Final Project Report

ECE1786

Tim-Cheng Gu
cheng.gu@mail.utoronoto.ca

Weijin Chen
wenjin.chen@mail.utoronto.ca

GiHub:https://github.com/ece1786-2022/HiveMindInvestor

Dec 13, 2022

Word Count: 1987 Penalty: 0

1 Introduction

It is well known that stock price fluctuations are influenced by a wide range of factors, one of which is market sentiment. However, manually reviewing and analyzing large quantities of market sentiments can be taxing and expensive. An alternative method is to use machine learning based sentiment analysis. This method will be utilized to extract and quantify the emotional content of the text data, allowing us to identify the overall sentiment of a given Reddit post as positive, or negative. Hence, the goal of this project is to use sentiment data from the previous week's Reddit posts to predict stock trends for the following week, with a focus on six specific stocks: Amazon, Apple, Meta, Tesla, and Netflix. The hope is to gain valuable insight into the general sentiment surrounding these companies via machine learning and potentially use this information to make accurate predictions about their future performance.

2 Background and Related Work

There exists previous work on applying NLP for reddit sentiment analysis for stock investment purposes [1]. The paper uses spaCy for word embedding and uses Native Bases, Logistic Regression, and Convolutional Neural Networks as their main model architecture for sentiment analysis. While their method was able to produce a relatively positive result, it reported some major challenges such as difficulty training the spaCy embedding.

Another work [2] uses traditional time-series analysis methods such as ARIMA and LSTM RNNs, in conjunction with LSTM based sentiment analysis based on tweets and reddit posts, to predict bitcoin prices. Bitcoin prices share many similar features as volatile stock prices, and the work is able to generate a very close prediction to the actual price change.

Neither of these methods explicitly uses transformers as their driver for the NLP sentiment classification part, which may limit the performance of their respective models. We aim to improve the performance of the methods above by using a transformer based sentiment analysis model.

3 Data Processing

Three datasets were used in total in this project. The first dataset is the Reddit Wallstreetbet dataset containing 6918 Reddit posts and comments, manually crawled from the subreddit r/wallstreetbets using keyword searches. Each stock has 700-1700 data points. There are 6 fields, title, post text, ID, URL, date, score, and upvote ratio for each datapoint. We cleaned data by removing special characters and URLs. The first 200 data points for each stock were manually labeled 1 for positive, 0 for negative, and -1 for unrelated, and used to fine-tune a sentiment model. An example of the data is shown in Figure 1. The distribution of labels is show in Figure 2. The labeled data is split into train-test ration 0.7/0.3 for fine-tuning.

Label	Post_Concat
-1.0	amazon is working ontv series about ftx and it...
1.0	bb quietly building towards the inevitable sof...
-1.0	upcoming department of defense contract itspos...
0.0	non residential construction forecast slowing ...
0.0	trading volume in put options for the consumer...
...	...
0.0	when didtry options hint aapl amzn amd tmussel...
1.0	amazon yolo no dd just regarded
0.0	to watch this week american indices around par...
0.0	yahoo finance big tech earnings and gdp data w...
-1.0	i spy ta monday october 24 2022 0dte scalpers ...

Figure 1: The data example of the labeled Reddit sentiment dataset

	AMZN	AAPL	GOOG	META	NFLX	TSLA
0	51.55	42.53	35.05	41.11	22.0	38.79
1	24.84	32.76	28.35	21.11	36.0	26.67
-1	23.60	24.71	36.60	37.78	42.0	34.55

Figure 2: Percentages of each label in each stock

The second dataset is the tweet stock sentiment dataset, which contains two .txt files with tweets of either bullish or bearish sentiment. There are 707 bullish tweets and 600 bearish tweets. The tweets were labeled as 1 for positive sentiment and -1 for negative sentiment, then split into training and test sets with a ratio of 0.7/0.3. Text preprocessing was applied to the tweets in the same way as to the Reddit dataset. An example is shown in Figure 3

	tweet_text	Sentiment
0	adding	1
1	we like to be early before explosions	1
2	added now just need patience	1
3	funny how these biotech washouts always come b...	1
4	looks like it wasnice dip buy yesterday	1
...	...	...
1302	aapl back at lows	0
1303	only peopleballs of steel are shorting aapl here	0
1304	fb goes red	0
1305	aapl down because its moved	0
1306	aapl flat the short stocks	0

