HypeFyn Project Final Report

Pietro Dominietto, Austin Ho-Chun Fan, Katie Jooyoung Kim

All team members contributed an equal amount of work to the project.

1 INTRODUCTION

Our group project was inspired from a simple question we asked during the bull market of early 2020: in a market that exhibits so much volatility on a day to day basis, how does positive or negative hype about a company and related equities influence the stock prices? There has been previous market research into this question [1][7][8], and as the bull market of early 2020 progressed into a volatile bear market, we had an interesting opportunity to observe both positive and negative sentiments [5]. In our investigation, the term "hype" was defined as how much discussion - *volume* - there was about a keyword, and how positive or negative - *sentiment* - the discussion about that term was. We decided to investigate this question using Twitter data, based on its real-time nature.

2 PROBLEM DEFINITION

Our problem is to investigate the correlation between Twitter hype (as defined above) and day-to-day changes in stock market data. In order to find out more about this relationship and to make it easily accessible and understandable, we decided to visualise the data in the format of graphs embedded within a website. We decided to focus our evaluation on the following keywords, companies and the corresponding stock tickers [9].

- Amazon / $AMZN
- corona / COVID
- Delta / $DAL
- Google / $GOOG
- Netflix / $NFLX
- Novartis / $NVS
- Pfizer / $PFE
- Tesla / $TSLA
- Tripadvisor / $TRIP
- ~~Uber / $UBER~~ [1]
- Zoom / $ZM

This choice was made based on the development of events following the spread of COVID-19. As we had an unprecedented opportunity to observe a very high volume of sentiment related with the global pandemic on Twitter, we wanted to analyse the industries that would be most heavily influenced. Therefore, we chose the categories travel, pharmaceuticals, and stay-at-home to classify the chosen firms, and added Google and Tesla as firms that had shown high Twitter volume before the pandemic to act as the control group.

(1) Travel: Delta, Tripadvisor, Uber
(2) Pharmaceuticals: Novartis, Pfizer
(3) WFH: Amazon, Netflix, Zoom
(4) Control: Google, Tesla

Once we collected the Twitter data, we aggregated the positive & negative sentiment about the keywords and the associated Twitter volume into a single metric titled the "Hype Score". Calculation of this metric involved sentiment analysis based on natural language processing.
Finally, we visualised the Hype Score alongside the actual stock market data that we collected separately.

3 METHODOLOGY

3.1 Data Collection

3.1.1 Twitter data. On Twitter, we collected data about the aforementioned keywords. We used the libraries `Tweepy`[2] and `GetOldTweets3`[3], both available on GitHub under the MIT license. While `Tweepy` is capable of data collection from only the past 7 days, `GetOldTweets3` is precisely a library developed to collect older tweets. Technical details included the following.

- The period of our data collection was for 30 days, from 2020-03-22 to 2020-04-20.
- The query is not case-sensitive, as is the case in the normal Twitter search API.
- For each search result corresponding to a keyword, we collected its text, number of favorites, number of retweets, and the time at which it was created.
- In most cases, the company name and corresponding stock ticker were distinct enough that we could rest assured that we were not collecting duplicate data. However, for the case of Uber Technologies, this was not the case. The Twitter search API treats both Uber and $UBER as the same search keyword[4]. This led us to exclude Uber from our final report.

To effectively store our Twitter data and streamline the subsequent NLP analysis, we decided to use services offered by Google Cloud Platform (GCP hereafter). On GCP, we used Cloud SQL to store the data collected by the Python libraries, and the NLP model was connected directly to this database to perform computations. This required us to use `PyMySQL`[5] to establish a connection between the local machine running the Python script and the cloud database.

3.1.2 Stock data. Stock price was collected in 1 minute intervals for each of the 9 stocks (including Uber before deciding on not using it) using the Python `yfinance` package. The `yfinance` package

[1] Excluded from final experiment. Refer to section *3.1.1*.

