Twitter Sentiment Analysis for Bitcoin Price Prediction

Sara Abdali
saraabd@stanford.edu

Ben Hoskins
bhoskins@stanford.edu

1 Introduction

All investing carries a certain degree of risk. While an investor might believe an asset's value will increase over time, when they purchase an asset it is almost impossible to know whether they will gain or lose money on that investment. Being able to predict the future price movement of an asset is consequentially an extremely powerful tool for any investor.

2021 has seen extremely volatile asset prices for both "meme stocks" and cryptocurrencies. These price movements seem to be completely removed from the asset's underlying value, and are instead shaped largely by the sentiment of investors. Being able to measure the sentiment of the average investor could, therefore, be useful in making projections about the future movements of asset prices. Since the average sentiment of investors is almost impossible to measure, we decided to test whether the average sentiment of tweets about a specific asset could be used as a substitute.

In order to narrow the scope of our paper and improve the usefulness of our model, we decided to focus solely on Bitcoin and its price movements. Our model took tweets mentioning Bitcoin and aggregated them into one-minute "buckets" to use as an input. We then used Naive Bayes and Support Vector Machines (in two separate models) to output a prediction about whether the price would increase or decrease over the following 24 hour period. We also compared these models to a logistic regression model that used features generated by feeding our twitter data into a language model called BERT (Bidirectional Encoder Representations from Transformers).

2 Related Work

Due to the significant financial upside of being able to predict the price movements of assets, many researchers have attempted to use online sentiment to predict how the value of stocks and cryptocurrencies will change.

A. Derakhshan and H. Beigy's *Sentiment Analysis on Stock Social Media for Stock Price Movement Prediction* focused on a relatively large number of stocks and used both English and Persian text from message boards such as Yahoo Finance [1]. They used both SVM and Neural Network models and attained an average accuracy of 56% in predicting positive or negative price movements, giving us a useful benchmark to compare our results to. In addition to being primarily predicting stock prices rather than cryptocurrencies, this paper also differed from our own by focusing on 18 US equities compared to our single currency. Our decision to focus on a single asset was made in order to increase size of the dataset on which we could train our model. Additionally, we chose to use Twitter due to both ease of data collection and also due to it being a more current platform than the Yahoo Finance dataset, which was collected during 2012 and 2013.

D. R. Pant, et al.'s *Recurrent Neural Network Based Bitcoin Price Prediction by Twitter Sentiment Analysis* was much more similar to the analysis we carried out [2]. Like our own, their model used Twitter data as an input and attempted to predict the future price of Bitcoin. However, while their paper achieved a relatively high accuracy it was also reliant on manually labeling tweets. This led to a relatively small dataset of 7454 Tweets, 1669 of which were deemed neutral or irrelevant. To increase the size of our dataset and potentially increase the usefulness of our model, we decided to instead

use future price movements to automatically label our tweets and assigned an average sentiment to groups of tweets rather than manually labelling each individual data point. Additionally, their model also incorporated historical price movements in their predictions. We decided not to do the same in order to test the predictive power of only using Twitter sentiment.

J. Devlin, et al.'s *BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding* provided details on the usage of the BERT model utilized in our experiments [3]. BERT allowed us to pre-train on our tweet texts using the entire context of the tweet rather than just its individual words.

3 Dataset and Features

3.1 Tweets

We initially planned to use the Twitter API to download tweets mentioning Bitcoin over a two week period to create a dataset. Unfortunately, the API placed a cap on the number of tweets we could download, meaning we were initially limited to a few thousand tweets per day. To expand this into a more generalizable dataset, we combined the tweets we had already downloaded with a larger dataset from Kaggle [5]. All tweets in this combined dataset mentioned Bitcoin in either a hashtag or in the main text of the tweet and were published between January and late May of 2021. This combined dataset contained 272,304 tweets.

We then sorted these tweets into "buckets" based upon the time they were published, segmenting the entire dataset into one-minute intervals. We were left with 34,294 buckets, each with a unique starting time. To extract features from these buckets, we bundled all the tweets in each bucket together. We then tokenized the aggregated text and removed all punctuation, emojis, urls, and stopwords. Stopwords are words that are neutral and do not aid us in the classification task (such as I, is, her, etc.)