Figure 3: The data example of the tweet stock sentiment dataset

The third dataset is 2 years of historical stock closing prices for selected stocks, from January 2021 to November 2022. The data was collected from Yahoo Finance and downloaded as .csv files, with 504 rows and 7 features per file. The dates were formatted as "YYYY-MM-DD" and the data was loaded into a pandas DataFrame for use in downstream models. Figure 4 shows an example of data, and Figure 5 shows the statistics.

	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-12-01	121.010002	123.470001	120.010002	122.720001	121.264313	127728200
1	2020-12-02	122.019997	123.370003	120.889999	123.080002	121.620026	89004200
2	2020-12-03	123.519997	123.779999	122.209999	122.940002	121.481689	78967600
3	2020-12-04	122.599998	122.860001	121.519997	122.250000	120.799881	78260400
4	2020-12-07	122.309998	124.570000	122.250000	123.750000	122.282082	86712000
...	...	...	...	...	...	...	...
499	2022-11-23	149.449997	151.830002	149.339996	151.070007	151.070007	58301400
500	2022-11-25	148.309998	148.880005	147.119995	148.110001	148.110001	35195900
501	2022-11-28	145.139999	146.639999	143.380005	144.220001	144.220001	69246000
502	2022-11-29	144.289993	144.809998	140.350006	141.169998	141.169998	83763800
503	2022-11-30	141.399994	148.720001	140.550003	148.029999	148.029999	111224400

Figure 4: The data example of processed historical stock price data of AAPL

	AAPL	AMZN	GOOG	META	NFLX	TSLA
count	495.000000	495.000000	495.000000	495.000000	495.000000	495.000000
mean	147.404242	150.725803	120.546328	260.126526	433.408586	265.459677
std	16.045988	23.350867	18.185350	78.367618	156.529466	51.054189
min	116.360001	86.139999	83.489998	88.910004	166.369995	177.589996
max	182.009995	186.570496	150.709000	382.179993	691.690002	409.970001

