

---

# Prediction of Bitcoin prices using Twitter Data and Natural Language Processing

---

**Eugene L.X. Wong**  
Pratt School of Engineering  
Duke University  
Durham, NC 27708  
eugene.wong@duke.edu

## Abstract

The influence of social media platform like Twitter had long been perceived as a bellwether of Bitcoin Prices. This paper aims to investigate if the tweets can be modeled using two different approaches, namely, the Naïve Bayes and LSTM models, to compute the sentiment scores in order to predict the Bitcoin price signal. Through the experiments conducted, the LSTM model indicates some degree of predictive advantage compared to the Naïve Bayes model.

## 1 Introduction

Cryptocurrency prices are often said to be erratic and unpredictable to some extent. However, popular social media platforms like Twitter had been proven to be a good bellwether for Bitcoin prices. Several questions have been raised such as to what extent a myriad of voices can influence Bitcoin prices? Do the prices fluctuate according to influencers' tweets or is the market efficient enough to eliminate the effects of the tweets? Consequently, the first objective of this paper is to investigate the effects of tweets on bitcoin prices by using machine learning models to compute the sentiment scores.

The proposed approach utilizes two different machine learning models, the generative and discriminative models. The generative model captures the joint probability the data, learns the probability estimates and likelihood to model the data points. Based on these parameter estimates, it then makes a prediction given an instance. On the other hand, the discriminative model captures the conditional probability of the data and learns the decision boundary between the classes. The decision boundary is then use to identify regions of the input class space that corresponds to each class during prediction. In this paper, the Naïve Bayes model and the Long Short Term Memory (LSTM) model are the generative and discriminative models used to model the tweets and to compute the sentiment scores. The second objective of this paper is to investigate how well the model performs, both qualitatively and quantitatively, when applied to synthetic and "real" data. The third objective is to compare the pros and cons when using either approaches.

The twitter data used for the analysis and experiments were gathered from Kaggle, and it contains 16 million tweets of different languages dating from January 2016 to November 2019. The Bitcoin prices were gathered from Gemini, a crypto exchange that provides minute level pricing data. In the later sections, the steps to preprocess, model and predict the data will be explained in deeper details.

## 2 Related work

There is a significant number of papers that worked on financial forecasting with NLP methods. This trend can be attributed by the surge of newer machine learning models been created in the recent

decade (1). The work on this paper draws inspiration from "NLP for Stock Market Prediction with Reddit Data" (2) where the author amalgamated different approaches such as sentence embedding, sentiment scores and convolutional neural network (CNN), together with social media data to predict the stock market with reasonable performance over the naive forecasting method. There is also another approach such the Valence Aware Dictionary and Sentiment Reasoner (VADER), a lexicon and rule-based sentiment analysis tool that is attuned to social media text to compute sentiment scores. A paper based on the VADER approach, 'A Complete VADER-Based Sentiment Analysis of Bitcoin (BTC)' managed to show that prices correlate well with sentiment score of tweets over a shorter timespan. The effects of tweets on the crypto market continues to evolve especially since the cryptocurrency status remains in the state of flux at present and it appears to be strongly influenced by public opinions on social media platforms. As such, the goal of this paper is to investigate the different models applied to twitter data in computing the sentiment scores that could well predict the Bitcoin prices.

### 3 Dataset and Features

#### 3.1 Dataset

The dataset contains various information pertaining to the tweet as shown in figure 1. The dataset extends across the periods from January 2016 to November 2019. However, due to demanding computation requirements, a subset of the dataset (January 2018 to November 2019) was used instead, of which 80% of the data will be used to train the model and the remaining 20% will be used to evaluate the model performance. In this section, the details of how the features were preprocessed, resampled and how the labels (positive: +1, negative: -1) were created will be described here.

