Does Social Media affect the stock market?

Ofir Nesher, Yuval Katz, Ofer Gross

Submitted as a final project report for the Natural Language Processing course, IDC Herzliya, 2021

1 Introduction

A meme stock is any stock that's seen excessive trading volume from retail investors who've targeted it on social media. In other words, this stock has "gone viral" on social media and has seen its price skyrocket as a result [1].

During the last few months, there's a hype around meme-stocks (like GameStop and AMC) in the stock market inspired by Reddit forums and Twitter [2]. Stocks are great for analysis because we can truly see whether the prediction is in fact a great indicator of the market movement [3].

1.1 Related Works

[1]There has been some previous work in similar areas to this project, largely using things like Twitter data. (Mittal and Goel, 2012) and (Nguyen and Shirai, 2015) are papers that look at the entirety of Twitter sentiment to predict market movements, focusing on the Dow Jones Industrial Average index. (Pagolu et al., 2016) focused on specific stocks and the Tweets being made that mention them, which reflects the approach in this paper of looking at specific stocks. (Li et al., 2014) looks at the sentiment of news to predict stock pricing, which is similar but does not rely on social media or internet message boards. There are also some non-academic efforts to solve this problem, like that of Redditor u/swaggymedia who created swaggystocks.com which is a realtime dashboard tracking various stock tickers along with their sentiment from recent posts on r/wallstreetbets. They use a bag of words sentiment analysis but did not publish more specifics. More generally, there are existing classifiers that have been trained on social media data that will be useful in the context of Reddit, which would more closely resemble other social media than traditional literature.

[1]https://github.com/KyleGrace/redditstonks/blob/main/RedditStonks.pdf

2 Solution

2.1 General approach

In this project, our goal is to analyze Twitter's tweets about a specific stock, more specifically its ticker (stock symbol). We want to see if the general sentiment about the company (by searching tweets with the ticker) reflects the trend of the stock's price (positive sentiments are followed by ascending price and negative sentiments followed by descending price).

In order to check the above, we scrape tweets with a relevant ticker hashtag (i.e. "#AMC") and compare 3 models for text sentiment analysis and discuss the results later in this paper:

1. **VADER Sentiment Analysis** (https://github.com/cjhutto/vaderSentiment)

   VADER (Valence Aware Dictionary and Sentiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media.

   About the Scoring: The sentiment property returns a an object of the following form:

   'pos': 0.303, 'compound': 0.3832, 'neu': 0.697, 'neg': 0.0 The compound score is computed by summing the valence scores of each word in the lexicon, adjusted according to the rules, and then normalized to be between -1 (most extreme negative) and +1 (most extreme positive). This is the most useful metric if you want a single unidimensional measure of sentiment for a given sentence. Calling it a 'normalized, weighted composite score' is accurate. NOTE: The compound score is the one most commonly used for sentiment analysis by most researchers, including the authors.

2. **TextBlob** (https://textblob.readthedocs.io/en/dev)

   TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

   The sentiment property returns a namedtuple of the form Sentiment(polarity, subjectivity). The polarity score is a float within the range [-1.0, 1.0], and this is the value we extract from each tweet. The subjectivity is a float within the range [0.0, 1.0] where 0.0 is very objective and 1.0 is very subjective.

3. **BERT** (https://arxiv.org/abs/1810.04805)

   BERT, stands for Bidirectional Encoder Representations from Transformers. BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for

a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

2.2 Design

The general steps of the project are as follows:

1. Reading tweets:
   Using the TWINT library: Twitter Intelligence Tool (https://github.com/twintproject/twint)

   Twint is an advanced Twitter scraping tool written in Python that allows for scraping Tweets from Twitter profiles without using Twitter's API.

2. Preprocess the data:
   Parsing and cleaning the data: We store all tweets as Pandas DataFrame. Then, we removed redundant columns (location, address, etc.). We also kept tweets in English only. Eventually, we mostly care about the content of the tweets, their date (we removed tweets from non-trading days - weekends).

   Models' final Input: The original data frame contains 61,430 tweets from 10 days.

   Models output: Each model (VADER/TextBlob/BERT) contributes a new column to the DataFrame with its sentiment score per tweet, in range from -1 to +1 (-1 being the most negative sentiment and +1 being the most positive).

