Can Twitter Predict the Bitcoin Market?

Jonathan Bhimani-Burrows
jonathan.bhimani-burrows@mail.mcgill.ca

Victor Elmir Verchkovski
elmir.verchkovski@mail.mcgill.ca

Abstract

Much work has been done in the finance industry to develop algorithms that remove the impulsive, human side of trading and make the process more automated. Behavioral economics tells us that emotional sentiment can significantly affect an individual's behavior and decision-making, however the average day-trader typically trades manually. We therefore hypothesize that social media may correlate or potentially have an effect on the outcome of the Bitcoin market. We compared the performance of various methods of determining market sentiment with and without semantic information, in a way that would allow us to determine if such knowledge helped in the prediction of the crypto-currency market. We found that memory and computation constraints prevented an exhaustive and long term analysis. Given this, our results are inconclusive.

1 Introduction

Twitter is one the most popular social media sites in the world, with over 300 million users. With the ability to tweet any message in under 240 characters, it is a source for an incredible amount of unique text for natural language processing algorithms to extract valuable information. We will examine several methods for aggregating, filtering and analyzing Twitter data to isolate trends in the market, specifically in the BTCUSDT market on Binance, an international, multi-language crypto-currency exchange. There is little chance to create a full trading system based off Twitter data alone, but we will investigate the possibility of soft indicator, which would be capable of confirming other trade signals, or throw a warning if there is negative sentiment.

1.1 Related Works

Using text to predict market sentiment is a controversial topic in the academic community. The famous (Johan Bollen, 2010) study claimed to have 87.5% accuracy using Twitter data, but further examination and peer review made it clear that the time frame was too short, and the success criterion too vague. Other studies such as (Beckmann, 2017) have found more meaningful results by examining news headlines for market surges over the course of over a year. Our approach builds on the methods already common and tests new ways to process the textual data and train a model to make accurate predictions.

2 Feature Design and Selection

2.1 Data Collection

Twitter

Twitter provides an API for basic functionality. Unfortunately, searching archived tweets older than a week is not so simple, and the cost for accessing a meaningful amount of past data was too expensive. Furthermore, searching past tweets is not exhaustive, so there is no guarantee of consistent sampling over search timeframes. With the help of the Tweepy wrapper, a stream was set up to collect all new tweets that matched the query 'bitcoin' from Dec 1st until Dec 11th.

Binance

Binance also has an API, which made it easy to download all the candlesticks that matched our Twitter data period. Binance had no issues accessing past data, and hypothetically could be used to access any past market state to a 1 minute precision. In order to get as many data points as possible, it made sense to use the same 1 minute precision.

2.2 Preprocessing

Twitter text can typically contain quite a bit of noise, such as tagged users, hyperlinks, emojis, and illegible slang which had to be removed. Hashtags were kept, however the hashtag character was removed. As an international platform, there was a vast amount of non-English tweets to be filtered. Another fact to consider is not all tweets are viewed by the same number of users, for this we also recorded the number of followers the original poster has, as well as if they are verified or not. The final preprocessing step was choosing how to group the data, which was narrowed down to two methods.

2.2.1 Hive Method

All tweets over the course of 1 minute candlestick were collected into one text vector, which was then filtered. Overlapping tweets would be concatenated and duplicate words were removed (i.e. binary bag-of-words).

2.2.2 Direct Method

Similar to the Hive method, tweets were vectorized. However, the Direct Method takes advantage of the time difference between the tweets, in that each tweet is considered as it's own data point. Text is filtered, and non-english tweets are ignored. Due to memory constraints, baseline classifiers would do partial fits on 10 000 datapoints at a time.

Vectorizing

The final preprocessing decision came down to the vectorizer. CountVectorizer was used to vectorize the tweets for the Baseline and the VADER algorithms, but when it came time to optimize the Recurrent Neural Networks (for the Direct Method), the training process was too slow. Therefore, for these methods only, the built-in Keras Tokenizer was used, along with zero padding (for max tweet length). Doing this, training speed was increased by nearly a factor of 100.