[2] https://github.com/tweepy/tweepy
[3] https://github.com/Mottl/GetOldTweets3
[4] https://help.twitter.com/en/glossary
[5] https://github.com/PyMySQL/PyMySQL

allowed us to use the download() function to download stock information following certain parameters, including start date, end date, interval, and respective stock ticker. The download() function returned the data to a `pandas` dataframe. Using the respective commands from the `pandas` package, the stock data was filtered and concatenated into a single csv file to allow for easy reading in the website using d3.dsv().

One small obstacle was the restriction that 1 minute data could only be collected in one week intervals (one week in between start and end dates) and could only go back 30 days from the current day. Our workaround for the first problem was to download the data for the 30 days into several dataframes, then append the dataframes for each individual stock into a single dataframe column before finally concatenating the dataframes from different stocks.

3.1.3 NLP training data. Our objective from the point of view of data analytics was to perform sentiment analysis on tweets regarding stocks. In order to do so, we needed to have a predictive model which was already trained on a separated labeled dataset. Therefore we found and downloaded the popular dataset "Sentiment140" [4]. It is made of 1,600,000 tweets labeled as Positive, Neutral or Negative, and these were collected via the Twitter API and automatically labeled through the presence of happy/sad emojis.
Before being inputted into the NLP model, both the training dataset and the tweets that we collected needed to be preprocessed. The class of Neutral tweets was excessively unbalanced with respect to the other two that we decided to discard it.

3.2 NLP Analysis

3.2.1 Preprocessing. Once we had the collected tweets and the training dataset we had to apply some steps of preprocessing before using them as input in the NLP sentiment analysis model, as in [3]. The raw text of the tweet right out of the API is not clean, it might have misspelled words, it contains tags and hyperlinks, as well as emojis. For this process, we used the python library `nltk` [2], which provided us with useful functions to perform the transformation.
First, the tweets' text has been tokenized, that is, has been split into individual words. Second, each word that isn't relevant is removed (links, tags, common words) and reduced to its base form. The way we do this is by predicting the POS (Part-of-speech) tag of the word in the sentence and through that being able to reduce it to the normal form via WordNetLemmatizer.
After this, we proceed by eliminating those words from the sentence which are stopwords (such as articles or pronouns). The processed text result for each tweet is a sequence of words, which sometimes have lost syntactical structure but still maintain meaning and transmit the same idea as the original text. These cleaned tweets' text are the input for the Machine Learning model. However, they cannot be inputted as sequence of words but they need to be converted to numerical form. This is why we apply a vectorizer technique called tf-idf, which maps each word in the training set into a number. This number is calculated based on how many times the word appears in total in the training sample (tf) and also how many times that word is present in different documents, in this case, in different labels (idf). The resulting number will be greater if the words appear in a lot of samples, but also smaller if the word is present in many (or all) categories of samples, such as, in this case, both positive and negative classes. This technique highlights words that are supposed to be indicators for one class, when they are frequently present in samples of that class with respect to the other.

3.2.2 Training. Regarding the choice of the type of model best suitable for sentiment analysis, we explored two main types: a Naive Bayes classifier and a Logistic Regression. We trained both models, one at the time on the "Sentiment140" data, and we tuned their parameters through cross validation. The performances that we were able to get are very similar to each other with accuracy of NB at around 76%, while logistic regression performed at 78%. We chose however to go with Naive Bayes classifier, since it is a simpler model and because of its power in handling text data. [6]
The next step was to generate the predictions from the tweets that we collected and that were present in the cloud database. The tweets were retrieved using `mysql` queries from the database, went through the steps of preprocessing and then were given to the model as input. The model assigned a class to each one of them and the prediction was recorded. Once we had all the predictions for every tweet, for every company, for every day, we could proceed to calculating the hype score.

