# Sentiment Analysis and Stocks

Dr. Khalaj


Department of Electrical Engineering

Arad Mahdinejad Kashani 400102028

Project Report
Foundations of Data Science

February 6, 2024

# Sentiment Analysis and Stocks

Project Report

Arad Mahdinejad Kashani 400102028

## Introduction

I did use Git for this project. I have messaged you on Telegram so as to add you to my private repository. If needed, you can contact my Telegram at *@aradmnk*. Not much is happening, though. There are only 4 python notebooks and the .csv file for the web-crawler ('divar.csv', because I crawled 'divar.ir').

The process is also documented in the notebook files themselves. There is text, and the code is segmented by the tasks in each phase of the project, so hopefully it is readable.

# Phase 1

After reading the files, I made a realization that there exists a certain tag (referred to a **CashTag $** in the paper mentioned in the project documentation) in the df['text'] column. I used regex to separate them in a different column and used that from then on.

## *Task 1*

**Task**

Find the most and least tweeted stocks.

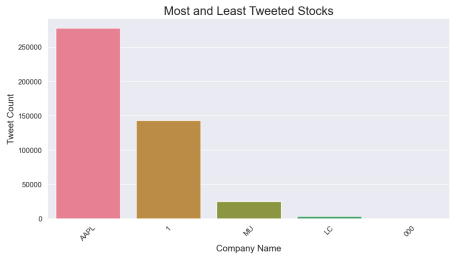
```
Most tweeted about:ticker          AAPL
tweet_count    277799
Name: 11288, dtype: object
Least tweeted about:ticker         000
tweet_count      1
Name: 3, dtype: object
```

Figure 1: Most and least stocks

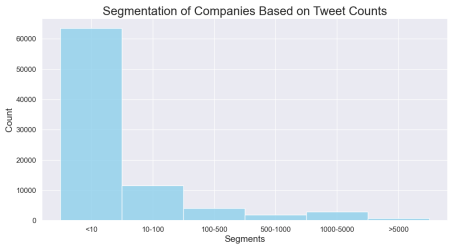
As we can see, the most tweeted about stocks is **Apple Inc.** ***(AAPL)***. The least tweeted about is irrelevant because there existed many tickers with only 1 tweet (I checked it manually, it is not included in the notebook).

**Task**

Segment the companies based on tweet counts.




(a) Some companies and their respective tweet counts




(b) Segmentation of the companies

Figure 2

The middle companies of *3.(a)* were hand picked from the middle, by index.

## *Task 2*

**Task**

Statistics on distributions of 5 individual stocks over time. Choose the individual stocks to perform reflect different sectors of the economy.