Figure 5: The properties of closing stock price data of all six stocks

4 Architecture Diagram

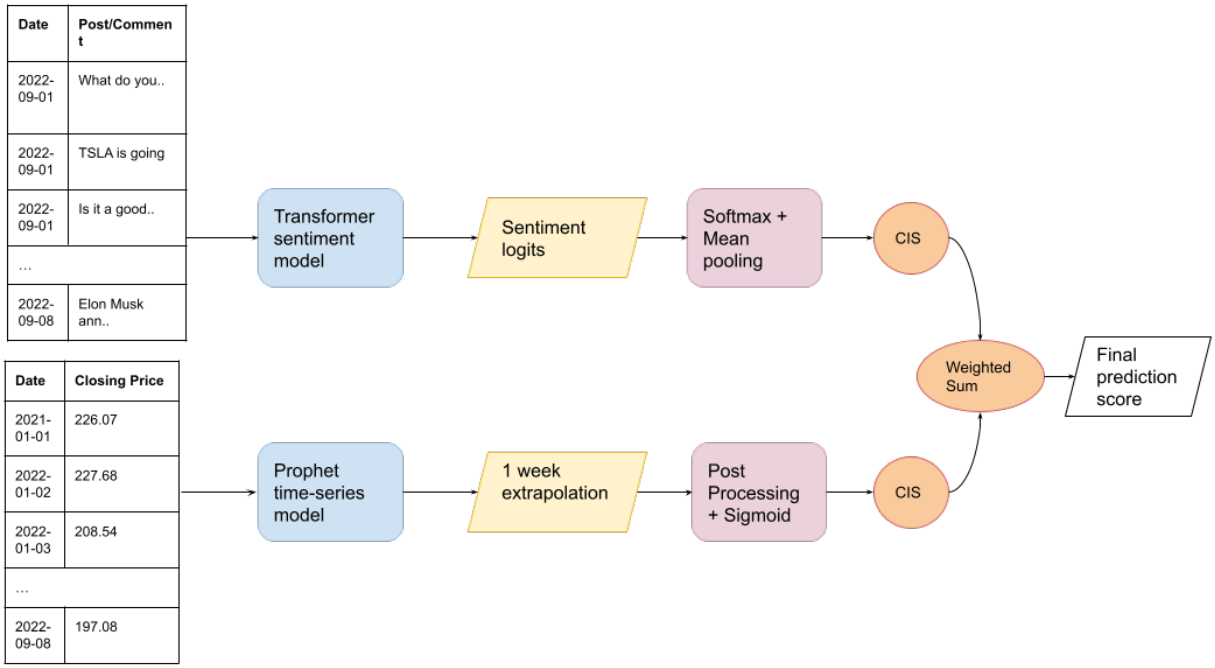

Figure 6: Overall architecture of our solution for one target period, for the Tesla stock.

5 Architecture and Software

The proposed architecture for our stock prediction system, as shown in 6 consists of two components: a transformer-based sentiment model and a time-series model powered by Meta Prophet. The goal is to use the output of the time-series model as a starting point and construct a more informed prediction with the results from the sentiment analysis.

For the time-series component, we use Meta Prophet, a powerful time-series forecasting model that is particularly well-suited for stocks and other financial data. We use data from January 1, 2021 up to the start of the target week as input to Prophet and extrapolate one week of predicted stock trends. To post-process the output, we have simply taken the difference between the closing price at the start and end of the target week and fed the result through a sigmoid function to obtain a final sentiment score between 0 and 1. A score greater than or equal to 0.5 indicates that the stock is expected to rise, while a score less than 0.5 indicates that it is expected to fall.

For the sentiment model, we choose the JulienSimon model, a pre-trained binary sentiment classification model from Hugging Face based on the DistillBert model. The model has 6 transformer blocks, each consisting of a multiheaded attention layer followed by 2 fully connected layers, for a total of approximately 40 million parameters. We use comments and posts from Reddit collected within the one-week period prior to the start of the target period as input to the sentiment model. We have passed the logits output for each post/comment through a softmax function and performed mean pooling on the values for positive sentiment across all posts/comments to obtain a final sentiment score between 0 and 1. As with the time-series model,

a score greater than 0.5 indicates that the model predicts that the stock will rise, and a score less than 0.5 indicates that the stock will fall.

To combine the two scores, we considered using a simple mean pooling method. However, this approach is not effective because the magnitudes of the two scores have vastly different meanings. For example, a 0.75 sentiment score and a 0.75 time-series score do not necessarily imply that the two models have equal confidence in their predictions. Another option is to use a weighted average, but this is also not sufficient because a weighted average is a linear combination of the two scores, yet the relationship between scores and confidence is not necessarily linear.

To address this issue, we pass both scores through a Cubic Inverse Sigmoid (CIS) function. This projects the values back to the infinite linear space, allowing us to linearly combine them as a weighted sum. The final output of the model is a real value between negative and positive infinity. A value greater than or equal to 0 indicates that the model predicts the stock price will rise, otherwise it will fall. The final combination equation is shown in the following.

$$\textit{Final Prediction Score} = \alpha * CIS(\textit{time-series score}) + \beta * CIS(\textit{sentiment score})$$

Where

$$CIS = -log\left(\frac{1-x}{x}\right)^3$$

$$\alpha, \beta \textit{ are hyperparameters}$$

6 Baseline Model

The purpose of comparing the Monte Carlo and Momentum methods to our proposed method is to provide a baseline for evaluating the performance of our approach. We have chosen these methods because they represent simple strategies that do not require any expertise in investing.

