Sentiment Analysis with News Headlines

Xinhui Gu, Tony Peng, Patrick Sun, Lue Xiong

Abstract
In this paper we aim to demonstrate the predictability of future stock movements using sentiment analysis, which specifically takes the form of news headlines. The way in which we will achieve this is through the employment of various machine learning techniques: starting from data collection, data cleaning, then using the data to generate various features that capture the signals in the market, and finally deploy our machine learning model using this carefully tuned dataset.

Keywords
Headlines — Machine Learning — Data Cleaning — Feature Engineering — Hyper-parameter

Contents

1. Introduction

Nowadays capital market in the U.S. is becoming increasingly efficient, as a result, the outcome of traditional alpha seeking methodologies which uses company fundamentals and economic characteristics are deteriorating rapidly. For example, when new information becomes available to the market, modern technology advancement enables it to spread at a rate such that when an investor receives the information, it is highly likely that the signal/value of this piece of information has already diminished. Therefore instead of looking for market signals using the traditional set of economic variables, many novel approaches has been developed to predict future equity market movement, and sentiment analysis is one of the most popular one of such category.

Trying to capture market sentiment is a good way of predicting future stock price movements because the price that we see in the market, is very much driven by the general majority opinion on the company, and such public opinion comes from mostly market news. Therefore using market news as a pool from which to extract signals is logically sound and can lead to trading strategies which potentially front-run the general public by anticipating the public sentiment on the stock.

Our approach starts with this idea and focuses on one particular company, which is Microsoft. Throughout the project we limited ourselves to this one company instead of trying to predict the entire market direction. This is because the broad marker is influenced by large and small companies, and for the smaller companies there is a lack of relevant news headlines. Hence if we were to consider the entire market, we would eventually end up using only sentiments of a few large companies to represent the entire equity market. This may introduce bias into the model by neglecting the aggregate medium to small cap companies contributions. Never the less our model is applicable to any single company in the market and to many investors this is still of great value.

2. Data Collection

At first, we tried to collect MSFT's market news data by scrapping from Google News headlines. We employed Beautifulsoup packages in Python to do the task. However, the scrapping was terminated by technical issues as we proceeded to get more data in longer time range. We also tried to scrape from other news search engine such as Bing, but it didn't end up with successful results.

Because of the difficulty in scrapping data, we decided to look for APIs to directly download the data. Bing, Financial Times, WSJ are some of the choices available, but time range of their news that we can select is limited.

Eventually we use the GlobalNews API service from Xignite, a market data provider for data collection. Even though its service only provides news headlines up to 2014-3-27, their data are highly organized by date and we obtained a total of 33,808 news spanning over 1265 days, which are enough for training and testing.

3. Data Cleaning and Feature Engineering

To analyze news sentiment and extract key features out of news text, we followed the following procedure:

1. Delete all the text data in the weekend in order to match news sentiment to trading days
2. Separate headlines into words for each day
3. Use the 2014 Master Dictionary from Prof.Bill McDonald's web page. The dictionary is an extension of the 12dicts word list, which are widely used for textual analysis in financial applications. The dictionary contains seven sentiment category identifiers that we are going to use in our model building. They are summarized below:

Selected Features:
Negative
Positive
Uncertainty
Litigious
Constraining
Superfluous
Interesting
Modal

4. Generate seven category features by mapping each day's words to the dictionary. Count the number of words that fall into each of the category each day.
5. Construct our momentum feature by calculating money flow index(MFI).

$$MFI_t = 100 - (\frac{100}{1 + MoneyFlowRatio_t}) \quad (1)$$

where

$$\text{Money Flow Ratio} = \frac{\$length - period positive flow_t}{\$length - period negative flow_t}$$

Money Flow =

$$volume * \frac{P_t^{high} + P_t^{low} + P_t^{close}}{3}$$

Money Flow is positive if the following condition holds:

$$\frac{P_t^{high} + P_t^{low} + P_t^{close}}{3} > \frac{P_{t-1}^{high} + P_{t-1}^{low} + P_{t-1}^{close}}{3}$$

MFI takes volume into account so that we can see volume's impact on the final results.

