Final Report

Stock Analysis by Hebbian Learning

Finding relationship between stocks

Name: Sun Wu Choi

Due: 2020/12/15

**High Level Description**

**[1] Motivation**

I read a book called “The Discovery of Intelligence, How Do Ants Think?” and I read this part about Hebbian learning. Hebbian learning is a method of how neurons create connections between them. The connection is strengthened by myelin sheath that preserves the signal between neurons. After finishing reading this book, I was interested in stocks.

I thought of combining those two concepts, Hebbian learning of neurons and stocks. If I can treat each stocks as neurons and if the stock fluctuation is the signal of each neurons, then I thought I could be able to apply Hebbian learning to stocks and sort out the list by myelin sheath score from highest to lowest to see which stock is the most connected with which stock. So I started looking for datasets first.

Diagram

Description automatically generated

**[2] Description**

The approach is simple, each stock will have the data from when it was founded and its stock data until 2020 April 1st. Some other stock data could be founded on the same day but that is mostly unlikely, so I the algorithm compares from the date when the two stocks starts to have the same day, which will be the younger company’s founded date.

Then we calculate the difference by subtracting the stock open price and stock close price. We divide this by the stock open price to normalize it by how much it increased in percentage. For example, if the stock was 5 dollars when it opened and 10 dollars when it closed, its difference will be 100% or in this case, 1.0.

We multiple the difference of two stocks at the same day to get a Hebbian learning score. We sum all the Hebbian learning score and divide by the compared length. This will give a nice Hebbian learning score, regardless of the compare length. But when we multiply, there is one major problem that I will describe later in this report. I fixed this issue and the fix was also motivated from neuroscience.

Diagram

Description automatically generated

**[2] Problem Statement, model, input output**

**[1] Problem statement**

Problem statement is simple, we will input two stock data and we want to return an output that represents the relationship between those two input stocks. After creating this function, we need to clarify what exactly this output relationship score means.

