Predicting possible Stock Hypes using Sentiment Analysis of News Papers and Web Forums

A Project for the Course Python for Finance II at the University of Vienna

Authors: Atif Gangar, Jasmine Cerno, Loris Sultano

Introduction

Stock market prices and returns for certain sectors are nowadays quite volatile. This volatility is due in certain cases to extreme optimistic expectations by investors, which are itself often the product of a culture of believe in these sectors, fuelled by the media and investors themselves. The result are steep increases in stocks such as GamesStop, Tesla or some cryptocurrencies.

We propose, that the information contained in newspapers and web forums could possible be used to predict such steep increases in prices, which we will call hypes, for a short to medium time span, such as one month to three months. Using sentiment analysis of articles of the Guardian, a newspaper, and various reddit forums, we try to train a model, to predict such possible hypes.

Data

As a first step in our data gathering, we need to decide on some stocks, which will be used for the training of the model. Since we try to predict price movements via the discussion about stocks in forums, we will use the ten most mentioned stocks in some reddit forums. As forums we will use the subreddits r/wallstreetbets, r/StockMarket, r/stocks and r/investing, since these are the biggest and most relevant subreddits regarding investing in and trading stocks.

To identify the ten most mentioned stocks, we look at 1000 posts from the 4 forums, as well as their comments, since most of the discussion happens in the comments to a post.

```
NVDA: 2108 mentions
AMD: 1598 mentions
INTC: 679 mentions
MSFT: 657 mentions
TSLA: 461 mentions
AAPL: 366 mentions
AMZN: 336 mentions
META: 315 mentions
GOOGL: 144 mentions
AVGO: 137 mentions
```

After identifying the ten most mentioned stocks, we continue using just these ten stocks. We limit ourselves to only ten stocks, due to the computational capabilities of our machines. So that training the models will be possible in reasonable time and also so that we have enough time to troubleshoot possible error encounter.

As a time span for which we will get our data, we will just use everything we can get since 1.2.2012, since then r/wallstreetbets was founded.

Stocks and Variable indicating a Hype

To identify a possible hype that leads to a steep increase in a stocks price, we simply just define a hype as 10% or more increase price over a timespan of one month. To capture this we just create a binary variable indicating, whether this happened or not over 20 days, since we assume for simplicity, that a month has 20 trading days.

Guardian
To get relevant articles from the Guardian, we search all articles since the beginning of 2012 for the names of the companies of the stocks as well for the keywords *stocks* or *economy*, to capture also overall economic news and not just specific news to our chosen stocks in our sentiment analysis.

Reddit
We also do the same for the 4 subreddits, with the only change, that we also search for the tickers of the stocks, since stocks are often called only by their tickers in the discussions on reddit and not by their companies full names.

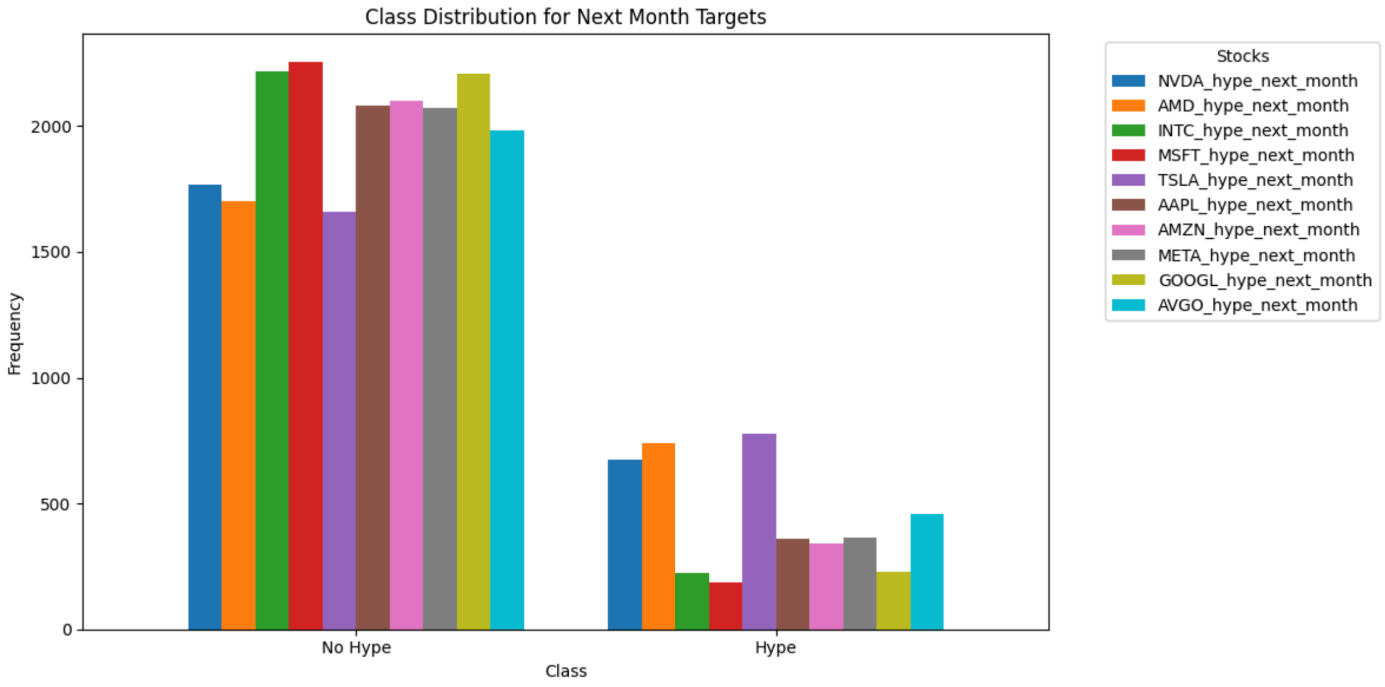
Finally we will combine both the articles from the Guardian and the post from reddit to one single data frame, so that we can create a bag of words model out of it. For this we will combine them according to the date, so we will drop possible information contained in the time of day at which a reddit post is posted. Because we are interesting in detecting hypes over a timeframe of roughly a month, we believe this information is not relevant, since at which time of the day a post is posted, could maybe indicate price movements on the next day, but not the next month.