Below is an example of the buckets of tweets before being processed to extract features:

Timestamp	Tweets
1612988520	#BITCOIN IS A BUY!! HOLD #BTC AND YOU WILL BE ...
1612988460	WSBChairman Own half of one doing alt coin tr...
1612988400	#Bitcoin #BTC #XBT If $BTC price temporarily ...
1612988340	Huge Changes Begin Quietly! #digitalassetmanag...
1612988280	$aave #aave target for $1k is ahead move is...

To extract features from the preprocessed text, we used one-hot encoding. We also tried extracting features using BERT, which will be further explained in the our methods section. We then divided the final dataset of extracted features into training, validation, and testing sets. We randomly assigned 20% of our data to the testing set, 20% to validation set, and the remaining to the training set.

3.2 Price Data

We collected historical Bitcoin price data from Binance's public dataset [6]. This data was provided in monthly files and provided accuracy up to the closest minute.

While each entry in the dataset contained 12 data points, we only used "Open time" and "Close" in our model. "Open time" was a UTC timestamp denoting the time at the beginning of each one-minute interval. We could then use these timestamps to align our labels with their corresponding tweet buckets. "Close" was the price of Bitcoin in US dollars at the end of that interval, which we could then use to find the ensuing change in price.

To create a label for the set of tweets observed in minute T, we first calculated

$$\%\Delta Price_T = \frac{Price_{T+1440} - Price_T}{Price_T}$$

Where $\%\Delta Price_T$ is the percentage change in Bitcoin's price from time T over the ensuing 24 hours.

We then used these changes in price to assign labels to each of the sets of tweets. We performed experiments using two different label methods. The first assumed sets of tweets had either a "bullish"

(positive) sentiment or a "bearish" (negative). We set the label to 1 if the price increased in the day following the tweets and -1 if the price decreased.

The second took into account situations where groups of tweets could be neutral by adding a third label. If the percentage change in price was in the top 80% of positive changes in price, the label was set to 1. Likewise if the percentage change in price was in the lowest 80% of negative changes in price, the label was set to -1. All remaining labels were set to 0.

4 Methods

4.1 Naive Bayes

We used Naive Bayes as a baseline algorithm for sentiment classification.The Naive Bayes algorithm assumes that the tokens in a tweet are conditionally independent of each other given the label of the tweet. Following this assumption, and using a Laplace smoothing parameter of 1, the Bernoulli Naive Bayes model yields:

$$\phi_{j|y=1} = \frac{1 + \sum_{i=1}^{n} 1\left\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\right\}}{2 + \sum_{i=1}^{n} 1\left\{y^{(i)} = 1\right\}} \tag{1}$$

$$\phi_{j|y=0} = \frac{1 + \sum_{i=1}^{n} 1\left\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\right\}}{2 + \sum_{i=1}^{n} 1\left\{y^{(i)} = 0\right\}} \tag{2}$$

4.2 Support Vector Machines

Another model that has been extensively used for text classification is Support Vector Machine. SVM is a classification model that finds a linear separation that optimizes the geometric margin between different labels of the data. The problem SVM optimizes for can be formulated as:

$$\min_{w,b} \frac{1}{2}\|w\|^2 \tag{3}$$

$$\text{s.t. } y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1, \quad i = 1, \ldots, n \tag{4}$$

4.3 BERT

We also used BERT pretrained models for feature extraction. BERT is essentially an NLP model that learns the contextual relationship between words [3]. Unlike other models, BERT is bidirectional, meaning it considers not only sentences to the left of a phrase but also to the right [3].

Each input sequence that is encoded with BERT starts with a special classification (CLS) token. This token is often used in classification problems as representative of the entire sequence. We used the vector representation of this token as the input to our training model.

BERT is a relatively new natural language processing technique. Despite its effectiveness, it has not been used much in sentiment analysis for financial assets and securities. We tried to bring some novelty to this field by using BERT as a feature extractor.