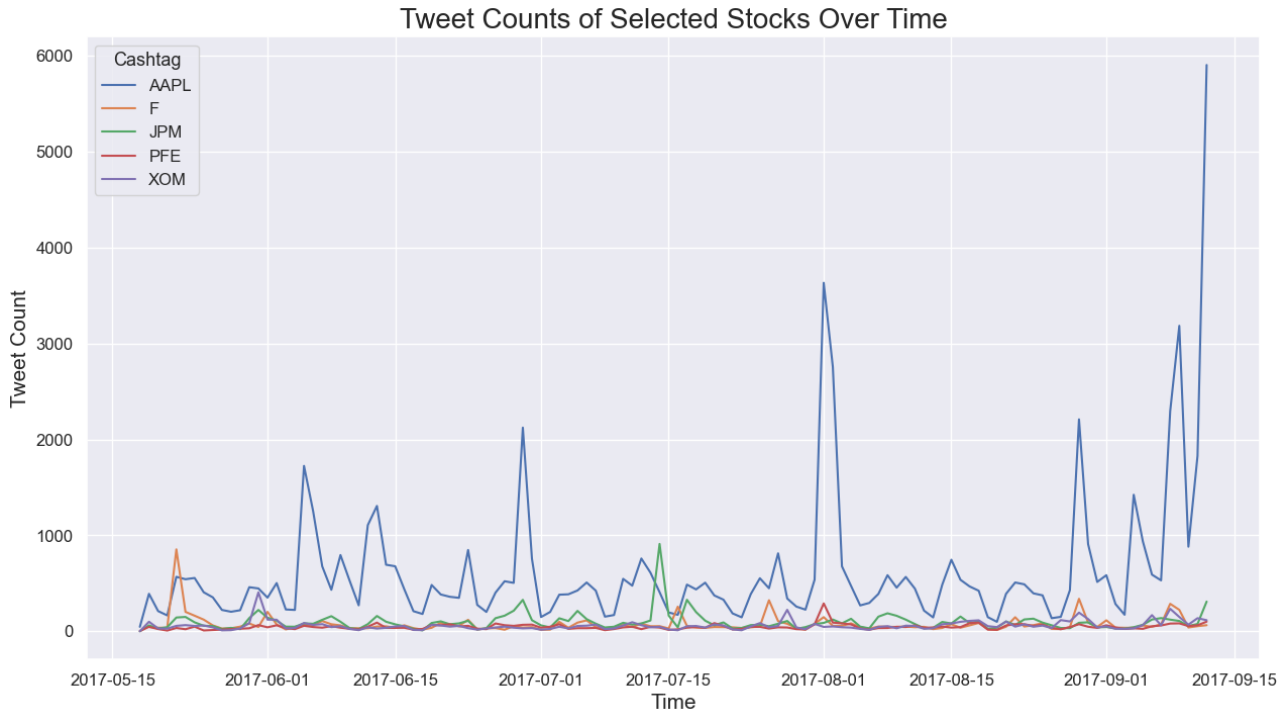



Figure 3: Individual stock distributions

I manually chose **Apple** (Technology), **ExxonMobil** (Energy), **Pfizer** (Healthcare), **JP-Morgan Chase** (Financial), and **Ford** (Automotive).

## *Task 3*

**Task**

Statitistics on distributions of all financial tweets over time.

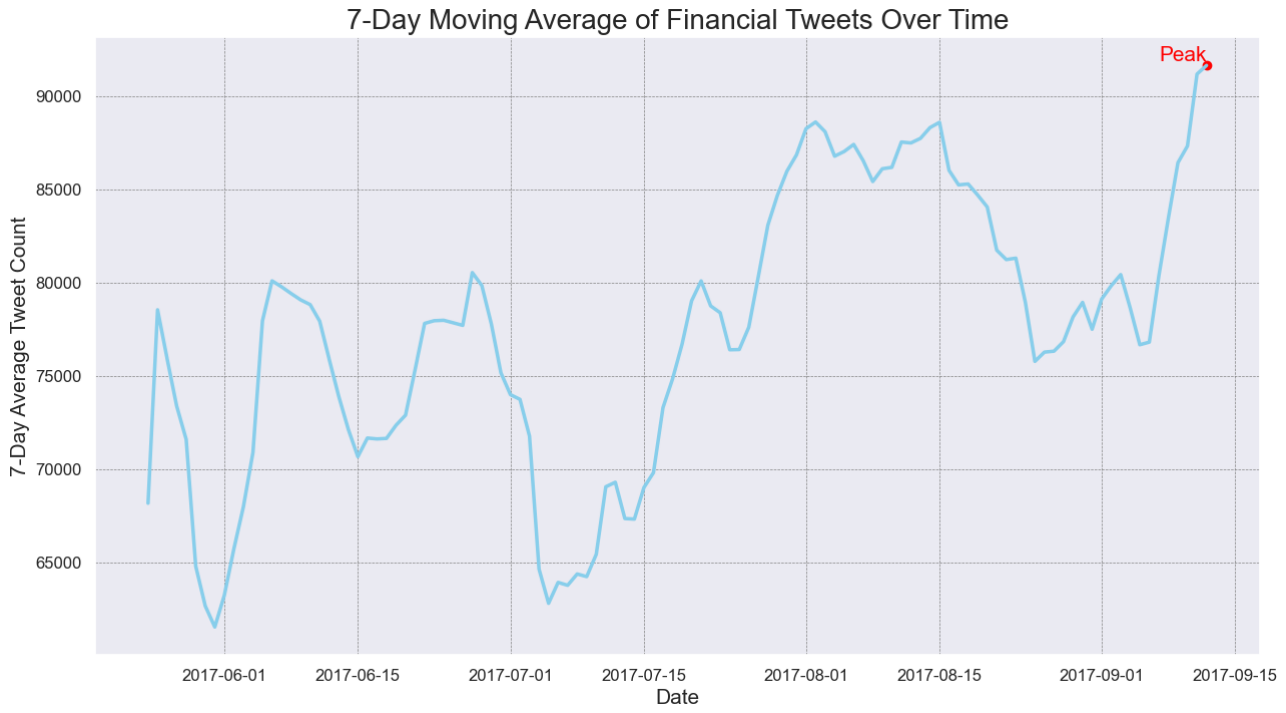



Figure 4: 7-day average of all financial tweets over time

For this one I took on a 7-day moving average approach, just for the sake of doing it.