Additional Features

For both the Hive and Direct method, two additional features were added: the number of followers a user has and whether or not they are verified. In the Hive method it takes the total, including non-English tweets. In the Direct method, only English tweets are considered. There are two main advantages/disadvantages at play for the two methods. The Direct method isolates sentiment for each tweet, but loses the follower count and verified count from non-English tweets. The Hive method takes advantage of non-English tweets but flattens the sentiment across a broader range of tweets.

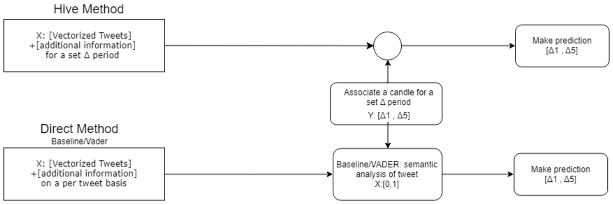
2.3 Labeling

For both Hive and Direct method, there were 2 possible labels: the change in price within the next 1 minute and 5 minutes. If the price change is positive, the label is 1, if the price change is negative, the label is 0. Volume was recorded but not used.

3 Methodology

The methodology of our project can be visualized the flowchart below. The principle of the baseline and the deep learning methodologies are the same, but there are a few minor differences.

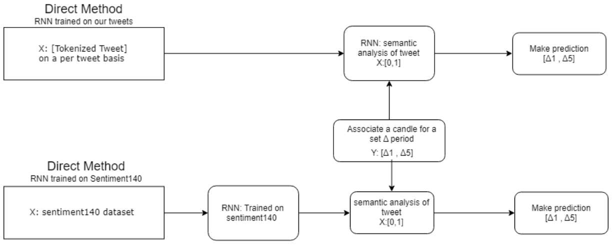
Figure 1: Baseline Methodology


Hive Method
X: [Vectorized Tweets] +[additional information] for a set Δ period
Make prediction [Δ1 , Δ5]
Associate a candle for a set Δ period
Y: [Δ1 , Δ5]
Direct Method
Baseline/Vader
X: [Vectorized Tweets] +[additional information] on a per tweet basis
Baseline/VADER: semantic analysis of tweet X:[0,1]
Make prediction [Δ1 , Δ5]

Baseline classifiers were trained on both the Hive and Direct method, without any additional semantic information, to see if we can predict short term directional movements in the market.

The deep learning methodology is similar to the baseline methodology, except the sentiment information was derived using Recurrent Neural Networks (more on this later). Once this was done, the sentiment information was added as an additional feature to the baseline classifiers, which would then

Figure 2: Deep Learning Methodology


Direct Method
RNN trained on our tweets
X: [Tokenized Tweet] on a per tweet basis
RNN: semantic analysis of tweet X:[0,1]
Make prediction [Δ1 , Δ5]
Associate a candle for a set Δ period Y: [Δ1 , Δ5]
Direct Method
RNN trained on Sentiment140
X: sentiment140 dataset
RNN: Trained on sentiment140
semantic analysis of tweet X:[0,1]
Make prediction [Δ1 , Δ5]

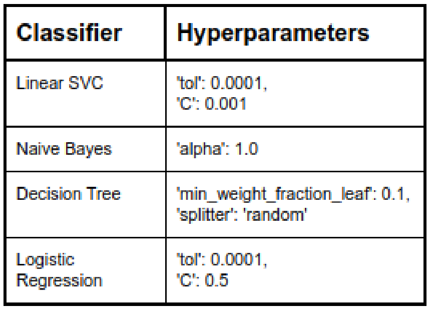
make a prediction on the 1 minute and 5 minute candles. It is important to note that the Naive Bayes and VADER algorithms also generated sentiment information that was used with our baseline classifiers, in the same way as with the RNNs.

4 Algorithms

4.1 Baseline Classifiers

LinearSVM, Naive Bayes, Decision Tree Classifier, and Logistic Regression Classifier are simple machine learning classifiers, commonly used for sentiment analysis and classification. They were trained on both Hive and Direct data, with hyperparameters tuned using GridSearchCV to predict short term directional changes. Due to time constraints, a fully exhaustive tuning was not performed, and for consistency, the same hyperparameter values were used for all final tests.