   Parsing models output:

   - Group the DataFrame by date and get the average sentiment score per day, per model. Eventually, the final DataFrame contains date, average sentiment per model and the actual stock price (all grouped and sorted per date)

3. Model architecture:
   The only model modified is BERT: Configuring the BERT model and Fine-tuning We will use Adam as our optimizer, CategoricalCrossentropy as our loss function, and SparseCategoricalAccuracy as our accuracy metric. The other two models were not modified.

3 Experimental results

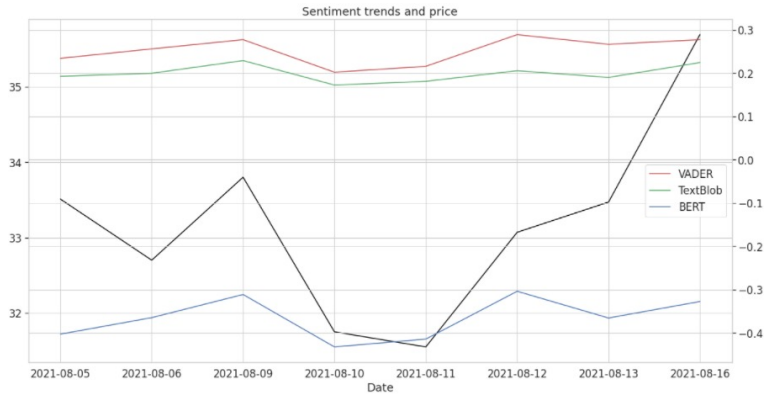

Figure 1: VADER, TextBlob, BERT Sentiment trends, stock price vs day

TextBlob:

		Predicted: Positive	Predicted: Negative
Actual	Positive	0	3
	Negative	0	5

Accuracy: 0.625
Recall: 1
Precision: 0.625

VADER:

		Predicted: Positive	Predicted: Negative
Actual	Positive	0	3
	Negative	0	5

Accuracy: 0.625
Recall: 1
Precision: 0.625

BERT:

		Predicted: Positive	Predicted: Negative
Actual	Positive	3	0
	Negative	5	0

Accuracy: 0.375
Recall: 0
Precision: 0

4 Discussion

(Provide some final words and summarize what you have found from running the experiments you described above. Provide some high level insights.
Note - your project will be evaluated for aspects, including the challenge you are facing with, the technique you selected, the rational of the experiments you decided to run, the insights you learned from this process and more. Remember, for the purpose of this course, the process is much more important than the accuracy of the algorithm in the end)

In figure 1 we see that all 3 models' sentiment measures averages are highly

("Spearman") correlated to one another (generally ascend/descend together) but are (unfortunately) not generally correlated with the actual price (in black).

The only difference is that VADER and TextBlob average scores are mostly positive (larger than 0) and BERT's are mostly negative (smaller than 0). This can explain the differences in the Confusion Matrices.

It is worth mentioning that we trained the BERT model on IMDB movie critics, and this can be the main cause for the different scores with the other models. We used this training data because this was the closest example we've found (critics on movies as an example of critics on stocks). We DID actually find a Twitter dataset but it was too large to train upon (15M entries).

The hypothesis of the stock price being highly influenced by social media does not hold vacuously.
We assume there are other variables which are also taking place. For example, the Stock's Index, employment rate, the company's earning, etc.

5 Further steps

- Use better training sets for the BERT model - tweets about stocks or finance topics.
- Scrape tweets for longer period of times (not only 10 days) - was hard to implement due to the slow runtime explained below.
- Improving runtimes - our code took quite a long time to run every time, which was problematic for development. We would prefer using Amazon's AWS services, the Spark library and better vectorized code.
- Learn and use the new language model GPT3 we are very curious about.

6 Visualizations

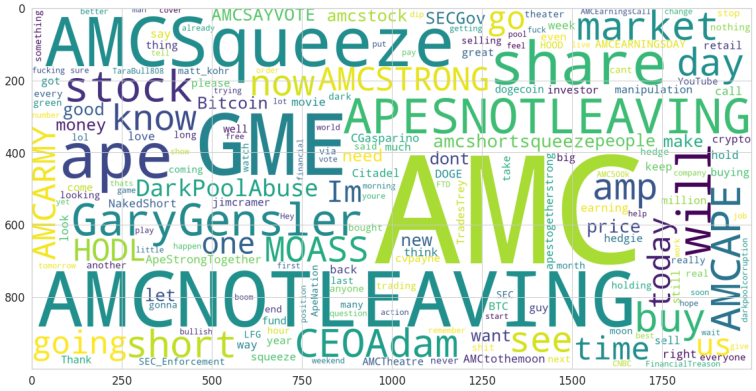

Figure 2: WordCloud portraying the most prevalent words in the whole dataset of 60K tweets