## *Task 4*

**Task**

Statistics on distributions of retweets per tweets including individual stocks (at least 2 chosen stocks) over time.

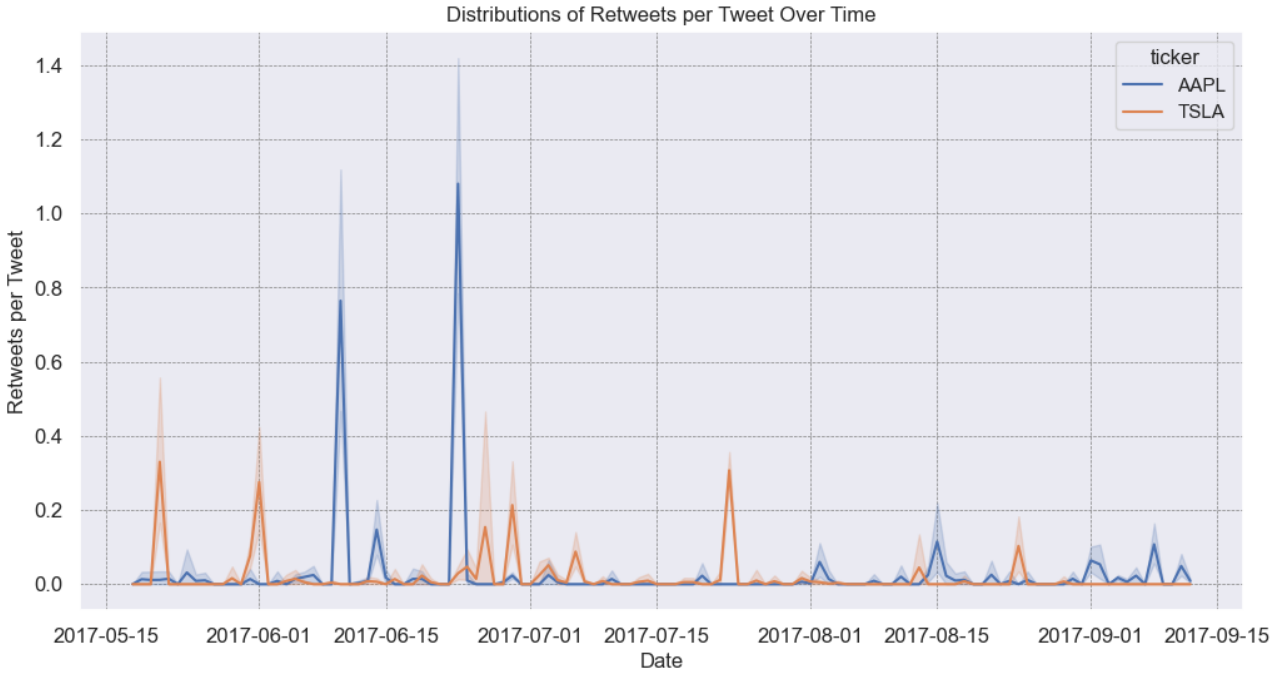



Figure 5: Distribution of retweets-per-tweet of **AAPL** and **TSLA** over time

I used Apple Inc. **(AAPL)** and Tesla Inc. **(TSLA)**, because I saw those in the demos and decided to use them for some reason.

## *Task 5*

**Task**

Statistics on most important financial information on individual stocks (at least 2 chosen stocks) computed solely from the financial information (not the tweets).