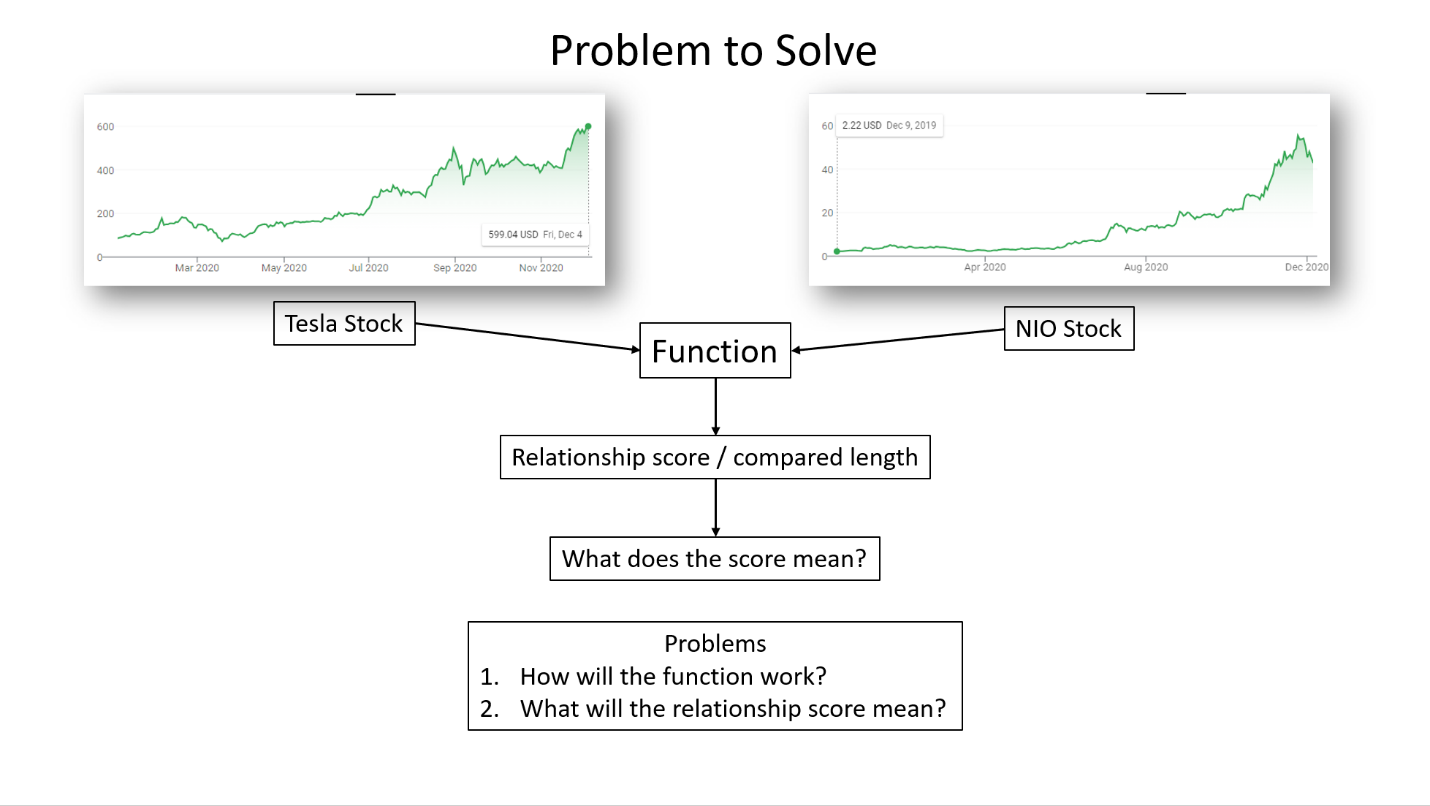
**[2] Model**

The model used in this project is a simple Hebbian learning equation anyone can find with some search, but I don’t think much people tried this on stocks.

This model will create a relationship score from two stocks regardless of their foundation date and end date. The relationship will be divided by compared length so that it is normalized by its compared length.

**[3] Input & output**

As I mentioned above, input will be two unique stocks which is not ETF or ETN. The output will be a relationship score generated by the model above.



**[3] Dataset**

**[1] Dataset**

First, I searched Rapid API website to see some financial apis to use. I found morning star, yahoo finance, Bloomberg and etc. But I figured out there will not be enough data if I get it day by day. So I needed a large training data from the past. Kaggle had many datasets that I was looking for. I found a dataset that was up to date to 2020 April 1st. This data contained stocks data including ETFs which was merely 8000 datas and each data was from about 1963 for the oldest and 2018 for the youngest.

This was perfect dataset for my project. But the problem was that ETF is a collection of stocks, which means it will look a lot similar like some stocks or they will look alike other ETFs. So I had to figure a way out to preprocess the dataset so that every ETF and ETN are removed before running Hebbian learning.

**[4] Methods**

**[1] Algorithm used**

Hebbian learning was implemented in this algorithm. The main concept is to multiple the difference percentage of two stocks but there is a problem. When one stock increases 100% and the other increases only 1%, then the score is calculated as 10%. But this is not how neurons work in Hebbian learning, the connection strength is strengthened when two neurons fire at a merely same strength.

The solution for this problem was to decrease the higher absolute value to be the same as the lower one. So if its +100% and +1%, then the score will be +1% \* +1% = 0.0001. If its +10% and +12%, then the score is +10% \* +10% = 0.01, when it’s the opposite value like -10% and +9%, then its -0.009.

This fix prevents when a stock that is so cheap, such as 0.5 dollars per stock, increases by 50% in a single day, can affect too much to the Hebbian learning when we just multiply them together. This problem was too strong when I ran the Hebbian learning by just multiplying them. The highest relationship with Tesla was some weird company that had nothing to do with Tesla.