Figure 3: Fine-tuned Hyperparameters Using GridSearchCV



Classifier	Hyperparameters
Linear SVC	'tol': 0.0001, 'C': 0.001
Naive Bayes	'alpha': 1.0
Decision Tree	'min_weight_fraction_leaf': 0.1, 'splitter': 'random'
Logistic Regression	'tol': 0.0001, 'C': 0.5

4.2 Semantic Analysis

In order to test our hypothesis, it was necessary to run a semantic analysis on our collected tweets, in order to obtain additional information that might be helpful for a prediction. To accomplish this, several different methods were attempted. For all methods, the goal was to obtain an additional feature, called the sentiment vector, which was a binary 0 or 1, for negative or positive. This vector would be added to the information in the Direct method, which should give our prediction algorithms additional information on which to make a decision.

4.3 Naive Bayes

The first model we used was the Naive Bayes algorithm, which is known to perform well on natural language tasks, as the algorithm is generative and stores probabilistic information of the data. For the semantic portion of this project, it was considered a pseudo-baseline.

4.4 VADER: Valence Aware Dictionary and sEntiment Reasoner

The second algorithm tested was a pre-trained one called VADER (C.J. Hutto, 2014). VADER is a rule based model, designed specifically to deal with the inherent nature of social media tweets. The system uses a rule based approach, which outperformed many state-of-the-art systems, as well as human raters, which means it is ideal for our purpose.

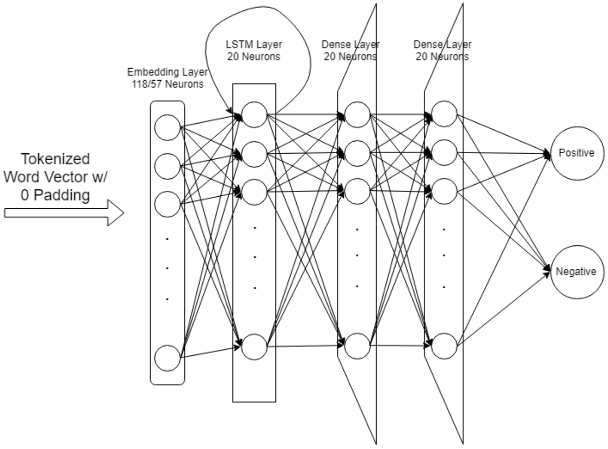
4.5 Recurrent Neural Network: Theoretical Basis

Recurrent Neural Networks, or RNNs, are a form of neural network, commonly used in Natural Language Processing tasks, that utilize the sequential nature of the data to predict the most likely next outcome. In our case, the sequential nature of the tweets gave us the impression that a properly modelled RNN would be a good candidate for sentiment analysis. As such, we chose to optimize RNNs in 2 ways, in an attempt to enrich our feature space and better predict our output.

4.6 Recurrent Neural Network: Trained on an Existing Dataset

The third algorithm tested for semantic analysis was a basic Recurrent Neural Network (RNN). RNN's are known to perform well on semantic disambiguation, and can take advantage of the sequential nature of data to help make a prediction. This RNN was trained on the Sentiment140 Dataset (Hult, 2013), which is a collection of 1.6 million pre-labelled tweets. Once the training and hyper-parameter tuning was done, this model would be used to classify our unseen tweets, and then this additional information would be passed into our prediction algorithms. The architecture of this, as well as the next model, is seen below.

Figure 4: Recurrent Neural Networks' Architecture



4.7 Recurrent Neural Network: Trained on our Tweets

The fourth and final algorithm tested was another RNN, but this time it was trained and tested on our own accumulated tweets with associated candles, as discussed in section 2.3 . Finally, for testing, our test tweets were passed in unlabelled and the sentiment was generated as the additional feature for our baseline classifiers.

5 Results

The results section is sub-divided for clarity: the semantic models and their performance will be detailed individually, then the Direct and Hive tweets on the baseline classifiers will be handled separately.

5.1 Naive Bayes

The Naive Bayes model performed remarkably well when compared to the other algorithms. The hyperparameters were optimized exhaustively and then were trained and tested on theSentiment140 dataset. At best, the validation error was 79 percent. Once this was achieved, the Naive Bayes was used to predict the sentiment of the unseen tweets.