id
user
fullname
url
timestamp
replies
likes
retweets
text
date

Figure 1: Twitter data features

#### 3.2 Data Preprocessing

##### 3.2.1 Tweets Preprocessing


In order to extract out meaningful text from each tweet, the following operations in the table below were performed. In addition, the words were also tokenized and lemmatized during the preprocessing. (Refer to code for extended details)

Preprocessing of Tweets	
Steps	Regular Expressions
Remove link	"http\S+"
Remove numbers	"([0-9])\w+"
Remove non-ASCII	"[\x00-\x7F]+"
Remove emojis	':-p', 'XD', etc.
Remove stopwords	'was', 'for', etc.
Remove emoticons	'u"\ufe0f"', etc.
Remove punctuations	!"%&'()*+,-./:;, etc.

### 3.2.2 Sentiment Score (Labels)

The sentiment score is computed based on the inter-day returns of Bitcoin prices as follows:

$$R_{cc} = \frac{x_c(t+1) - x_c(t)}{x_c(t)} \quad (1)$$

where  $R_{cc}$  is the general expression for Close-to-Close prices and the time,  $t$ , refers to the present trading day (now) and  $t+1$  refers to the prediction day. If  $R_{cc} > 0$ , then sentiment score equals 1. Likewise, if  $R_{cc} < 0$ , then sentiment score equals -1, and if  $R_{cc} = 0$ , then sentiment score equals 0. 

### 3.2.3 Merging Dataset and Cleaning

The tweets were resampled into minute level segmentation so as to match and merge with the Bitcoin pricing data. After which, rows that did not contain tweets (text) were removed.

## 4 Methods

### 4.1 Naïve Bayes

The Naïve Bayes is a generative model based on applying Bayes' theorem. It comprises of two types of probabilities computed from the training data, namely, probability of each class and the conditional probability of each class given each feature  $x$  value. This model also assumes strong independence between features of the data. Bayes' theorem is defined as follows: The Naïve Bayes is a generative model based on applying Bayes' theorem. It comprises of two types of probabilities computed from the training data, namely, probability of each class and the conditional probability of each class given each feature  $x$  value. This model also assumes strong independence between features of the data. Bayes' theorem is defined as follows:

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)} \quad (2)$$

The sentiment score of the the tweet is then calculated as follows:

$$\prod_{i=1}^m \frac{P(w_i|pos)}{P(w_i|neg)} \quad (3)$$

where  $i$  is the  $i$ th unique word and  $P(w_i|class)$  is the probability of the word appearing in that class. However, there are some instances where the word may only appear in one class in the entire corpus. This will result in a zero probability for that word. As such, the Laplacian smoothing is used to eliminate this situation. With lapacian, the probability of the word given a class is given as follows:

$$P(w_i|class) = \frac{freq(w_i, class) + 1}{N_{class} + V} \quad (4)$$

where  $N_{class}$  is the frequency of all words in class and  $V$  is the number of unique words in the vocabulary. The log is applied to equation (2) to avoid numerical overflow issues as  $m$  gets too large.

$$\log\left(\frac{P(pos)}{P(neg)} \prod_{i=1}^m \frac{P(w_i|pos)}{P(w_i|neg)}\right) \quad (5)$$

Note that  $\frac{P(pos)}{P(neg)}$  is used to counteract the imbalance of positive to negative labels. With the log, equation (4) can be broken down into log prior and log likelihood as follows:

$$\log\left(\frac{P(pos)}{P(neg)}\right) + \sum_{i=1}^m \frac{P(w_i|pos)}{P(w_i|neg)} \quad (6)$$

The log likelihood is computed for each word and the sum of all the log likelihoods for words appearing in a tweet will give the sentiment score. It's important to note that the log likelihood value ranges from  $-\infty$  to  $+\infty$ , where 0 is neutral.