The Monte Carlo method is based on performing a binary random guess to determine whether an investment should be made. We conduct a series of trail runs, where we perform a guess for each 1-week period across all 11 target periods. We conduct a total of 100 trail runs and take the mean performance as the final performance.

The momentum method is another simple approach that assumes the trend of the target period will be the same as the week before it. For example, if the previous week's stock price rises, the momentum method assumes that the next week's prices will also rise.

7 Quantitative Results and Discussion

To evaluate the model's performance, we compared its predictions with the actual closing price data for the target period. We tested the model over a period of 11 weeks between September 8th and November 24th, and calculated the accuracy by dividing the number of correctly predicted weeks by the total number of weeks. We compared the performance of our model with four other models, including time-series prediction and sentiment analysis, as well as the Monte Carlo method and Momentum. The results, as shown in Figure 7, show that our model performed best overall compared to the other methods, only losing to pure sentiment model on the tesla stock.

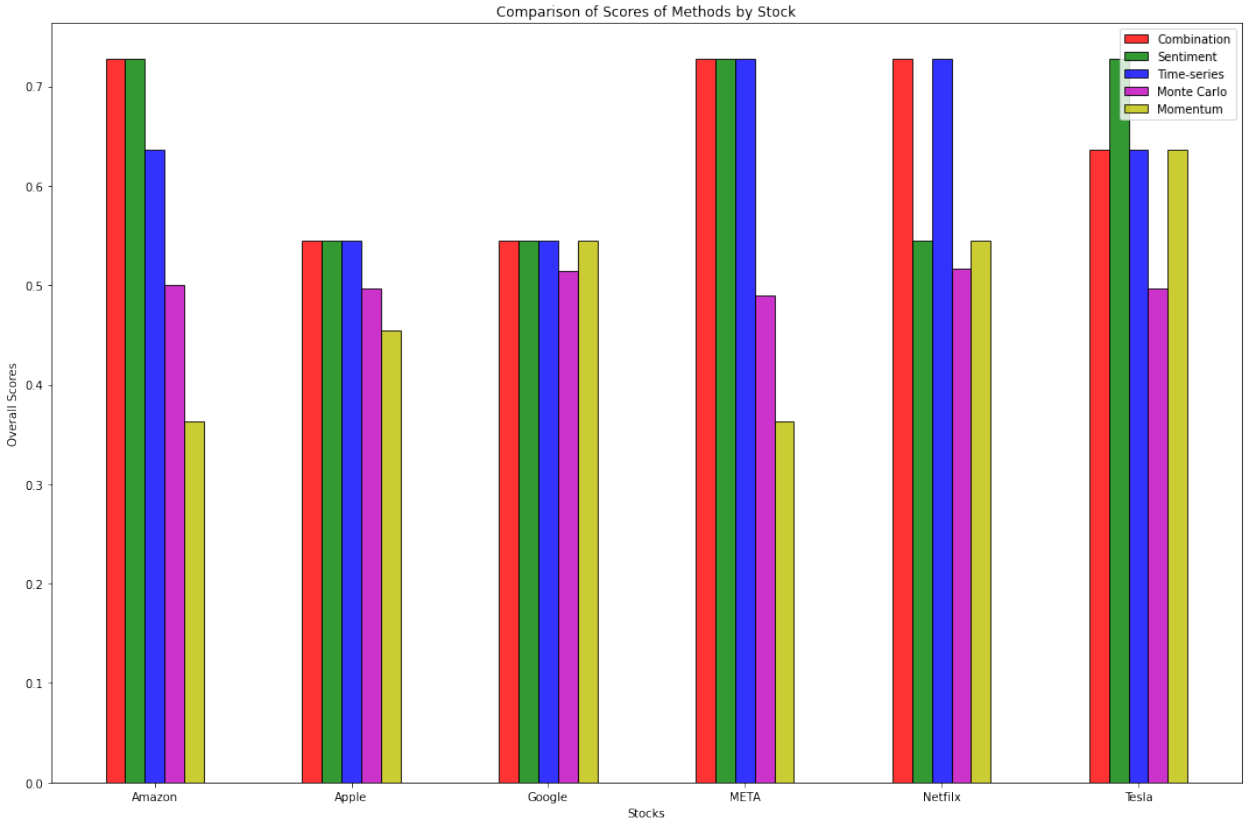

Figure 7: Overall model accuracy between all models for 6 stocks