5.2 VADER

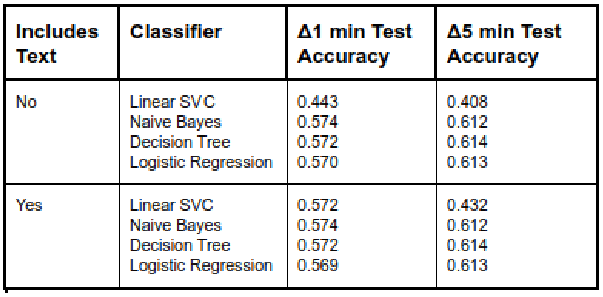
As VADER is a software designed for tweet sentiment analysis, we had the expectation that this would perform very accurately on sentiment analysis. For this method, first, VADER makes a combined prediction on the tweet with real valued outputs, indicating a positive, negative or neutral sentiment of the tweet, as well as the certainty that the tweet has this sentiment. For our purpose, we had to convert these real values into positive or negative classes (sentiment certainty was removed) with hyper-parameters using a grid search algorithm. As there was no training per se, the entire dataset was divided into a validation set, for the hyper-parameter tuning, and a testing set, and which gave a final validation accuracy of 66 percent. While this is a bit disappointing, the sentiment vectors were still used with our baseline classifiers for comparison purposes.

5.3 Recurrent Neural Networks: Trained on an Existing Dataset/Trained on our Tweets

For this discussion, we will talk about both RNNs together, as both had similar deficiencies and the end result was nearly the same. The goal of the RNNs was to see if we could train them on either a labelled dataset, or with our own tweets, in order to then apply this to our unseen data in order to make a sentiment prediction. Unfortunately, neither were able to achieve an accuracy above 50 percent. There are a few possible reasons for this. First, there are significantly more non-trainable parameters vs trainable ones, as the keras tokenizer with zero padding drastically increases the input dimensions. We attempted to fix this by increasing the number of dense and LSTM layers, as well as the number of hidden units, but this didn't achieve a better result. Further, running grid searches on those parameters, as well as the learning rate (and trying bidirectional LSTMs) didn't help either. It is possible that a much larger, more complicated RNN would help, but a lack of computing resources prevented this. Another possible reason for the poor performance is the nature of the tweets themselves: spelling mistakes, random characters and odd wording would increase the likelihood that the RNN might not have seen the previous information before, making any form of prediction difficult. For completeness, sentiment on our unseen tweets was still generated as a feature to our baseline classifiers, even though the accuracy wasn't promising.

5.4 Hive Method

Figure 5: Comparison of Performance with and without text on 100K Hive Tweets

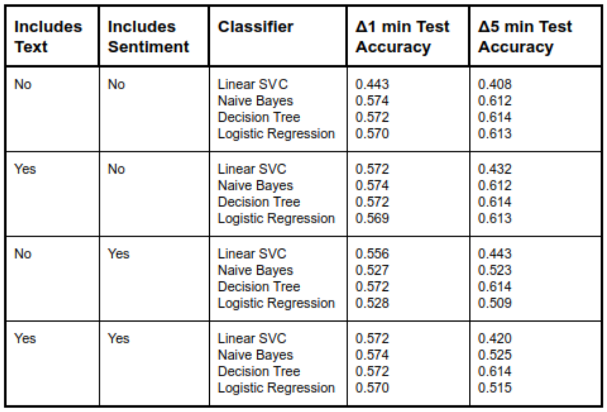


Includes Text	Classifier	Δ1 min Test Accuracy	Δ5 min Test Accuracy
No	Linear SVC	0.443	0.408
	Naive Bayes	0.574	0.612
	Decision Tree	0.572	0.614
	Logistic Regression	0.570	0.613
Yes	Linear SVC	0.572	0.432
	Naive Bayes	0.574	0.612
	Decision Tree	0.572	0.614
	Logistic Regression	0.569	0.613

Out of all the baseline classifiers, Decision Trees performed best but only slightly. A possible explanation could be that trees have an advantage in non-linear mapping, and complex market structure is almost certainly non-linear. It should be noted that a Dummy classifier, which chooses the most common class scored 0.570 and 0.598 respectively thus our predictive power is very weak.