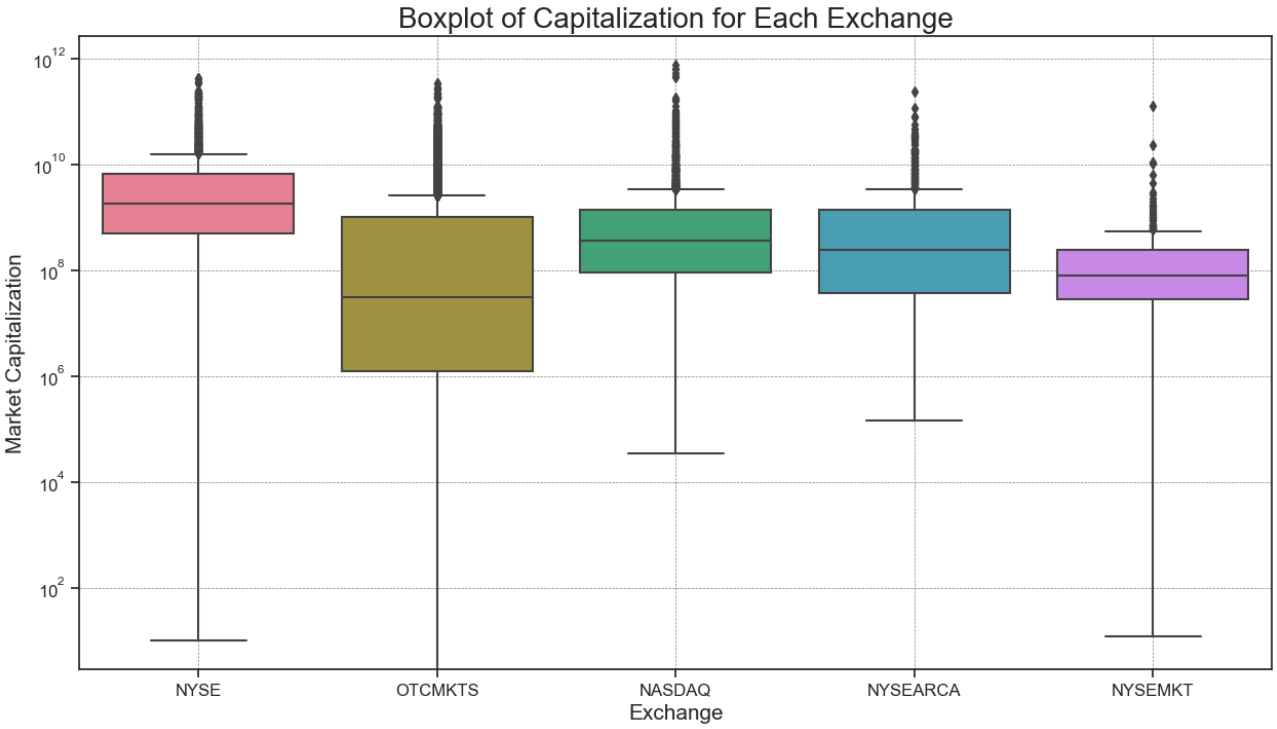
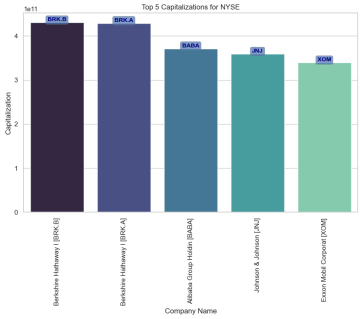
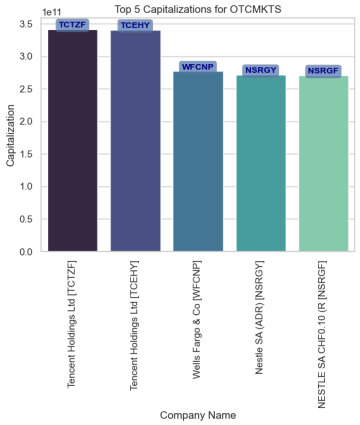



Figure 6: Box-plot of capitalizations per exchange (logarithmic)

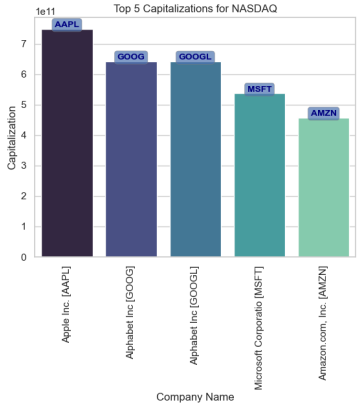
The data for the box-plots was obtained from the 'companies.csv' file.