## 4.2 Long Short Term Memory (LSTM)

The LSTM model is a derivative of the RNN, a sequential feedforward neural network that utilizes internal states to process variable length sequence. The underlying differences lies in the handling of the vanishing gradient problem during backpropagation. There are three main components to the LSTM cell, namely, the forget gate, the input gate and the output gate. The input gate informs what new information will be stored in the cell state. The forget gate indicates what information will be discarded away from the cell state. The output gate is used to provide the activation of the final output of the LSTM model. The equations of gates are as follows:

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \quad (7)$$

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f) \quad (8)$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o) \quad (9)$$

$$\tilde{c}_t = \tanh(w_c[h_{t-1}, x_t] + b_c) \quad (10)$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \quad (11)$$

$$h_t = o_t + \tanh c_t \quad (12)$$

where  $i_t$  is the input gate,  $o_t$  is the output gate and  $f_t$  is the forget gate.  $w_x$  refers to the weight at the respective gate(x) and  $h_{t-1}$  is the hidden state from the previous timestamp. The cell state which is also known as the long term memory, is represented by  $c_t$  and  $h_t$  is the hidden state, representing the short term memory. Note that  $t$  is current timestamp(t).

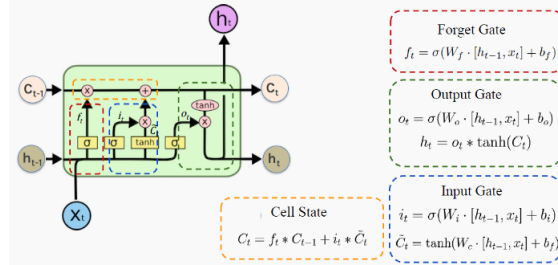


Figure 2: LSTM Model (Source: P. Protopapas, Harvard University)

## 5 Implementation

### 5.1 Naïve Bayes

This section will explain the core functions used to build the Naïve Bayes model.

#### 5.1.1 count\_tweets

The 'count\_tweets' function is used to map the (word,label) pair to the frequency of times it appears in the corpus. For example, the pair [('happy',1):5] implies that the word 'happy' associated with the positive sentiment tweet appears five times in the entire corpus.

#### 5.1.2 train\_naive\_bayes

The train\_bayes function is used to calculate and return the log prior and the log likelihood in equation (6). The inputs to this function are the dictionary, that is (word, label):frequency, the list of tweets, and the list of labels [-1, 0, 1] corresponding to the tweets.

### 5.1.3 naive\_bayes\_predict

The naive\_bayes\_predict function predicts the sentiment score by taking the sum of all the likelihoods of each word in the tweet and the log prior.

## 5.2 LSTM

This section will explain the important details of the LSTM model.

- Each tweet is tokenized and padded at the end of the sequence to ensure a fixed length. The maximum sequence length was set to 200.
- Spatial dropout drops entire 1D feature maps instead of individual elements; set to 0.25.
- The number of units in the LSTM was set to 50.
- The dropout was set to 0.5 where it drops out the input/output gate.
- The recurrent dropout was set to 0.5 where it masks connections between the recurrent units.
- The dropout was set to 0.2 just before final output.
- The final output layer was to a sigmoid so as to generate an output between 0.0 to 1.0.

For full details of the model architecture is illustrated in figure 3.

```
Model: "sequential_5"
```

Layer (type)	Output Shape	Param #
embedding_2 (Embedding)	(None, 200, 32)	15848512
spatial_dropout1d_2 (SpatialDropout1D)	(None, 200, 32)	0
lstm_2 (LSTM)	(None, 50)	16600
dropout (Dropout)	(None, 50)	0
dense_2 (Dense)	(None, 1)	51

```
=====  
Total params: 15,865,163  
Trainable params: 15,865,163  
Non-trainable params: 0  
None
```