6. Incorporate RSI-relative strength index

$$RSI_t = 100 - (\frac{100}{1 + RS_t}) \quad (2)$$

where

$$RS = \frac{averageof\$length - periodupcloses}{averageof\$length - perioddowncloses}$$

RSI is a good indicator for whether the stock is in oversought or overbought condition.

Equipped with these features, we proceed to train and test the machine learning model to predict the stock movement each trading day.

4. Weekly Progress Milestones Accomplished

4.1 Phase 1: Start of Project

We started our project by analyzing the right metric for predicting the stock return movement. As mentioned earlier, we decided to explore the signals in the news headlines of Microsoft. We used the finance dictionary to get the daily number of the difference of positive and negative words and use this as the signal to predict the Microsoft stock movement. The prediction accuracy is 51% which is very close to flipping coins.

4.2 Phase 2: Improvement of Models

We modify and enhanced our model by adding more labels. Because the prediction was poor if we only use 2 labels. We also added more features to classify the headlines. We used the seven features as mentioned earlier in the feature engineering part. The accuracy better than flipping coins. Since we have three prediction classes, the model prediction of 38.5% is better than 33.3%.

4.3 Phase 3: Tuning the Model and Modification of Metric

We reformulated the model and the metric. Because we believe that when we have many good news, market may over respond to the news or there may exists latency in the market to respond to the news. To accommodate for this time lagging effect, we decided to instead of predicting next day's movement, we tried to predict the movements over the next time period of X number of days. The accuracy is improved greatly and the explicit description of the model and metric are as below.

5. Metric for Measuring Performance

Our prime metric of performance measure is test set accuracy, where among all the dates in test set, accuracy is defined as the proportion of days where we correctly predicted the sign of return on that day.

We would compare our model performance on the test set with a benchmark model and a naive model as detailed in section 5.

6. Latest Model

Using daily return data from 3/27/2014 to 9/12/2017 and a training set, validation and test set split of 70%, 20% and 10%, we compare the performance of 3 models. We aim to predict 10 days return, due to the fact that on a 10-day basis, return is negligible, we can assume a mean of 0 and have a band around the mean to make sure that the positive and negative labels are significant enough. With this band, we created three labels as follows:

- If the return on a particular 10-day period is greater than 0.3 times the standard deviation of all 10-day returns, or in other words 1.3%, we label that day as positive.
- If the return on a particular 10-day period is smaller than 0.3 times the standard deviation of all 10-day returns, we label that day as negative.
- else, we label that day as neutral.

6.1 Naïve Model

For the naïve model, we tried the simplest approach possible. In the test set, we calculated the percentage of positive labels to be 68%, negative labels to be 29% and the rest being neutral. Therefore, our naive model is that we would randomly predict an up move on each day with probability 68% and 29% for a down move on our test set. Then we would compare our prediction with the actual positive or negative returns on that day according to the label rule described before. The result is that our naive model had a prediction accuracy of 48% in the test set.

6.2 Baseline Model

For the baseline model, we used a slightly more sophisticated approach than just a Trinomial distribution. Every day we count the number of positive words and negative words in all the relevant headlines and subtract them. Since on average there should be equal number of positive and negative words for the company on a day, we can again assume this difference to have a mean of 0, then out baseline model's prediction rule is as follows:

- If the difference on a particular day is greater than 0.3 times one standard deviation above 0, we label that day as positive
- If the difference on a particular day is smaller than 0.3 times one standard deviation below 0, we label that day as negative
- If the difference on a particular day is between 0.3 and -0.3 times the standard deviation around 0, we label that day as neutral.

