Identifying Hyped Stocks on Reddit Using Natural Language Processing

Malene Hansen   Sebastian Harvej

May 2021

Abstract

Identifying hyped stocks on Reddit requires understanding and processing huge amounts of bodies of text. If it can be done successfully it can be compared to the stock market and ultimately be used to attempt to predict stock prices. By using Natural Language Processing it is possible to detect mentions of stocks and to analyse the context they are mentioned in. How to do just that is what we try to investigate in this article.

Contents

1 Introduction

Trading stocks have been around for decades and traditionally the price of a stock would go up, if the company did well financially and made profit following the principles of supply and demand.

But in recent times a new trend is appearing on the stock market, where the price is no longer connected to the economy or potential of a company but still follows supply and demand. This is best illustrated by the gamestop stock which skyrocketed from $ 20 to $ 347 in less then a month [8].

These kinds of stocks have been nicknamed ”meme stocks”. A meme stock is defined by a stock where the value of the stock primarily comes from hype on social medias like Facebook, Twitter, Reddit etc. [6]

In this paper we would like to use natural language processing to analyse these social media sites to see if it is possible to detect this hype.

1.1 Defining hype

Before attempting to identify a hyped stock it is necessary to reflect on the definition of ”hype”.

According to the Cambridge Dictionary the meaning of hype is:

> ”A situation in which something is advertised and discussed ...
> a lot in order to attract everyone’s interest ...” [1]

To identify a hyped stock it therefore seems reasonable to observe the quantity of mentions. And also to pay attention to the emotional tone connected to distinguish advertising contributions.

In this article we will explore how that can be done using natural language processing.

1.2 Natural Language Processing

Natural language processing is an umbrella term concerning the practice of making computers understand natural human language. There are a lot of language concepts that makes it difficult for computers to understand human language. Things like irony, tone and slang can alter the meaning of a text. That is why nlp libraries like spacy, which is what is used in this paper,is useful for understanding text. In this paper it is used for named entity recognition, lemmatization and stopwords.

2 Identifying Stock Mentions

Stocks are mentioned in different formats: Typically by the organization name e.g. GameStop or GameStop Corp. or by their ticker symbol, e.g. GME or $GME.

It becomes complicated as we discover that there exist ticker symbols that can be confused with commonly used words like for example ONE, GO or AI.

To capture stock mentions by searching through posts and comments, looking for a list of words, is therefore inadequate. The words must be understood in their context. Which is the core use case of Named Entity Recognition.

2.1 Named Entity Recognition (NER)

Named Entity Recognition (NER) is a form of Natural Language Processing (NPL). NER is the task of detecting named entities ("real-world" objects like people, places, organizations etc.) from a body of text and classifying them into a set of categories. [7]

NER relies on trigger words i.e. words that help understand the meaning of an entity. For example would "had lunch at" in the sentence "he had lunch at the new yorker" indicate that "the new yorker" is a restaurant, and not a citizen of New York. [**ner2**]

2.2 NER using python and spaCy

spaCy is an open-source library for Natural Language Processing in Python. spaCy offers various pre-trained pipeline packages ready to use. Here is a demonstration of how to use it:

Installation

```
pip install spacy

python -m spacy download en_core_web_sm
```

`en_core_web_sm` is an english model trained on blogs, newz and comments provided by spaCy. [2]

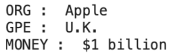
```python
# Import libraries
import spacy

# Load model
nlp = spacy.load('en_core_web_sm')
```

`nlp()` loads the trained NER pipeline that we have downloaded.