Figure 3: LSTM Model parameters

## 6 Analysis/Results/Discussion

### 6.1 Qualitative Performance

The Naïve Bayes and LSTM models were evaluated based on 14 synthetic tweets as shown in Figure 4 to evaluate if the sentiment scores were reasonable in the context of a human interpreting the same text. There are two approaches in evaluating the qualitative performance. The first approach is to determine if the **predicted sentiment scores could be use to compare** if one tweet is more positively sentimental over the other. For this approach, the following steps were taken: first, compare two similar tweets and compute the respective sentiment scores. Next, calculate the difference between the two scores. If the difference is more than zero, it indicates that the model could determine which tweet is more positively sentimental over the other. Based on the Naïve Bayes model prediction, it was able to **distinguish 6 out of 7 examples** correctly. The same evaluation is approach is applied to the LSTM model and it was only able to distinguish 4 out of 7 examples as shown in Figure 5. From these observations, it can be concluded that the Naïve Bayes model does a better job at **distinguishing similar tweets compared** to the LSTM model. Approach 1 will not be applied to the real dataset due to limitations in getting similar tweets for comparison.



The second approach is to evaluate if there's an **observable threshold** value (sentiment score) that can clearly define the positive and the negative tweets. For example, comparing the positive tweet in example [1] to the negative tweet in example [5], it is observable that a sentiment score of 1.0 is a good decision boundary. However, this decision boundary is not applicable in example [2] since the sentiment score of the negative tweet is clearly larger than the positive sentiment score of the tweet in example [1]. Now, when the Naïve Bayes model was applied it to the real dataset as shown in Figure 6, the observation is similar and there's no boundary that separates the positive and negative sentiment tweets. Likewise, the LSTM model was also applied to the same dataset and the results were similar (no conclusive boundary) to the Naïve Bayes model as shown in Figure 7. In conclusion, **there's no universal threshold that can define this boundary**. Consequently, the Naïve Bayes and the LSTM model **does not perform well in predicting if the tweet is positive without a relative comparison**.

Examples	Sentiments	Tweets (Synthetic Data)	Sentiment Score	Differences
1	Positive	BTC gonna increase sky high	1.57	0.47
	Negative	BTC big drop is coming	1.1	
2	Positive	Elon musk thinks btc is here to stay	2.54	0.09
	Negative	Elon musk thinks btc is not to keep	2.45	
3	Positive	BTC will reach 80k end of the year -	2.73	0.14
	Negative	BTC drop to 0 end of the year	2.59	
4	Positive	Market increase will continue	1.33	0.1
	Negative	Market loss will continue	1.23	
5	Positive	Buy btc quickly	0.74	-0.11
	Negative	Sell btc quickly	0.85	
6	Positive	economy is expected to grow	1.58	0.02
	Negative	economy is going into recession	1.56	
7	Positive	Bloom time for crypto, elon time says buy more BTC now!	3.74	0.25
	Negative	Dangerous moment for crypto, elon time says sell BTC now!	3.49	

Figure 4: Sentiment scores computed from synthetic data using Naïve Bayes Model

Examples	Sentiments	Tweets (Synthetic Data)	Sentiment Score	Differences
1	Positive	BTC gonna increase sky high	0.42	-0.11
	Negative	BTC big drop is coming	0.53	
2	Positive	Elon musk thinks btc is here to stay	0.49	-0.03
	Negative	Elon musk thinks btc is not to keep	0.52	
3	Positive	BTC will reach 80k end of the year	0.58	0.02
	Negative	BTC drop to 0 end of the year	0.56	
4	Positive	Market loss will continue	0.55	0.01
	Negative	Market increase will continue	0.54	
5	Positive	Sell btc quickly	0.5	0
	Negative	Buy btc quickly	0.5	
6	Positive	economy is expected to grow	0.51	0.02
	Negative	economy is going into recession	0.49	
7	Positive	Bloom time for crypto, elon time says buy more BTC now!	0.48	-0.11
	Negative	Dangerous moment for crypto, elon time says sell BTC now!	0.59	