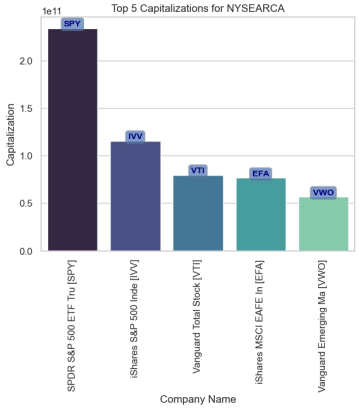

(a) NYSE



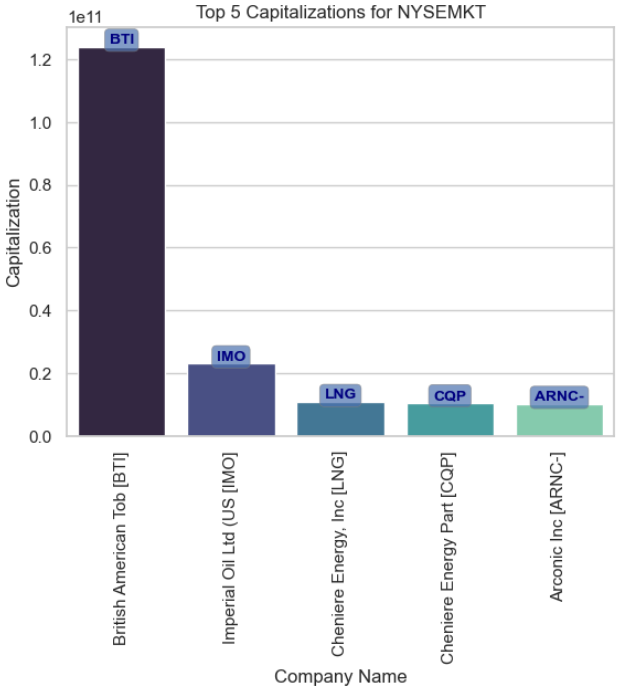

(b) OTCMKTS




(c) NASDAQ




(d) NYSEARCA



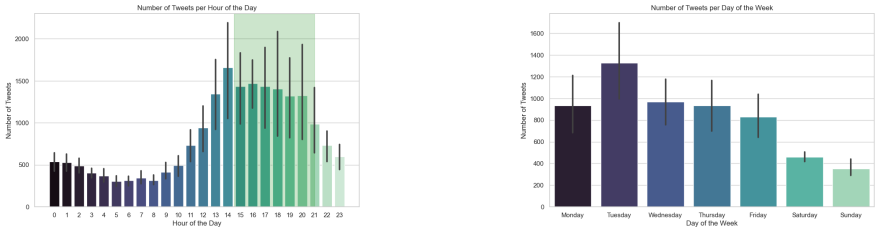

(e) NYSEMKT

Figure 7: Top 5 capitalizations for each exchange

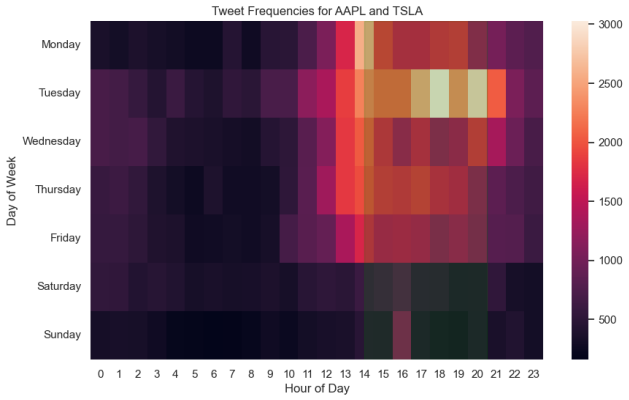
## *Task 6*

**Task**

Time series movement directions through time for individual stocks (at least 2). Choose companies you are familiar with. Try to explain the reason behind these directions from real world news.