To further evaluate the effectiveness of our model, we conducted a simple investment simulation. During each week of the simulation, we used the output of the model to determine whether or not to invest. If we are investing, always invest $5,000 in stocks at the beginning of the week and sell all of the stocks by the end of the week. We then sum up the net gain across all 11 weeks. The results of the simulation in Figure 8 showed that our model performed well for practical investment purposes, netting a positive of $1500.

	names	Monte Carlo	Momentum	HiveMindInvestor
0	AMZN	-944.431562	-672.831334	44.823519
1	AAPL	-217.580525	-134.601410	67.684407
2	GOOG	547.007556	-529.345704	0.000000
3	META	-69.456735	-2000.586913	0.000000
4	NFLX	360.342641	796.963004	1362.544097
5	TSLA	-428.249592	-216.090454	25.596542

Figure 8: Total sum of investment simulation across 11 weeks

8 Qualitative Results and Discussion

To gain more insights on how the models perform for each individual prediction period, we look at the raw prediction values, which are the model output normalized against the actual stock difference between the end and the start of each target period, for each stock. We calculate the values based on the following

equation:

$$Raw\ Prediction = \frac{Model\ Output}{Actual\ Stock\ Difference}$$

To evaluate the performance of our model, we compare the predicted stock price changes with the actual stock price changes. Since both the model output and the actual stock difference have a range between positive and negative infinity, we used the sign of the values to determine if the prediction was correct or not. This allowed us to easily evaluate the accuracy of the model.