Figure 5: Sentiment scores computed from synthetic data using LSTM Model

Sentiments	Tweets (Real Data)	Sentiment Score
Positive	bitcoin btc price change h market cap ranking bitcoin btc bitcoin high	4.4
Negative	bitcoin bitcoin fail hashpower centralized tho miningpoolchinaattackve	2.84
Positive	ruble established bitcoin established bitcoin halving prediction co dh	3.6
Positive	hurry pump alert ncash price ha moved e btc within minute koinex	4.43
Negative	smart money said skip bitcoin bet blockchain co ktdmshzvsq bobby shmur	3.86

Figure 6: Sentiment scores computed from real data using Naïve Bayes Model

Sentiments	Tweets (Real Data)	Sentiment Score
Positive	bitcoin btc price change h market cap ranking bitcoin btc bitcoin high	0.47
Negative	bitcoin bitcoin fail hashpower centralized tho miningpoolchinaattackve	0.69
Positive	ruble established bitcoin established bitcoin halving prediction co dh	0.62
Positive	hurry pump alert ncash price ha moved e btc within minute koinex	0.33
Negative	smart money said skip bitcoin bet blockchain co ktdmshzvsq bobby shmur	0.59

Figure 7: Sentiment scores computed from real data using LSTM Model

## 6.2 Quantitative Performance

The Naïve Bayes and LSTM model were evaluated quantitatively based on the following metrics: accuracy, precision and recall. Based on Figure 8, it can be observed that the Naïve Bayes performance is as good as random given that the accuracy is only 50%. The recall score of 1.0 for positive sentiment tweets (+1) indicates that the model has no sensitivity towards the negative sentiment tweets. The cause for this abnormal sensitivity can be attributed to the fact that the predicted sentiment scores for the synthetic and real data were all greater than zero. Instead of taking zero as the arbitrary value as the decision boundary, the median of the sentiment scores based on training data can be used instead. That is, if sentiment score is above 156, it is 1.0, else it is -1.0. Consequently, recall becomes more balance between the +1 and -1 sentiment tweets as shown in Figure 9. However, the accuracy and precision did not perform any better than before. With the LSTM model, the accuracy when predicting the synthetic tweets' sentiment was 36% and the precision for both positive and negative sentiment tweets were 25% and 40% respectively as shown in Figure 10. However, when the LSTM model is applied to the real data, it's performance significantly improved to an accuracy of 51% with precision of both sentiments also reaching 51%. In conclusion, based on the quantitative metrics, both the Naïve Bayes and LSTM model do not perform well on the synthetic data. However, it is noticeable that the LSTM model perform better than the Naïve Bayes model when tested on the real data. Although the LSTM model has an accuracy slightly above 50% (51.29%) over 61,613 samples, it does indicate a limited predictive power in forecasting the market sentiment compared to the Naïve Bayes model.

	precision	recall	f1-score	support
-1	0.00	0.00	0.00	7
1	0.50	1.00	0.67	7
accuracy			0.50	14
macro avg	0.25	0.50	0.33	14
weighted avg	0.25	0.50	0.33	14

Figure 8: Accuracy, precision and recall for Naïve Bayes model on synthetic data

	precision	recall	f1-score	support
-1	0.47	0.45	0.46	30849
1	0.48	0.50	0.49	30764
accuracy			0.48	61613
macro avg	0.48	0.48	0.47	61613
weighted avg	0.48	0.48	0.47	61613

Figure 9: Naïve Bayes model on real data with new arbitrary threshold



	precision	recall	f1-score	support
-1	0.25	0.14	0.18	7
1	0.40	0.57	0.47	7
accuracy			0.36	14
macro avg	0.33	0.36	0.33	14
weighted avg	0.33	0.36	0.33	14

Figure 10: LSTM model on synthetic data

	precision	recall	f1-score	support
-1	0.51	0.48	0.50	30849
1	0.51	0.54	0.53	30764
accuracy			0.51	61613
macro avg	0.51	0.51	0.51	61613
weighted avg	0.51	0.51	0.51	61613