(a) Number of tweet per hour of the day(b) Number of tweet per day of the week




(c) Tweet frequencies per day of the week and hour of the day

Figure 8: Time series movement for number of tweets

- In figure *8.(a)*, we can see a peak at the trading hours (highlighted green) and even an hour before trading hours. A psychological analysis may be that the traders will be checking tweeter an hour before trading to see how to market may be going, or maybe those interested in selling their shares may try to persuade buyers to buy their share of stock.
- In figure *8.(b)*, we can see a drop on the days off (Saturday and Sunday). Tuesday seems to have a peak as well, but I lack knowledge of the market to be able to analyize why!
- In figure *8.(c)*, we can see the joint distribution per day and week. The most frequent trading time is the trading hours in the working days. Tuesdays seem to be very heated (and I do not know why)!

## *Task 7*

**Task**

Co-occurence of various stocks in the same tweets.

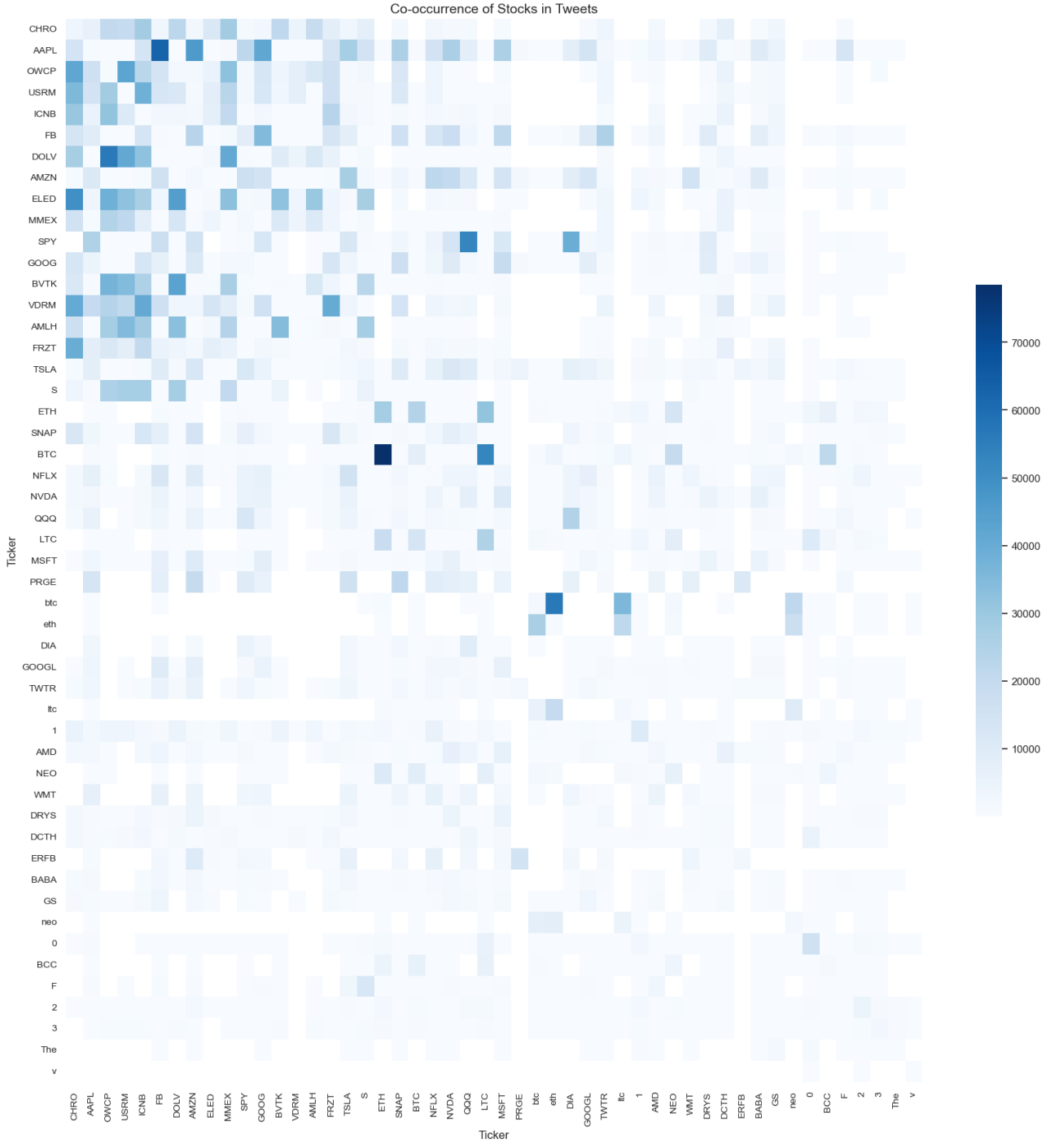



Figure 9: Co-occurence matrix of various stocks in the same tweet

The process is very time-consuming. I had to get the top 50 most common tickers to be able to computationally handle the task. It seems **ETH** and **BTC** occur the most together. Another contestant to that is **AAPL** and **FB**.

