn money. Then we can get a sequence of symbols which can represent the chart.

Based on this sequence, construct the de Bruijn graph.

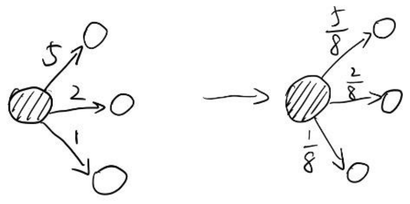
Frist we need to choose a positive parameter, say $n = 4$.

Pick the first 4 symbols in the sequence as one node in de Bruijn graph, recursively, move backward one step at a time, every time combine 4 symbols to 1 node in the graph. At the same time, add directed edges between nodes based on the order of sequence.

In order to convert it to Markov chain, we put a count m to each edges, where m is the number of appearance of substring in the original sequence. In this case, the length of substring is 5.

Step4. Normalization

Normalize the de Bruijn graph into a Markov chain. We need to make the sum of probabilities on each node be 1. For example, one node in de Bruijn graph can be normalized as follow:



Construct the transition probability matrix T.

$$T_\alpha = [P_{ij}] = \begin{bmatrix} P_{1,1} & \cdots & P_{1,n} \\ \vdots & \ddots & \vdots \\ P_{n,1} & \cdots & P_{n,n} \end{bmatrix}$$

where P_{ij} means the probability from station P_i to P_j, and the summation of every row is 1.

So far, as long as we have the chart of 2000 previous days' data and today's station (symbol), we can know the probabilities of going different next stations with the win rates. Then, we can roughly predict the price for the next day.

III. Approach to Determine the Similarity Between Two Charts

From the precious section, we know we can hash one stock chart, and then use Markov chain to construct the transition probability matrix *T*. For every stock chart, we will get a unique transition matrix. Thus, the question of determining the similarity between two charts can be translated into determining the similarity of two transition probability matrices of charts.

The similarity between two matrices can also be considered as the distance between two matrices.

Assume that we have matrix A which we get from chart α and we have matrix B which get from chart β. Now, we want to determine the distance between A and B.

Transition matrix must be square, so one way to calculate the distance between (A,B) is using Frobenius distance F [1]:

$$F_{A,B} = \sqrt{trace((A-B)(A-B)')}$$

where B' represents the conjugate transpose of B, and the trace of matrix is the sum of the diagonal elements. The smaller $F_{A,B}$, the more similar the two matrices.

There are also some other ways to determine the distance between two matrices:

$$d_1(A,B) = \sum_{i=1}^{n}\sum_{j=1}^{n}|a_{ij} - b_{ij}|$$

$$d_2(A,B) = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}(a_{ij} - b_{ij})^2}$$

$$d_\infty(A,B) = \max_{1\le i\le n}\max_{1\le j\le n}|\,a_{ij} - b_{ij}|$$

For all of above methods, the smaller $d(A,B)$, the smaller distance between two matrices, the two matrices are more similar, the similarity between two charts are higher.

IV. Conclusion

In this paper, I proposed one method to predict the price of next day with the input is a single chart, and several methods to determine the similarity between two stock charts. For the first question, if we have the stock chart for each day and the price for every time, we can use integral to compute the measure of area, the probabilities we get will be more accuracy. For example, one day's close price is much lower than open price, we still could make money at that day because at some point in time, the price can be very high. After building a appropriate hash function and hashing stock chart, the stock chart can be represented by a finite symbol sequences, build de Bruijn graph out of this sequence, normalize it into Markov chain, and construct transition probability matrix. So far, we can predict the price for the next day based on the transition matrix. In the second question, I translated the original question into determining the similarity between two matrices, then I associated "distance" with "similarity". There are many approaches to measure the distance between two matrices, the question is whether we can consider "distance" and similarity as the same.

References

[1] Synex.(https://math.stackexchange.com/users/88832/synex), Distance/Similarity between two matrices, URL (version: 2013-09-29): Retrieved from: https://math.stackexchange.com/q/507742