Models

Bag of Words Model
To perform sentiment analysis using the collected data, we first need to fit and transform the text data. For this we use the *TfidVectorizer* , because it not only focuses on the frequency of words present in the corpus but also provides the importance of the words.

In addition to creating the bag of words model, we also have a quick look on distribution of our binary target variables, since one has to expect imbalances between these two targets, since hypes might not occur so often.

When visualising the target variable, it is immediately clear, that we deal with a great imbalance between the two states. Non hypes are much more frequent for all stocks we are looking at. This is something we have to consider and address in the specification of our models, as well at the interpretation of their performance.



Random Forest

Our first model we try to train is a simple multioutput random forest classifier. To address the imbalance we make sure to set the *classweight* attribute of the function to *balanced*.

Since sentiment analysis in finance involves dealing with noisy data where outliers and conflicting sentiments can occur frequently, using a random forest classifier can make sense, because it is an ensemble method that aggregates predictions from multiple decision trees, which helps in smoothing out noise and reducing the impact of outliers.

```
Overall Metrics:
 Dataset  Accuracy  Precision   Recall  F1 Score
Training  0.197131   0.626748 0.642031  0.495554
    Test  0.009820   0.161221 0.213130  0.116689

Per-Stock Metrics:
                Stock  Training Accuracy  Test Accuracy  Precision   Recall  F1 Score
 NVDA_hype_next_month           0.802459       0.423895   0.381779 0.724280  0.500000
  AMD_hype_next_month           0.508607       0.703764   0.833333 0.027027  0.052356
 INTC_hype_next_month           0.948361       0.841244   0.000000 0.000000  0.000000
 MSFT_hype_next_month           0.958197       0.891980   0.000000 0.000000  0.000000
 TSLA_hype_next_month           0.745492       0.271686   0.260000 0.993631  0.412153
 AAPL_hype_next_month           0.902869       0.561375   0.137097 0.386364  0.202381
 AMZN_hype_next_month           0.394262       0.823241   0.000000 0.000000  0.000000
 META_hype_next_month           0.400000       0.731588   0.000000 0.000000  0.000000
GOOGL_hype_next_month           0.941393       0.869067   0.000000 0.000000  0.000000
 AVGO_hype_next_month           0.428279       0.772504   0.000000 0.000000  0.000000
```

Although some test accuracies, for instance Google, Microsoft and AMD are quite promising, when looking at the precision metric, it does not look that good any more.
For most of the stocks the precision is 0, even if the test accuracy was quite high, like in the case of Google. This finding is most likely driven by the high imbalance, since the model predicts quite accurately, only because there are much lesser hypes than non hypes and it is not able to detect any hypes for Google. Looking at the graph above, we can see this also, since for Google, the imbalance is one of the greatest for the stocks.
An interesting case is AMD, since the test accuracy is quite good and precision too, the positive predictions are mostly correct, but recall is really low, indicating we miss many true positives.