# Phase 2

After reading the data and processing the time-series data, I was ready to begin the training process. The problem was that the data size was too huge for me to work with properly (as I tend to do the work with trial and error), so I sampled the data and worked with a $N = 100,000$ sample dataframe. All of the models were trained and evaluated this way.

I also had to use **Kaggle** for this Phase (and the next), because my device did not have a usable GPU for training the models; I only had a GPU there. Hopefully the file directories will make more sense when you read the notebook now.

## *Cleaning the data*

Task

Clean the data.

- Removing duplicate values and useless data (both columns and rows).
- Handling upper/lower case, etc.

First the links were removed. Then, the hashtags were separated from the text data, and punctuation was removed from the tweet. All non-english characters were removed, and tokenized with the stop-words removed. The stop-words were obtained from nltk.stop_words.

## ***Bag Of Words***

### ***Support Vector Machine (SVM)***

| | |
|---|---|
| Train accuracy | 86.71% |
| Validation accuracy | 52.25% |
| Test accuracy | 51.77% |

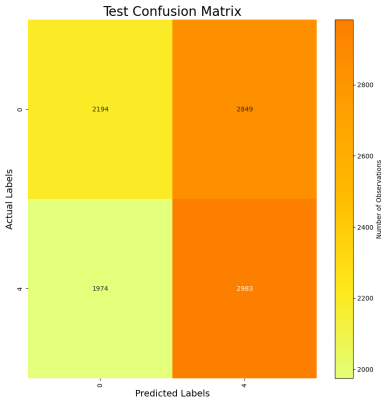



Figure 10: SVM + BOW Confusion Matrix

### ***Random Forest***

| | |
|---|---|
| Train accuracy | 96.43% |
| Validation accuracy | 52.12% |
| Test accuracy | 52.32% |

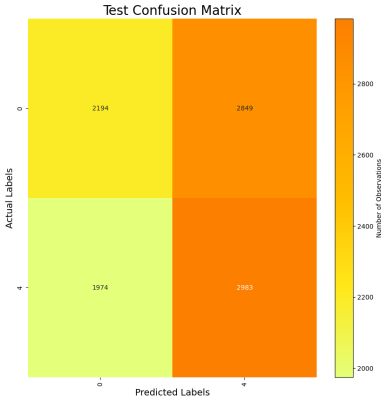



Figure 11: Random Forest + BOW Confusion Matrix

The Random Forest already looks more promising, judging by the confusion matrices. The main diagonal of the confusion matrix seems brighter. It seems sentiment analysis is not linearly separable (what a surprise).

## ***TF-IDF***

### ***Support Vector Machine (SVM)***

| | |
|---|---|
| Train accuracy | 94.92% |
| Validation accuracy | 77.62% |
| Test accuracy | 77.27% |

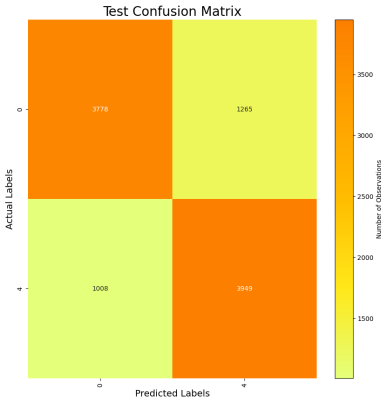



Figure 12: SVM + TF-IDF Confusion Matrix

## ***Random Forest***

| | |
|---|---|
| Train accuracy | 99.46% |
| Validation accuracy | 75.63% |
| Test accuracy | 75.54% |

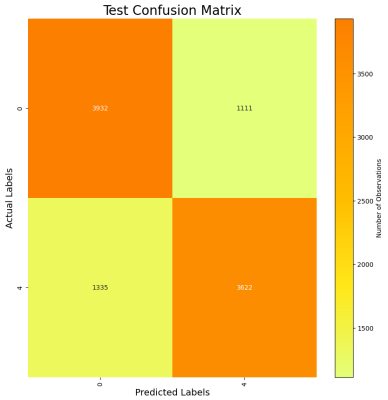



Figure 13: Random Forest + TF-IDF Confusion Matrix

The Random Forest already looks more promising, judging by the confusion matrices, as the main diagonal of the confusion matrix seems brighter. TF-IDF is already performing better than the BOW method.