The result that we found from the baseline model is presented below:

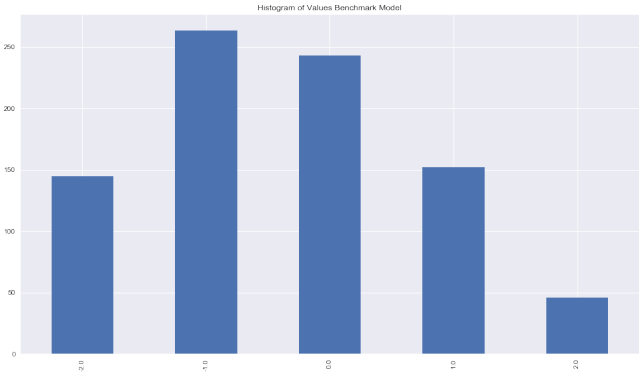

Figure 1. Baseline Model Error Result

Figure 6 is a histogram of the difference between baseline model prediction and actually label, both labels takes values in {-1,0,1} hence the difference takes values in {-2,-1,0,1,2}. As we can see, using the baseline accuracy is 28.62% which is the proportion of 0s out of all data points.

6.3 State-of-the-Art Model

Here we used a multi-class support vector machine method to classify the returns as three types: positive, neutral and negative as labeled before, which is our y variable. Our independent variables came from a master dictionary from online source (see reference), where a list of around 90,000 words are divided into the following categories: Negative, Positive, Uncertainty, Litigious, Constraining, Superfluous, Interesting and Modal. The features that we use are the number of times that words of that category appeared on a particular day's news headlines. With a few added features such as momentum. The result on the test set is an accuracy of 60.7%, however all predictions are 1s.

We think there are a few factors that may have contributed to the poor performance of the model:

- When we labeled returns each day to be positive, negative or neutral, the band was too narrow. In other words, there are days where the return is labeled as positive, however it is only very slightly positive, and hence not

significant enough. Therefore when we run our sentiment analysis, we see that there is actually no strong evidence of a positive sentiment, however due to the narrow neutral band, we ended up labelling the data point as positive. This is a very important factor as sentiment is very sensitive to the change of such parameters, because in the news headlines, there tends to be two types of extremes: days where there's not much news about the company and days where there is a lot of news about it, due to a positive or negative event. If we the band around neutral is not wide enough to filter out the days where there is not much news, then the result will be very noisy.

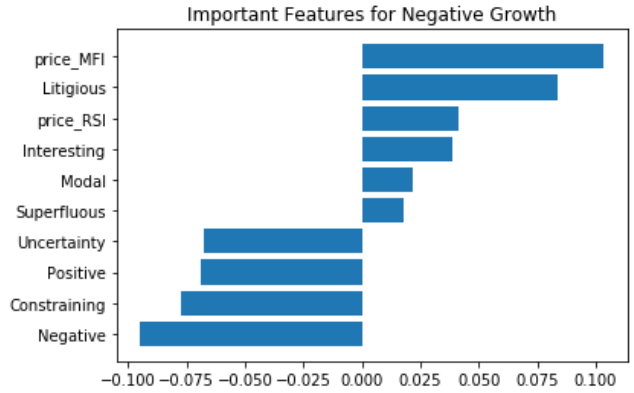

Figure 2. Feature Importance for Negative Growth

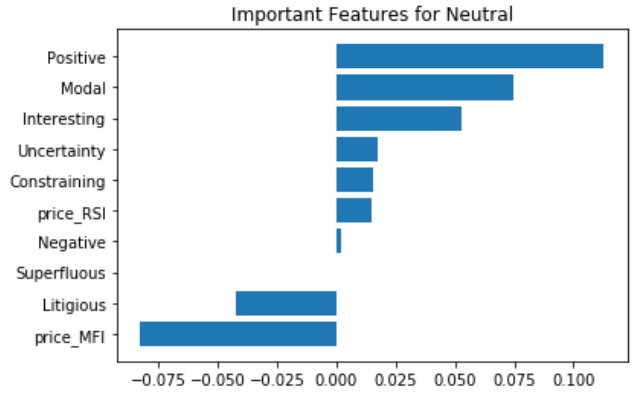

Figure 3. Feature Importance for Neutral

6.4 Improvements on State-of-the-Art Model

For the improved model, we try to address the problems identified above in the following way:

To solve the problem we increased the band around neutral returns, specifically we used a 1.15 standard deviation, which is a 5% return as a threshold, where if the 10-day return is above 5%, we label the that day as positive, and if it's below -5%, it is labeled as negative. Anything in between is neutral. For this, we achieved a test set accuracy of 84.5% however all prediction are 0s.

In essence, the problem is, the larger the threshold, more labels are 0s and the smaller the threshold, more labels are 1s. Therefore we have the ends of two extreme. This is a big problem and we will address this in the next section. For now we analyze the importance of the feature as follows:

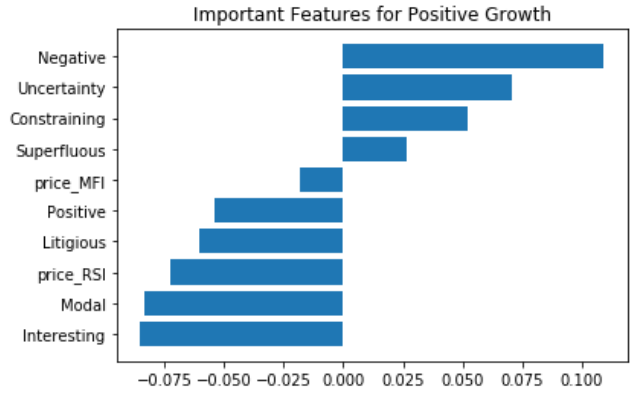

Figure 4. Feature Importance for Positive Growth

The larger the absolute value of a coefficient, the more significant a feature is. Here we can observe that negative sentiment actually provided a strong negative signal for negative growth and a strong positive signal for positive growth. This suggests that market tends to over-react to negative news immediately when the news published and then in the

following 10-day period the price would gradually revert back to its true value.

7. Biggest Problem and Reconciliation

We can clearly see that the model accuracy relies greatly on how we label the data, which in turn is determined by the threshold of the neutral band we choose. If we use a very high threshold for the neutral band, more label are 0s and all our predictions will be 0s. On the contrary, if we use a very low threshold for the neutral band, most labels are 1s and hence all predictions will be 1s. Both achieving very high accuracy of around 60% to 80%. However the model becomes trivial because it only predict 0s or 1s. This is a very interesting phenomenon, ideally we hope in the middle somewhere there is a peak of accuracy even though it's not as high as accuracy at either end. Hence we ran a simulation where we treat the threshold as a hyper-parameter by trying a range of threshold values, and plot the accuracy against the fraction of standard deviations we use as a threshold, and found the following result:

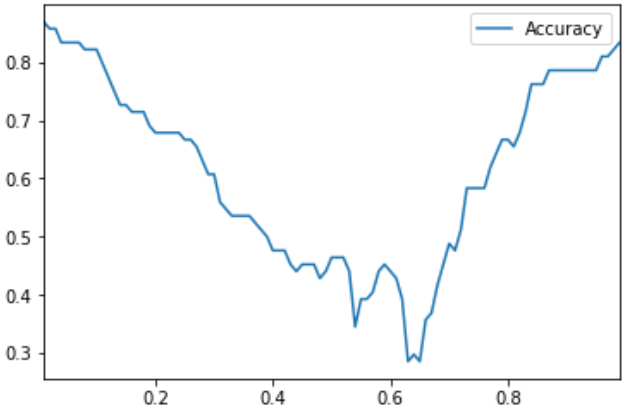

Figure 5. Accuracy against threshold value

From this graph, we see a peak of accuracy at around 0.5 times the standard deviation and a accuracy of around 53.6%. We also calculated the F1 score of accuracy shown below, which takes imbalanced data into account. We see the same problem as the accuracy score. The core of the problem is to find a optimal parameter value that gives a fairly balanced dataset, and the results of all three models are summarized below:

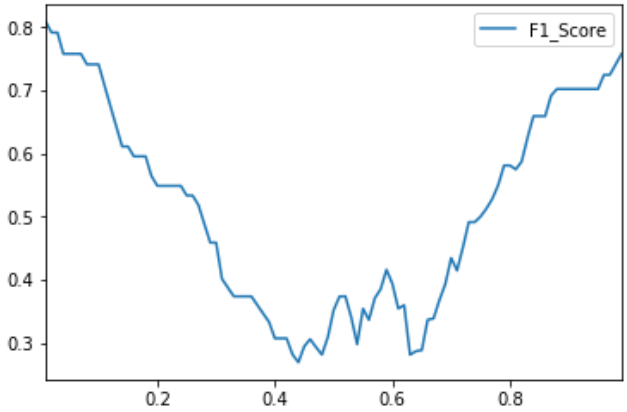

Figure 6. F1 score against threshold value

Accuracy Comparison:	
Model	Accuracy
Naïve	48%
Baseline	29%
State-of-the-Art	53.6%

8. Potential Applications and Limitations

As mentioned before, our model focused on predicting the performance of one stock in the future. However this is the building block for many other applications:

8.1 Long Short Trading Strategy

Since we can predict the movement of one stock, this easily generalized to a portfolio of stocks. For example, if we have a portfolio of various stocks each with a portfolio weight, then we can perform this stock by stock analysis on each one of them, and for the portfolio, we long the ones that we predicted to be positive over the next period and short the ones with negative prediction.

However this does not tell us how much we should long or short each one, a modification that can fix this problem is to use a machine learning algorithm

that instead of outputting a categorical result, a ordinal number, for example ranging from 0 to 1. Then we can determine the amount of short and long proportional to the size of the outcome. In addition, this can help us to limit the trading cost, because if the output value is very close to neutral values for a stock, we do not need to change our position for that particular stock due to the trading cost that it will incur. Hence we can set a threshold that gives us an inaction band.

8.2 Index Prediction

If instead of a single stock, we would like to determine predict the movement of a index, for example S&P 500, this can also be done with our model. First we would like to identify the top stocks that accounts for X% of the capitalization of the index, then we would run our model on these large cap stocks. The reason being if we run the model on every single stock in the index, it is likely that the small cap stocks don't have much news headlines that are related to them every day, resulting us picking up a lot of noise from these small stocks which undermines the prediction accuracy.

8.3 Generalization to Beyond Equity Market

The basis of our analysis is that we try to take something in the news, that relates to some asset, and predict the change in value of that asset, hence by no means is this limited to stock returns. We can easily imagine a situation where if we want to predict the price of some corporate bonds, we can use the sentiment within headlines related to that company. However there is a problem which is that stock market is a lot more liquid than fixed income markets for example, therefore we cannot use 5-day re-balancing horizons and need to expand our time period of prediction at the cost of reducing the amount of data available for training.

9. Conclusion

Through this project, we have built a model using machine learning techniques that predict the return from news headlines and other factors. One of the very interesting phenomenon that we discovered during the project is that the convexity of the accuracy function resembles a convex function. This problem makes it difficult for us to tune the hyper-parameters even if there exists a optimal hyper-parameters set. Also the models predictive power are greatly dependent on the selection of the metric.

10. Acknowledgments

We want to thank the professors for their insightful thoughts on the latest machine learning techniques. Also our gratitude goes to the helpful and knowledgeable GSIs which solved our problems promptly. Finally we want to thank everybody who has helped us along the way, we could never accomplish the project with their help.

References

Master dictionary from Bill McDonald's Word List Page: https://www3.nd.edu/ mcdonald/Word_Lists.html