Figure 11: LSTM model on real data

### 6.3 Advantages and Disadvantages of both approaches

#### 6.3.1 Quality and Correctness

Based on the quantitative evaluation of both models, the **LSTM model has a slightly better accuracy** and precision compared to the Naïve Bayes model. This could be attributed to the LSTM model been able to capture the sequential relationship between the information from the previous and current timestamp to make a prediction in the next timestamp as indicated in equations (7) to (12). In contrast, the Naïve Bayes model strongly assumes that the features (words) are independent and computes this probability solely based on the frequency a word appears in either positive or negative tweets. As tweets usually have some language structure, it is more likely that the LSTM model does better at capturing this relationship compared to the Naïve Bayes model.

#### 6.3.2 Computational Requirements

The Naïve Bayes model can be trained at a fraction of time compared to the LSTM model. For this particular application, the Naïve Bayes model took an approximate 10 minutes to train the model as compared to 5 Hours and 35 minutes when training the LSTM model with 5 epochs. The reason for this extensive time difference lies in the backpropagation process to optimize the weights of the hidden layers which is dependent on the number of neurons and the learning rate. Comparatively, the Naïve Bayes does not need to perform backpropagation, but simply apply arithmetic operations to calculate the prior probabilities. During the prediction process, the Naïve Bayes model is also faster than the LSTM model because it can easily retrieve the prior probability of each word and use it to calculate the posterior. Contrary, the LSTM model has to pad the sequence first with zeros to ensure a specific length before processing it.

#### 6.3.3 Interpretability

The **Naïve Bayes** model has an advantage over the LSTM model in terms of interpretability because the equations (2) to (6) can easily explain how the probability of the word given a class can be calculated based on the dictionary of words. There are several reasons why the LSTM model is difficult to interpret. First, the embedding layer changes the form of the original input. Second, the many-to-many neuron connections between layers coupled with dropout makes it difficult to trace the underlying decision making process. Third, application of the activation function (i.e. sigmoid function used in this application) has a non-linear effect on the output. As a result, it is difficult to decode the reason why the LSTM model computes a certain score for a tweet.



## 6.4 Potential Improvements

There are several areas the models can be improved on. First, although much attempts have been taken to clean the tweets before training the model, there are some words like 'qwxxwww' or 'zzzzzz' that does not provide any useful information for model training. They could be identified earlier and added into the 'stopwords' dictionary. Second, the number of layers designed for the LSTM model was fixed. This was mainly due to the strenuous computational requirements to train and evaluate the model each time. With more GPUs, the model can be fine-tuned further and may increase in it's accuracy. Lastly, the labels were created based on Bitcoin price returns. The assumption is that the sentiment scores and the price returns are highly correlated and not subjected to external influence such as macroeconomic conditions or that some influencers yield stronger effects with their tweets. As such, it's important to investigate deeper on the impact of external influence on the price returns (sentiment score) compared to the effect of the tweets.

## 7 Conclusion/Future Work

In conclusion, even though the accuracy was not significantly greater than 50%, the LSTM model indicates some predictive power compared to the Naïve Bayes model in forecasting the Bitcoin price signal. However, when synthetic data were used to evaluate the models, both models fail to capture the right sentiment scores. It is likely that the language structure used in real tweets is different from the correct (close to) English form of the synthetic tweets. Each approach has it's advantageous and disadvantageous positions, and depending on circumstances, the LSTM model might be a better for sentiment score prediction because of it's capability to capture sequential relationships in the tweet. Finally, there are several areas for improvements that should be looked into for future work, namely, better data preprocessing, optimizing the hyperparameters further and addressing the human 'biasness'.

## References

- [1] Xinwen Zhang *The Evolution Of Natural Language Processing And Its Impact On AI* Forbes, Nov 2018
- [2] Muxi Xu *NLP for Stock Market Prediction with Reddit Data* , 2021
- [3] Toni Pano and Rasha Kashef *A Complete VADER-Based Sentiment Analysis of Bitcoin (BTC)*, 9 Nov 2020