### ***SpaCy***

There is not much to say here really, except that I used the 'textcat' pipeline, already pre-prepared for our purpose (binary text-classification).

```
Validation Set Performance:
              precision    recall  f1-score   support

    NEGATIVE       0.79      0.79      0.79      4980
    POSITIVE       0.79      0.79      0.79      5020

    accuracy                           0.79     10000
   macro avg       0.79      0.79      0.79     10000
weighted avg       0.79      0.79      0.79     10000

Test Set Performance:
              precision    recall  f1-score   support

    NEGATIVE       0.78      0.79      0.78      4965
    POSITIVE       0.79      0.78      0.78      5035

    accuracy                           0.78     10000
   macro avg       0.78      0.78      0.78     10000
weighted avg       0.78      0.78      0.78     10000
```

Figure 14: SpaCy classifier metrics

### ***Fine-tuning HuggingFace: Distil-BERT***

For the next section, I chose not to use the GPT API (because I did not have enough time to calculate the token and chose to accept risk) and instead used a fine-tuned version of the Distil-BERT model from HuggingFace. The reason I used SpaCy, was the fact that I did not like having two BERT models for this project.

| | |
|---|---|
| Train accuracy | 99.06% |
| Validation accuracy | 78.95% |
| Test accuracy | 77.85% |

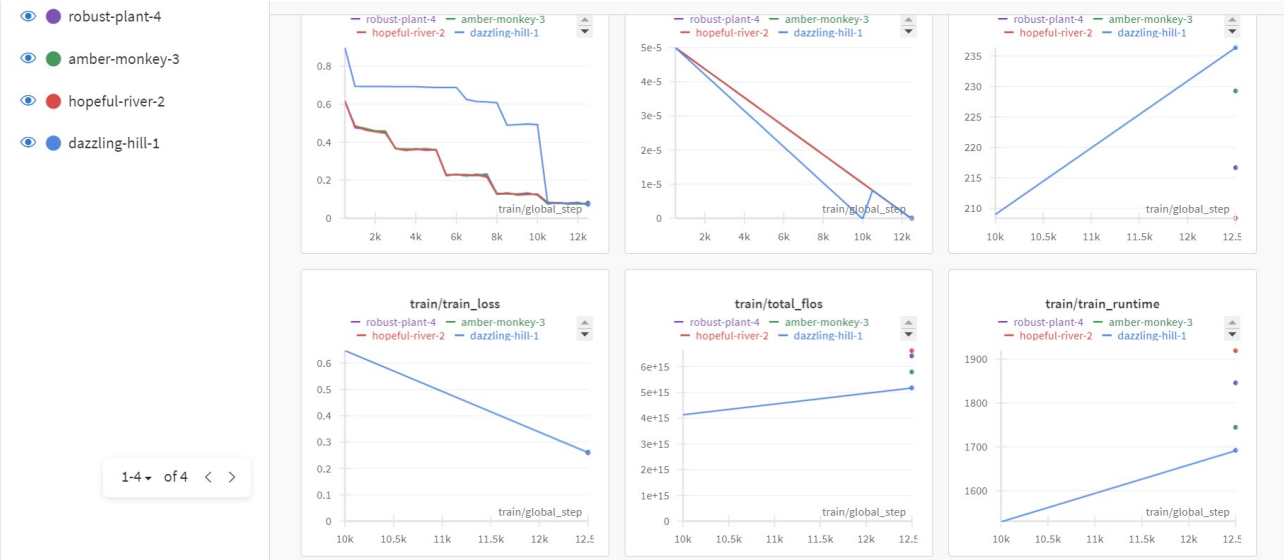



Figure 15: HuggingFace workspace dashboard (robust-plant-4 is the final model, purple)

The SpaCy model and the Distil-BERT model were saved and used for the next phase, as they had the best results.

## Phase 3

# End of Project Report