Extracting entities

```python
text = 'Apple is looking at buying U.K. startup for $1 billion'

doc = nlp(text)

# Print Label and Entity Name
for ent in doc.ents:
    print(ent.label_,': ', ent.text)
```

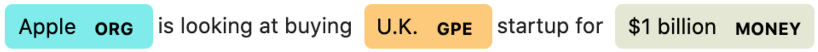
`nlp(text)` processes our text body and returns a spaCy Document Object. The Document Object holds the entities that spaCy has identified within the text and can be accessed by the property `.text` and their label by the property `.label_`. This is the output from our sample text:



```
ORG :  Apple
GPE :  U.K.
MONEY :  $1 billion
```

Visualizing entities

```python
from spacy import displacy
displacy.render(doc, style='ent')
```

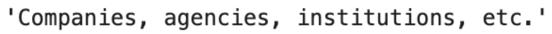


Apple ORG is looking at buying U.K. GPE startup for $1 billion MONEY

Discovering labels

The definition of the labels can be programmatically found with `spacy.explain()` :

```python
ORG = spacy.explain('ORG')
ORG
```



```
'Companies, agencies, institutions, etc.'
```

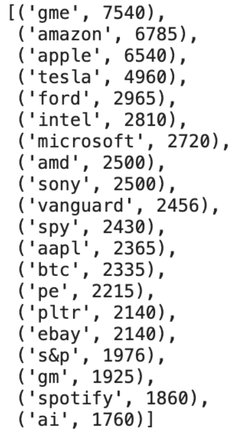
2.3 NER on subreddit dataset

On a test data set consisting of 145.000 posts and comments from the subreddit `https://www.reddit.com/r/investing` [3], we can loop through each submission to identify and extract entities with the ORG label:

```python
# Creating a list of ORG labeled entities
org_list = []
for text in data_set:
    doc = nlp(text)
    for entity in doc.ents:
        if entity.label_ == 'ORG':
        org_list.append(entity.text)
```

And to obtain an overview we count the discovered entities respectively and print out the top 20:

```python
# Print 20 most mentioned ORGs
from collections import Counter
org_freq = Counter(org_list)
org_freq.most_common(20)
```



```
[('gme', 7540),
 ('amazon', 6785),
 ('apple', 6540),
 ('tesla', 4960),
 ('ford', 2965),
 ('intel', 2810),
 ('microsoft', 2720),
 ('amd', 2500),
 ('sony', 2500),
 ('vanguard', 2456),
 ('spy', 2430),
 ('aapl', 2365),
 ('btc', 2335),
 ('pe', 2215),
 ('pltr', 2140),
 ('ebay', 2140),
 ('s&p', 1976),
 ('gm', 1925),
 ('spotify', 1860),
 ('ai', 1760)]
```

Entities like "vanguard" and "sp" do not represent single stocks. And entities like "Apple" and "aapl" refers to the same stock. But overall there is a solid basis for further analyse on classified submissions.

3 Sentiment analysis

Sentiment analysis is the practice of predicting a sentiment using machine leaning. In this case the most basic form is used with only two possible outcomes; positive or negative but sentiment analysis can also be used to detect neutral sentiment or even things like mood (sad, happy or angry) but it all depends on the data sets there are available and what makes sense in the context of the system that are being developed.

3.1 Data gathering

To train any machine leaning model data is needed, it can be data that one have collected or data found on the internet. In this example review data from IMDb, Amazon and Yelp is used. The data we use can be found here [4]. The data set consist of 2748 review texts with a sentiment value 0 for negative and 1 for positive. The ratio of positive and negative reviews is split evenly 50/50. In the table below you can see an example from each of the 3 data sets.

Review	sentiment
I advise EVERYONE DO NOT BE FOOLED!	0
The Songs Were The Best And The Muppets Were So Hilarious.	1
Not tasty and the texture was just nasty.	0

3.2 Data preparation

To use the data it needs to be prepared for training. The code displayed below is the code used here to clean the data. First the sentence is converted into a spacy doc, which is a sequence of tokens where the tokens are the individual words, then the collection is iterated over and the token is the converted into its base form, unless it is a pronoun which do not have any base form. A check is then made too know if the word is a punctuation or stop word. Stop words are a collection of some of the most commonly used words and they do not give much value.