Gradient Boosting

To further address the imbalance in the target variable, which most likely causes the really low precision, we try gradient boosting via *XGBoost*. Since *XGBoost* allows for the use of weighted loss functions, it could help to counter the imbalance. Here we use for each stock their imbalance ratio as a weight, which should help to train the model better.

```
Overall Model Metrics:
   Metric    Score
 Accuracy 0.248773
Precision 0.135536
   Recall 0.034389
 F1 Score 0.047248

Individual Stock Metrics:
                Stock  Test Accuracy  Precision   Recall  F1 Score
 NVDA_hype_next_month       0.540098   0.288889 0.106996  0.156156
  AMD_hype_next_month       0.679214   0.296296 0.043243  0.075472
 INTC_hype_next_month       0.836334   0.200000 0.010309  0.019608
 MSFT_hype_next_month       0.891980   0.000000 0.000000  0.000000
 TSLA_hype_next_month       0.644845   0.236842 0.171975  0.199262
 AAPL_hype_next_month       0.854337   0.333333 0.011364  0.021978
 AMZN_hype_next_month       0.823241   0.000000 0.000000  0.000000
 META_hype_next_month       0.731588   0.000000 0.000000  0.000000
GOOGL_hype_next_month       0.869067   0.000000 0.000000  0.000000
 AVGO_hype_next_month       0.772504   0.000000 0.000000  0.000000
```

This model has an overall better test accuracy, but similar low precision and recall scores, still indicating that the accuracy is only due to the high imbalance. It seems to just happen to guess correct since most targets are non hypes.

Balanced Radom Forest

As a last try we go back to a random forest model, but this time we bootstrap each sample by undersampling the majority class. By doing this, we try to create a balanced dataset for each tree in the forest, hoping the imbalance is addressed effectively.

```
Overall Metrics:
 Dataset  Accuracy  Precision   Recall  F1 Score
Training  0.005328   0.277125 0.650829  0.343713
    Test  0.000000   0.224638 0.650772  0.268434

Per-Stock Metrics:
                Stock  Training Accuracy  Test Accuracy  Precision   Recall  F1 Score
 NVDA_hype_next_month           0.725410       0.418985   0.399281 0.913580  0.555695
  AMD_hype_next_month           0.468852       0.510638   0.284091 0.405405  0.334076
 INTC_hype_next_month           0.720082       0.173486   0.158863 0.979381  0.273381
 MSFT_hype_next_month           0.740984       0.134206   0.102916 0.909091  0.184900
 TSLA_hype_next_month           0.721311       0.296236   0.264249 0.974522  0.415761
 AAPL_hype_next_month           0.735246       0.155483   0.145695 1.000000  0.254335
 AMZN_hype_next_month           0.347951       0.711948   0.179245 0.175926  0.177570
 META_hype_next_month           0.382377       0.708674   0.347826 0.097561  0.152381
GOOGL_hype_next_month           0.718443       0.158756   0.133446 0.987500  0.235119
 AVGO_hype_next_month           0.420492       0.738134   0.230769 0.064748  0.101124
```

Using this model, the test accuracies are now quite lower overall while the precision and recall is much higher overall, indicating, that the model is overall more correct when identifying a hype and also does not miss that many hypes as the previous models. Bootstrapping the samples seems to have worked to better the problem of imbalance. But the F1 score is still quite low.

Conclusion

In our analysis of multi-output classification models for predicting stock hype via sentiment analysis, we tested several approaches to tackle the challenges of class imbalance and noisy financial data. Our initial model, a random forest classifier, utilized ensemble averaging to manage noise. However, despite applying balanced class weights, the model struggled with class imbalance. While some stocks, like Google and Microsoft, showed high test accuracies, the precision was near zero, indicating that the model often predicted the dominant non-hype class without accurately identifying actual hype events.

To address this, we implemented gradient boosting with *XGBoost*, leveraging weighted loss functions to counter imbalance. Although test accuracy improved, the precision and recall remained low, suggesting that the model still missed many true hype events.
Our final approach was a balanced random forest, which undersampled the majority class to create a more balanced dataset. This improved precision and recall, indicating better hype detection. However, test accuracy decreased, and while F1 scores improved, further refinement is needed to balance precision and recall effectively.

Therefore we conclude that, given our data and modelling, we were not able to predict hypes of stocks using sentiment analysis.