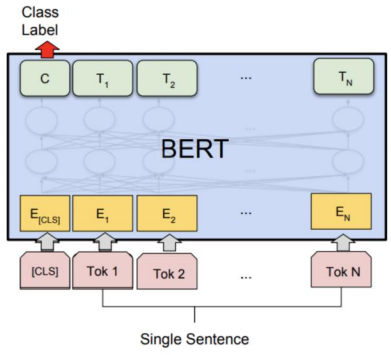

Figure 1: Graphical Representation of BERT [3]

5 Experiments/Results/Discussion

5.1 Naive Bayes

After splitting data into train, val, and test sets,
we used the sklearn implementation of the Naive Bayes model for training. We tuned the hyper-parameter alpha (Laplace smoothing parameter) for the model. Fig. 2 shows how the accuracy of our model varies with changing alpha. This model had a training accuracy of 77% and a peak **accuracy of 58%** on the test set when alpha = 0.01.

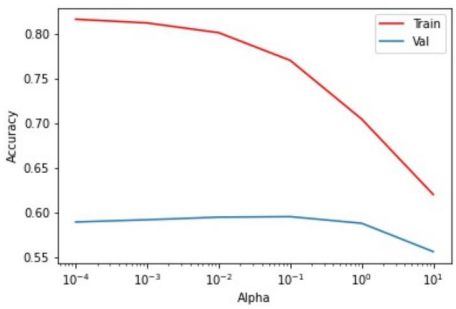

Figure 2: Naive Bayes accuracy vs. smoothing parameter

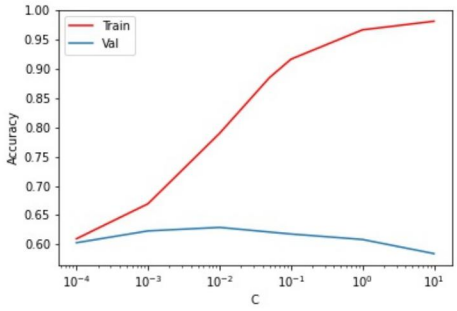

Figure 3: SVM accuracy vs. regularization parameter

5.2 Support Vector Machines

For this algorithm, we used sklearn implementation of linear support vector classifiers for training. We tried fitting our data to support vector machines with radial and sigmoid kernels but they did not seem to work any better than a random classifier. After tuning regularization parameter C, which is proportional to how much the algorithm penalizes wrong classifications, we achieved a training accuracy of 78% and a peak **accuracy of 63%** on the test set when $C = 0.01$. Fig. 3 shows how the accuracy of model on train and val sets versus different values of C. As expected the training accuracy keeps going up with increasing C, since the optimizer penalizes wrong classifications more strictly and over-fits to the training set more heavily.
We also used this algorithm on our dataset with 3 labels (bullish, neutral, bearish) which yielded **an accuracy of 53%** on the test set and 71% on the training set with optimal $C = 0.01$. The following table shows the precision of this algorithm in each label.

# Labels	Bullish	Neutral	Bearish
2	61.8%	-	63.2%
3	52.8%	41.2%	55.7%

Precision values for SVM

# Labels	Bullish	Neutral	Bearish
2	74.7%	-	48.7%
3	76.0%	9.2%	50.7%

Recall values for BERT

As we can see, the model underperforms when it comes to classifying neutral tweets. This might be due to the fact that a bucket of tweets labeled neutral might have an equal number of both very bullish and bearish tweets, and it might be harder for the algorithm to see how they balance each other out.

5.3 BERT + Logistic Regression

We used the vector representation of the CLS token as the feature for our model. We then fed these features into a logistic regression model, returning an accuracy of 58% with 2 labels and 50% with 3 labels on our test set.

# Labels	Bullish	Neutral	Bearish
2	58.8%	-	58.0%
3	50.6%	27.2%	49.5%

Precision values for BERT

# Labels	Bullish	Neutral	Bearish
2	71.5%	-	43.9%
3	72.4%	4.4%	46.3%

Recall values for BERT

5.4 Model Comparison

The result from our experiments show that in training using both 2 or 3 labels, SVM outperforms our baseline algorithm, Naive Bayes. Surprisingly, SVM also outperforms BERT. A possible reason behind this could be that a pretrained BERT model is not familiar with financial terms and online slang used by cryptocurrency investors. Additionally, BERT can only accept up to 512 characters while most of our tweet buckets were substantially larger than this limit. As a result, a large amount of information might have been lost while converting these buckets into data that could be fed into BERT.