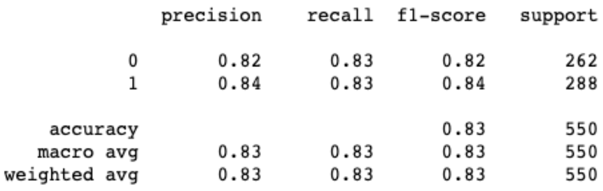
```python
def text_data_cleaning(sentence):
    doc = nlp(sentence)

    tokens = []
    for token in doc:
        if token.lemma_ != "-PRON-":
            temp = token.lemma_.lower().strip()
        else:
            temp = token.lower()
        tokens.append(temp)
    cleaned_tokens = []
    for token in tokens:
        if token not in stopwords and token not in punct:
            cleaned_tokens.append(token)
    return cleaned_tokens
```

3.3 Training the model

The data is then split into training and test data 80 % are being used for training and the remaining 20 % for testing. The model is then being fitted with the data, so it is able to predict the sentiment outcome.

3.4 Testing of the model

After the model have been trained with the test data, one could try to use the model to do predictions on the test data and compare the result of the model with the value given in the data. This can be performed by using a classification report. A classification report visualizes how many percent of the calculated values are right.



```
              precision    recall  f1-score   support

           0       0.82      0.83      0.82       262
           1       0.84      0.83      0.84       288

    accuracy                           0.83       550
   macro avg       0.83      0.83      0.83       550
weighted avg       0.83      0.83      0.83       550
```

Figure 1: classification report results.

As the figure shows, the results of the test was okay but not perfect but it can be hard to know when the precision is good enough.

So what numbers are acceptable and what numbers can realistically be achieved?

One would believe that humans would get a manual sentiment analysis right nearly every time but the score is often around 80-85 % so this is a good baseline to aim for [5]

The score of the test is around the baseline score but is it good enough?

To answer this question, one would have to look at what the purpose of the task you want to automate is and the critically of its purpose. In this case, it is considered that since the focus is not on the individual messages but rather on a large amounts of data. So in this case it is decided that it is okay.

3.5 Testing with reddit data

The model is then ready to be tested with the data gathered from reddit, to see if the model will still perform as acceptable when using it with its intended purpose.

It is being done by collecting a sample of the data, where one can perform its own analysis before it is analysed by the model. In this case a small sample are being used; 5 positive and 5 negative. This is a relatively small sample and one could argue that it is not sufficient but for this experiment it is decided that it is okay.

When comparing the result of the models result with the ones predetermined sentiment score, the result is not very good. The model only predicts right in 60 % of the cases which is only a small improvement compared to if it was just guessing randomly.

there can be multiple reasons why it has such a big error ratio. It could be because the test samples with reddit data is so small and then it looks worse than it actually is. Another reason could be that the model needs more training to be able to predict it correctly. In this case it is believed that the main issue is that the training data and the reddit data is not closely enough related to each other.

4 Conclusion

Nlp is very effective when working with text. We got impressive results detecting stock mentions using ner. We where also successive in training an sentiment analysis machine learning model even though our success was more limited here but the training pipeline is ready to train the model with new data to get better results.

References

[1] URL: https://dictionary.cambridge.org/dictionary/english/hype.

[2] URL: https://spacy.io/models.

[3] URL: https://github.com/PBASOFT/Data-Science-Project/tree/main/data/r_investing_sample.

[4] URL: https://github.com/laxmimerit/NLP-Tutorial-8---Sentiment-Classification-using-SpaCy-for-IMDB-and-Amazon-Review-Dataset/tree/master/datasets.

[5] Paul Barba. *Sentiment Accuracy: Explaining the Baseline and How to Test It*. URL: https://www.lexalytics.com/lexablog/sentiment-accuracy-baseline-testing. (accessed: 06.05.2021).

[6] Erin Gobler. *What Is a Meme Stock?* URL: https://www.thebalance.com/what-is-a-meme-stock-5118074. (accessed: 05.05.2021).

[7] Christopher Marshall. *What is named entity recognition (NER) and how can I use it?* URL: https://medium.com/mysuperai/what-is-named-entity-recognition-ner-and-how-can-i-use-it-2b68cf6f545d. (accessed: 04.05.2021).

[8] *yahoo finance*. URL: https://finance.yahoo.com/quote/GME/chart/. (accessed: 05.05.2021).