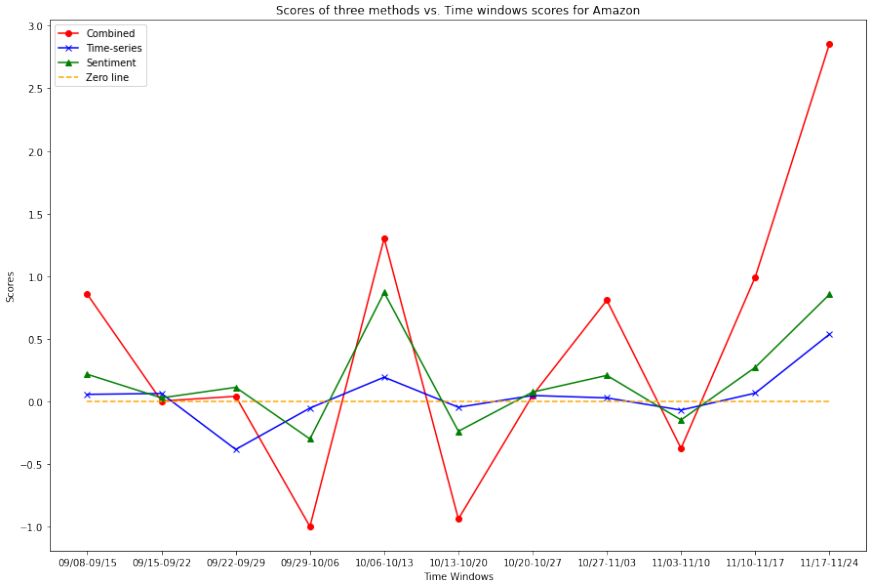

Figure 9: Raw predictions for Amazon

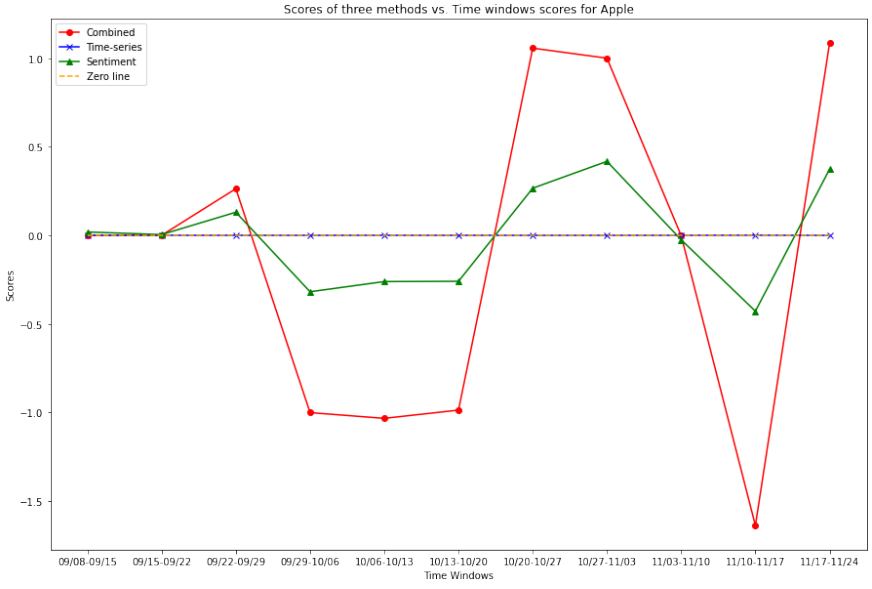

Figure 10: Raw predictions for Apple

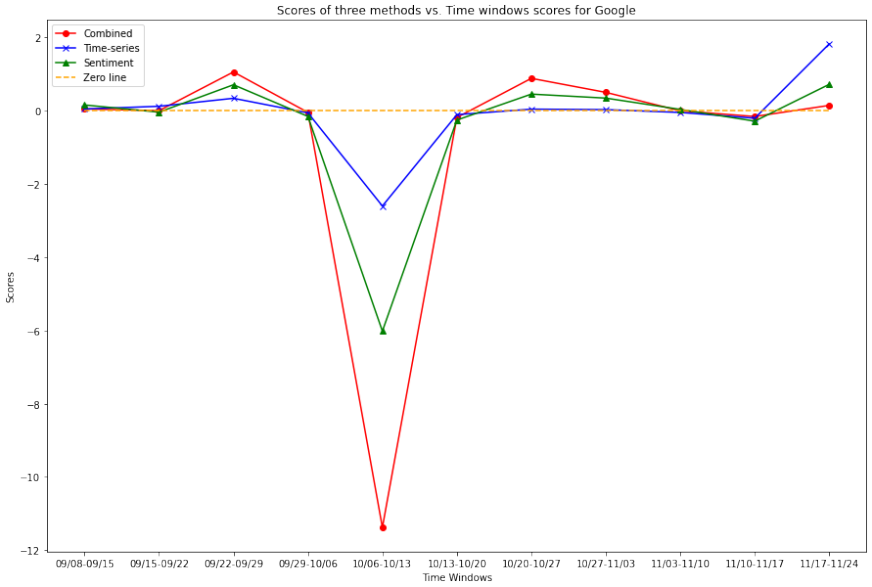

Figure 11: Raw predictions for Google

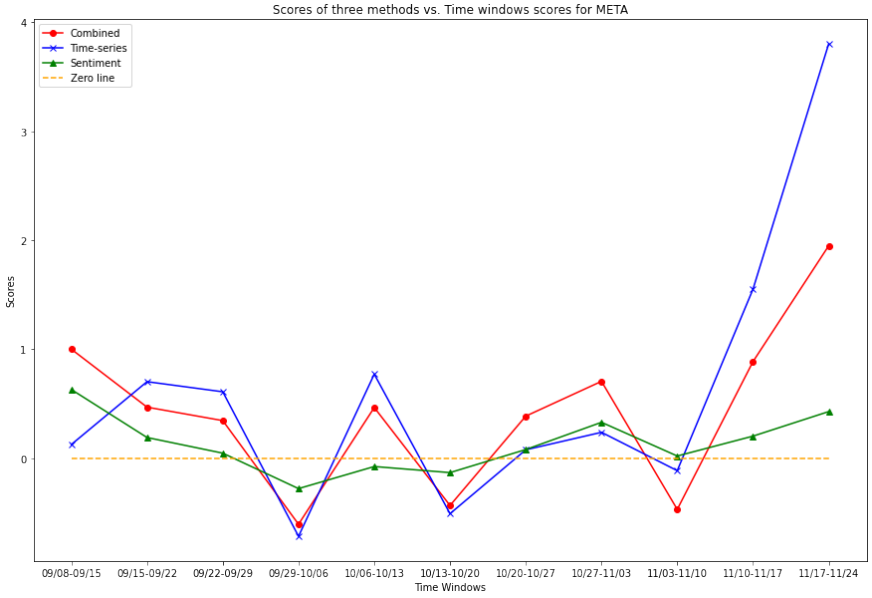

Figure 12: Raw predictions for Meta

Figure 13: Raw predictions for Netflix

Figure 14: Raw predictions for Tesla

We observed that for different models, the raw magnitudes of the time-series prediction and sentiment analysis can vary greatly. Furthermore, we observed that the time-series model and the sentiment model were surprisingly consistent with each other: when one model was correct, the other tended to also be correct, and when one was wrong, the other was likely to be wrong as well. This was particularly prominent for Google, where the change in magnitude between the sentiment and time-series model almost completely mimicked each other. One hypothesis is that a lot of the sentiments for stocks are based on time-series predictions. If the stock is projected to fall, sentiment is likely to be negative, although further investigation is needed to validate this theory.

9 Discussion and Learning

During this project, we learned a few key lessons. The first lesson is to not make the architecture overly complex. The original architecture in the project proposal had many moving components, one of which was to combine sentiment scores for comments and posts as a weighted sum with the number of likes for each post or comment. We quickly abandoned this idea because it significantly complicated the logic of the architecture, and it was difficult to obtain the number of likes through the Reddit API.