3.3 Hype Score

The computation of the hype score is entirely original from our reasoning and is performed as follows: for each day of the collected dataset, the tweets for each company are grouped together with their predictions from the previous step. Then, they are splitted into Positive and Negative classes, based on the predicted sentiment. For each class, we compute a summary hype score which is given by the sum of: the total number of retweets in that class, plus the total number of favourites in the class and then the count of tweets in the class. Therefore we have computed two scores, a positive hype and a negative hype. We subtract the negative from the positive and we apply a logarithmic transformation to shrink the absolute value of the hype score, while keeping the sign equal ($\log_2(hype)$ if $hype > 0$, $-\log_2(-hype)$ if $hype < 0$, and it remains 0 if $hype = 0$).
Doing this, we have compute a single hype score per company per day. Now we can save it in a file and we can proceed to visualize the score vis a vis the stock market data.

3.4 Website

Several ideas with the website were explored. Originally, we had a website where the stock price over time was plotted as a line graph, with bar graphs representing the hype score for the corresponding day is plotted behind the line graph. The chart data also provided an interactive component. When the user hovers the mouse over a point on the stock data time series plot, the bar corresponding to the hype score of that day would be highlighted. This first implementation is shown in figure 1:

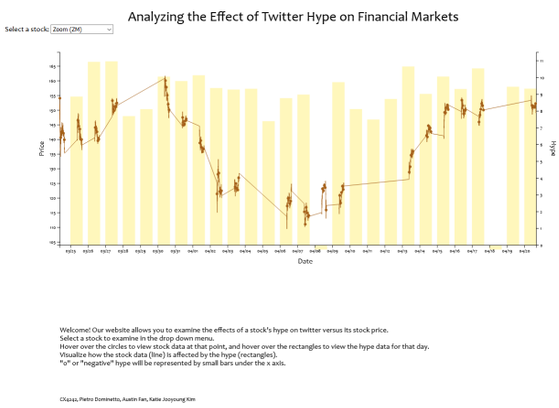

Figure 1: First Website Iteration

However, we realized that this method of visualizing the data is not optimal, as this method fails to clearly convey our original postulation for the hype's effect: the hype should affect the change in stock price for the short-term, such as the very same day or the next day. Therefore, we, made several changes:

- Instead of make the line graph the stock price at certain points, make the line graph represent the percent changes in stock price for that day
- More clearly display negative vs positive hype by allowing the x axis to be fluid, intersecting both axes at 0.
- More clearly display negative vs positive hype by making negative hype a darker red shade

These changes are shown in figure 2:

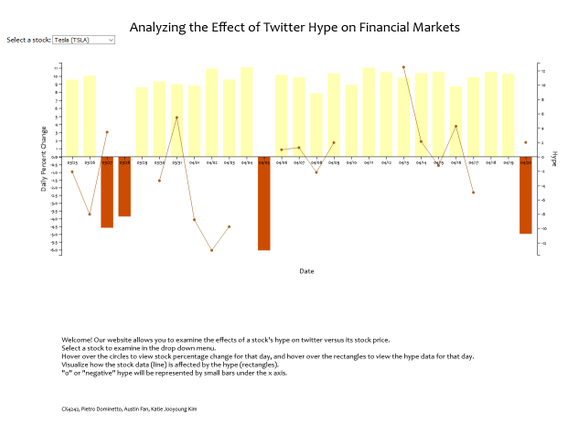

Figure 2: Second, Final Website Iteration

Holes/breaks in the line were due to weekends or federal holidays. Several more features were included in the website, including hover functionality for examining the data more closely (explained later).

The data was made using HTML, CSS, Javascript, and D3, a Javascript library, while taking in outside stock and hype data in csv formats.

4 EXPERIMENTS

4.1 Observations

To what extent does short-term hype on Twitter about specific companies and related equities influence the stock prices of those companies on a day-to-day level?

This question initially sparked our project, and to find our own answers to the question, we used both visualisation and calculations. Once we used the NLP analysis detailed in section *3.2*, we had two sets of numbers: the numeric stock data and the Twitter data transformed into numeric hype scores. As we had two measures that are not on the same unit or scale, we needed a more intuitive way to interpret the data. In order to achieve this, we decided to implement interactive visualisation.