**[5] Empirical Results**

**[1] Results**

The list below is the top 30 relationships from the highest to lowest.

|  |  |  |  |
| --- | --- | --- | --- |
| NIO | 361 | 0.055032 | 0.000152 |
| GLUU | 2427 | 0.344327 | 0.000142 |
| ROKU | 601 | 0.078771 | 0.000131 |
| CSIQ | 2427 | 0.310159 | 0.000128 |
| SLDB | 519 | 0.063017 | 0.000121 |
| SPWR | 2427 | 0.293803 | 0.000121 |
| TWTR | 1580 | 0.190712 | 0.000121 |
| PGEN | 1644 | 0.198251 | 0.000121 |
| PS | 442 | 0.053301 | 0.000121 |
| SMAR | 456 | 0.054935 | 0.00012 |
| ESTC | 344 | 0.041331 | 0.00012 |
| PTCT | 1678 | 0.201135 | 0.00012 |
| NIU | 334 | 0.039672 | 0.000119 |
| APLS | 571 | 0.067614 | 0.000118 |
| SQ | 1068 | 0.125367 | 0.000117 |
| JKS | 2427 | 0.283434 | 0.000117 |
| FEYE | 1614 | 0.188422 | 0.000117 |
| ZIOP | 2427 | 0.281169 | 0.000116 |
| HALO | 2427 | 0.280203 | 0.000115 |
| DDD | 2427 | 0.277854 | 0.000114 |
| CDLX | 509 | 0.058262 | 0.000114 |
| YELP | 2004 | 0.229266 | 0.000114 |
| RUBY | 400 | 0.045538 | 0.000114 |
| PRNB | 359 | 0.040656 | 0.000113 |
| EDIT | 1018 | 0.115195 | 0.000113 |
| ARLO | 388 | 0.043803 | 0.000113 |
| SSYS | 2427 | 0.273731 | 0.000113 |
| SRPT | 2427 | 0.273223 | 0.000113 |
| LTHM | 340 | 0.038146 | 0.000112 |

NIO had a high relationship of 0.000152, and the reason is because NIO is also an electric car company. GLUU and ROKU are entertainment companies, and most of the high ranked companies in the list are solar power related or 3D printing or pharmacy & biotech companies.

This shows that not also it shows the related industry like solar power and 3D printing, it also shows which industry has a similar pattern, like eco energy, entertainment, and bio moves together. This is because some trends move together.

**[6] Discussion**

**[1] Future directions**

My father gave me many intuitions and guided me through this project with his knowledge in years and years of stock experience. He suggested me to make a parameter to set the compare length to recent one year or two or five. Because it is easy to know which company is related to which, but this algorithm has a strength that it can figure out recent relationships such as contracts between two companies. 3D companies showed up to be high in relationship with Tesla only because Tesla used that company’s 3D printing technology after they made a contract. This means the Hebbian learning can find out recent relationships between two companies and that is a really good data. I might further develop this and check the market if I can actually make a website with it. I will definitely add It in my resume so I can get an intern at a financial company in the future.

Another thought is that if I was able to derive such data from Hebbian learning, what if tinker the algorithm more precisely and upgrade it to derive even more relationships or data? I might further study in machine learning and learn more tools so I can get back at this in the future.

If I can give any advice for the students who are doing a project like this, I would definitely say to do something that you truly enjoy and love every moment with it. When I was doing this project, I was enjoying it so much and stayed up until 2am every time I was working on it. Not because I had problems, but because it was just so fun tinkering the algorithm and looking at it. This advice is not only for CS5100 but it’s an advice in life, just don’t care about money or GPA, just enjoy and love the moment and those will follow you naturally.

To sum it up, this project taught me that I am capable to create something from nothing using computer science tools. It gave me confidence that I can do well in the future AI industry. Also, my interest in neuroscience has been useful for the first time and I am so proud to myself that I was able to combine the two subjects that I love so much.