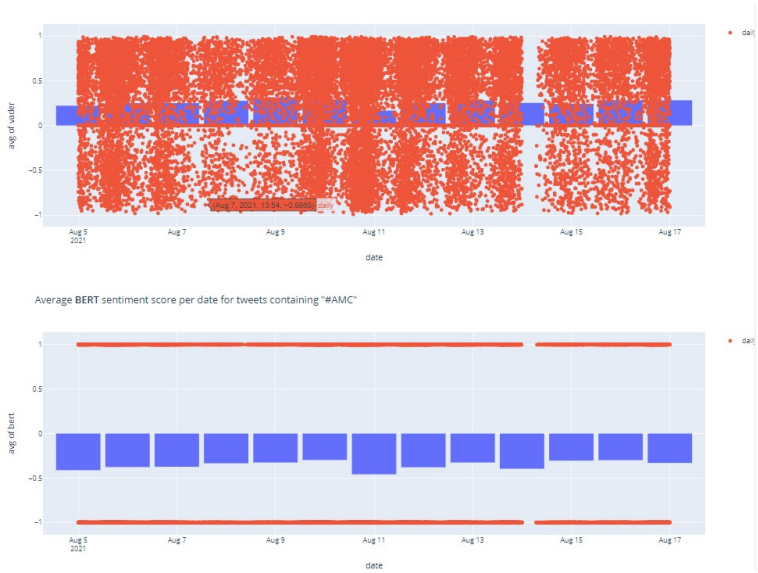

Figure 3: scatter plot of sentiments score for each tweet for VADER and BERT models vs date

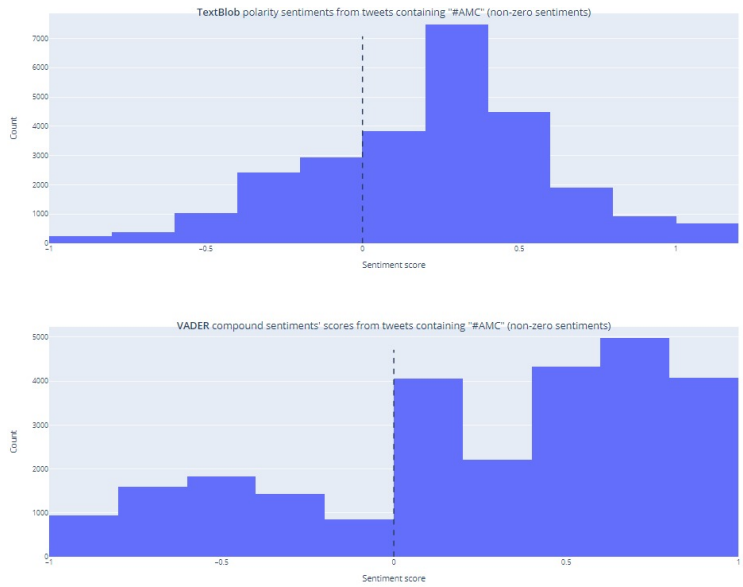

Figure 4: For these histograms, sentiment values (polarity/VADER) around zero [-EPSILON, EPSILON] have been removed, and a break has been added at zero, to better highlight the distribution of sentiment values. We can see that VADER's compound attribute separates the sentiments more clearly, and we can quickly deduce the general sentiment (positive/negative).

7 Code

Google Colab notebook:
https://colab.research.google.com/drive/14jdn4j4UD$_t xToNn3q3HUYrithGHTPRu?usp = sharing$

- More links (to some libraries we have used) are provided in the code.

References

- https://github.com/KyleGrace/redditstonks/blob/main/RedditStonks.pdf
- https://towardsdatascience.com/sentiment-analysis-in-10-minutes-with-bert-and-hugging-face-294e8a04b671
- https://github.com/twintproject/twint
- https://github.com/cjhutto/vaderSentiment
- https://textblob.readthedocs.io/en/dev
- https://arxiv.org/abs/1810.04805