4.2 Visualisations

As in *3.4*, after this first visualisation, we realised that it would be more pertinent to have time series data that emphasises the daily change in stock prices. This was because of our initial investigation question: if daily hype plays a role in influencing the stock prices, higher hype would lead to a larger change in prices, either up or down. Because absolute stock prices for each stock is highly dependent on external factors, it would be incorrect to hypothesise a direct relationship between the current stock price and the hype score.

With the new visualisation in figure 2, we had less intuitiveness but higher interpretability with regards to our research question.

5 DISCUSSION

We encountered many challenges throughout the project, and they provided much food for thought, both in cases when we managed to figure out solutions and when we had to accept the problem. An instance of the latter was when we first started collecting Twitter data. While many company names were discussed with regards to the market sentiment towards the stock, especially in the case of pharmaceuticals, certain company names turned out to be way too commonplace. That is, for the companies Google and Netflix, there was always a large volume - more than 1000 collected per day - of tweets, making it impossible for us to determine changes in volume. This was attributed to the commonness of both names in colloquial speech: "to google" has become a synonym for "searching online", and Netflix has also become a common enough household activity to have many discussing it on Twitter, regardless of the stock price indices.

The solution to this problem would be to scale the

Another challenge that we face was how to create the hype score from the sentiment predictions. We decided that it would be a combination of favourites, retweets and volume, since those are the characteristics that we think of when we describe the concept of hype. Needless to say, the entire project and its conclusions are based on how we define this measure. A slightly different definition could produce very different results, and therefore change completely the correlation with the stock data. Unfortunately, we didn't have time to investigate deeper into the matter. We tried to play a little bit with the weights of the hype score functions (how much importance to give to favourites, retweets, and volume) but we didn't find anything consistent, so we decided to leave them as we thought them. In case the project might be taken over in the future, we suggest to explore in depth this concept.

REFERENCES

[1] Sitaram Asur and Bernardo A. Huberman. 2010. Predicting the Future with Social Media. *CoRR* abs/1003.5699 (2010). arXiv:1003.5699 http://arxiv.org/abs/1003.5699

[2] Steven Bird, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python.* O'Reilly Media Inc.

[3] Edilson Anselmo Corrêa Júnior, Vanessa Queiroz Marinho, and Leandro Borges dos Santos. 2017. NILC-USP at SemEval-2017 Task 4: A Multi-view Ensemble for Twitter Sentiment Analysis. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)* (2017). https://doi.org/10.18653/v1/s17-2100

[4] Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *Processing* 150 (01 2009).

[5] Niels Joachim Gormsen and Ralph S. J. Koijen. 2020. *Coronavirus: Impact on Stock Prices and Growth Expectations (April 8, 2020).* University of Chicago, Becker Friedman Institute for Economics Working Paper. http://dx.doi.org/10.2139/ssrn.3555917

[6] Daniel Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (1st ed.). Prentice Hall PTR, USA.

[7] F. Kaske, M. Kugler, and S. Smolnik. 2012. Return on Investment in Social Media–Does the Hype Pay Off? Towards an Assessment of the Profitability of Social Media in Organizations. In *2012 45th Hawaii International Conference on System Sciences*. 3898–3907. https://doi.org/10.1109/HICSS.2012.504

[8] John R. Nofsinger. 2005. Social Mood and Financial Economics. *Journal of Behavioral Finance* 6, 3 (2005), 144–160. https://doi.org/10.1207/s15427579jpfm0603_4 arXiv:https://doi.org/10.1207/s15427579jpfm0603$_4$

[9] Jianfeng Si, Arjun Mukherjee, Bing Liu, Qing Li, Huayi Li, and Xiaotie Deng. 2013. Exploiting topic based twitter sentiment for stock prediction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 24–29.