5.5 Direct Method

Figure 6: Comparison of Performance with and without Sentiment on 10K Direct Tweets



Includes Text	Includes Sentiment	Classifier	Δ1 min Test Accuracy	Δ5 min Test Accuracy
No	No	Linear SVC	0.443	0.408
		Naive Bayes	0.574	0.612
		Decision Tree	0.572	0.614
		Logistic Regression	0.570	0.613
Yes	No	Linear SVC	0.572	0.432
		Naive Bayes	0.574	0.612
		Decision Tree	0.572	0.614
		Logistic Regression	0.569	0.613
No	Yes	Linear SVC	0.556	0.443
		Naive Bayes	0.527	0.523
		Decision Tree	0.572	0.614
		Logistic Regression	0.528	0.509
Yes	Yes	Linear SVC	0.572	0.420
		Naive Bayes	0.574	0.525
		Decision Tree	0.572	0.614
		Logistic Regression	0.570	0.515

Again, Decision Trees performed best by a slight margin. It should be noted that due to memory constraints, only 10K tweets were used, but this is not nearly enough to make any meaningful prediction, thus this model is heavily overfitting, even more so than Hive method. Dummy classifier performed only slightly worse, 0.572 and 0.610, and we are unable to gain any statistically significant edge.

6 Limitations & Future Plans

The most evident constraints for our work was certainly time and memory. With more computation time and speed there are many more creative techniques we wish to test. For instance, as was seen in (Beckmann, 2017) , societal trends can be certainly measured over long periods of time. A possible long term study could include collecting hive data over the course of a year, then training an RNN for market swings at 1 hour or 1 day intervals. Such an approach would give time for sentiment to translate to collective day-trader decisions, and possibly be able to capture more market sentiment.

Direct method should be avoided, as it clutters the data with too many points, and when successive individual tweets share the same output anyways it makes little sense to separate them, particularly as it increases our run time exponentially.

Supposing a trend was indeed found, there would still be much work incorporating this into a full trading system. A fully automated trader would have to use some sort of reinforcement learning such as Markov Decision Processes with Q-learning, as well as focus more on other indicators and market influences. From the 'committee of experts' point of view, Twitter sentiment could very well contribute some value, but given the complexity of market trends, other features must be considered.

7 Conclusion

Stock market prediction will continue to be a very pertinent topic, and with the rise of social media, which allows users to post their thoughts and emotions, and so will rise likelihood of correlation between the stock markets and Twitter. Our hypothesis was that more knowledge of the sentiment of tweets occurring at a given time might give us a better ability to predict the swings in the Bitcoin market. Unfortunately, due to a variety of constraints listed in the above section, we had inconclusive findings, in that we cannot say with any certainty that the market will go up or down given a set of recent tweets. However, we remain optimistic that collective sentiment can be measured and used to predict market swings, as has been shown by many other studies discussed.

8 Statement of Contributions

Jonathan worked on defining the problem, developing the methodology, implementing the sentiment classifiers, performing data analysis, and writing the report.

Victor worked on defining the problem, data collection, pre-processing, feature extraction, baseline classifiers, performing data analysis, and writing the report.

Acknowledgments

A huge thank you to Prof. Jackie Cheung and the team of TAs for teaching us.

References

Marcelo Beckmann. 2017. *Stock Price Change Prediction Using News Text Mining*. ResearchGate, https://www.researchgate.net/publication/313473231_Stock_Price_Change_Prediction_Using_News_Text_Mining.

Eric Gilbert C.J. Hutto. 2014. *VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text*. The Georgia Institute of Technology, http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf.

Hult. 2013. *Sentiment140 tweet Dataset*. Sentiment140, http://help.sentiment140.com/for-students.

Xiao-Jun Zeng Johan Bollen, Huina Mao. 2010. *Twitter mood predicts the stock market*. Journal of Computational Science, https://arxiv.org/abs/1010.3003.