The accuracy of the three models we tested, along with the number of labels used in each experiment, has been condensed into a single table below.

	Two Labels			**Three Labels**		
Model	NB	SVM	BERT	NB	SVM	BERT
Training	77.0%	78.1%	58.4%	68.1%	71.7%	52.3%
Testing	58.8%	63.2%	59.3%	41.7%	53.2%	49.5%

Our two-label methods returned an average accuracy of approximately 60% on our test set. This appears to be in line with previous attempts to make price predictions using sentiment analysis, with other researchers tuning their models to a comparable 56% as mentioned in our related works section [1]. Our models all significantly under-performed that of D. R. Pant et al, whose model attained an accuracy of 77.62% [2]. However, this model also used historical price data and manually labeled tweets in predicting future price movements meaning the results are not completely comparable.

6 Conclusion/Future Work

Given that markets are highly efficient systems, we believe our results are relatively significant. If the bitcoin market was perfectly efficient it would be impossible to make predictions since all available data would be immediately priced in. This means we would expect an accuracy of 50% - the same as a random classifier. However, we see that SVM outperformed a random classifier by 13 percentage points. This suggests that bitcoin markets are not entirely efficient and it is therefore possible to predict future prices with a relatively high accuracy.
Nonetheless, we think that there is much room for improvement. One of the biggest flaws in our training process is our labeling strategy. We think that the model can be significantly improved if we come up with better strategies to label the tweets with their true sentiment rather than just the market movements. Additionally, it might be worth testing whether grouping tweets by the minute is the optimal time frame, or whether hourly or daily buckets work more effectively. Finally, feeding BERT into more complicated models that can capture the nuances of our corpus such as deep neural networks can be a promising direction to lead this project towards.

Ultimately, our project has demonstrated that it is possible to predict Bitcoin price movements with relatively little data sourced solely from social media.

Contributions

In addition to performing a literature review and identifying the motivation of the project, Ben contributed to the project by collecting Bitcoin price data from Binance and converting it into usable labels. Additionally, he combined the Twitter API data pulled by Sara with the Kaggle dataset and converted it into the bucket form needed to feed into our model.
Sara contributed to the project by accessing the Twitter API and pulling relevant tweets. She also identified BERT as a viable option for the NLP portion of the project alongside other useful packages. Finally, Sara also carried out most of the training and testing of our models.
In the final paper, Ben wrote sections 1-3 while Sara focused on sections 4-6. Both contributed to editing and review of the document.

References

[1] A. Derakhshan and H. Beigy. “Sentiment Analysis on Stock Social Media for Stock Price Movement Prediction.” Engineering Applications of Artificial Intelligence, vol. 85, 2019, pp. 569–578., doi:10.1016/j.engappai.2019.07.002.

[2] D. R. Pant, P. Neupane, A. Poudel, A. K. Pokhrel and B. K. Lama, "Recurrent Neural Network Based Bitcoin Price Prediction by Twitter Sentiment Analysis," 2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS), 2018, pp. 128-132, doi: 10.1109/CCCS.2018.8586824.

[3] J. Devlin, et al. “BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding.” Proceedings of the 2019 Conference of the North, 2019, doi:10.18653/v1/n19-1423.

[4] R. Wigglesworth. “Rise of the Retail Army: the Amateur Traders Transforming Markets.” Subscribe to Read | Financial Times, Financial Times, 9 Mar. 2021, www.ft.com/content/7a91e3ea-b9ec-4611-9a03-a8dd3b8bddb5.

[5] K. Suresh, “Bitcoin Tweets, Version 8,” Kaggle, 13-Feb-2021. [Online]. Available: https://www.kaggle.com/kaushiksuresh147/bitcoin-tweets. [Accessed: 28-Jun-2021].

[6] "Monthly Klines, BTCUSDT, 1m" in Binance Market Data. [Online]. Available: https://data.binance.vision/?prefix=data/spot/monthly/klines/BTCUSDT/1m/. [Accessed: 28-Jun-2021]