The second lesson was that sentiment analysis does not always have to be binary. During our labeling process, we noticed that many posts were not related to the specific stocks we were interested in. Although we excluded these data from the training process, we did not have the bandwidth to exclude them from the testing set. This may have had a negative impact on the accuracy of the model. Allowing the sentiment analysis model to determine this automatically might have helped solve this problem.

10 Individual Contributions

The contributions of Tim-Cheng Gu:

- Labeled 600 Reddit sentiment data.
- Implemented the Prophet model.
- Created the wrapper classes for the Prophet and Sentiment model.
- Designed and implemented the combination logic.
- Created the test bed for the combined model.
- Implemented the baseline methods.
- Implemented the tests.
- Hand-tuned all the models.

The contributions of Wenjin Chen:

- Collected Historical stock price dataset
- Collected tweet stock sentiment dataset
- Collected Reddit sentiment dataset
- Labeled 600 Reddit sentiment data
  (All datasets above were labeled in Google Drive and uploaded to GitHub by Tim)
- Responsible for training the sentiment model
- Responsible for visualizing the results

References

[1] S. Lindskog and J. A. Serur, "Reddit sentiment analysis," *Available at SSRN 3887779*, 2020.

[2] S. Raju and A. M. Tarif, "Real-time prediction of bitcoin price using machine learning techniques and public sentiment analysis," *arXiv preprint arXiv:2006.14473*, 2020.

11 Permissions

Team Member	Post Video?	Post Final Report?	Post Source Code?
Tim-Cheng Gu	Yes	Yes	Yes
Wenjin Chen	Yes	Yes	Yes