Algorithmic Trading of Bitcoin using Twitter Sentiment and Technical Analysis

A Report by Jason R Becker

September 30, 2017

Contents

1 Definition

1.1 Project Overview

The ability to predict the future value of stocks and other traded financial instruments has long been studied with the ambition to provide superior returns and hefty profits. The recent rise of social media has provided an additional measure, public sentiment, which has been studied to this end. Bitcoin, one of numerous digital payment systems known as cryptocurrencies, has experienced massive appreciation in recent months, leading to many speculators entering the Bitcoin market. Many of these speculators come from technological backgrounds and use social media communities to broadcast and probe predictions. The present project integrates social media sentiment with conventional stock/currency analysis using an artificial neural network to predict one hour movements in Bitcoin. An Accuracy of 57% was achieved for direction of price movement, and profits from simulated market trading surpassed profits from either holding Bitcoin or holding the S&P 500 over an identical time-frame. This work demonstrates the ability to utilize social media sentiment in conjunction with deep learning techniques to improve predictions for future value of a financial instrument.

1.2 Background

Recently, social media sentiment has been used to predict market movements of both securities and other financial instruments [1]. Stock market prediction has historically been a difficult task as it is affected by many complicated, interconnected factors including macroeconomic conditions, political events, and investor sentiment. The market has long been considered a "random walk," a stochastic process which consists of a succession of random steps with moves up or down with equal probability [2]. However, more recent studies utilizing machine learning and artificial intelligence techniques have achieved predictability of 65%, overcoming a true random walk predictability of 50% [3]. This suggests that relevant information coupled with effective prediction techniques can be used to predict financial markets. Sentiment from popular social medias sites such as Twitter, a free service allowing its 330 million active users to micro-blog 140 character messages, has been particularly studied for its ability to provide stock predictions on short time scales [1]. More recently Souza et al. have found that of the thirty DIJA components, Twitter sentiment had linear causality with three and non-linear causality with ten components, an overlooked part of previous studies which may have undervalued social media sentiment's impact on stock price [8].

Cryptocurrencies are currently gaining traction as an alternative to fiat currencies, consisting of hundreds of decentralized "crypto coin" types. Each coin has its own unique cryptographic foundation, enabling secure peer-to-peer transactions through several exchange networks. Since its inception in 2009, Bitcoin (XBT) has been the most traded cryptocurrency, increasing in value over 50,000 from $0.08 per coin in 2009 to a peak of $4,965.00 per coin in 2017 and is often used as the base currency on cryptocurrency exchanges, similar to USD in forex markets. An affinity for Bitcoin and other cryptocurrencies within tech circles has led to many tech savvy traders and investors entering Bitcoin exchanges, fueled by promises of Bitcoin disrupting physical currencies. Many of these technically proficient traders are active on social media, and frequently post about and discuss cryptocurrencies on such platforms. Since most Twitter posts, or tweets about Bitcoin come from speculators and traders themselves, I believe the sentiment value of tweets about Bitcoin may be far superior to that of DIJA components, where sentiment from investors and traders is mitigated by tweets from everyday people around the globe. Further, I believe this creates an ideal

opportunity to apply machine learning and artificial intelligence principles to form predictions about future Bitcoin value using social media sentiments.

1.3 Problem Statement

The aim of this project is to design an algorithm using machine learning and artificial intelligence techniques that is capable of predicting the future value of Bitcoin from social media sentiment and Bitcoin value history. This complex problem can be simplified into a classification problem in which an artificial neural network (ANN), given an input with recent social media sentiment and technical factors calculated from time series data of previous Bitcoin values, is able to make an optimal trading strategy to *Buy*, *Sell*, or *Hold*.

1.4 Metrics

The ANN solution model can be measured on two metrics, overall accuracy in predicting the direction which Bitcoin value will move, and total returns that would be generated if the algorithm were implemented and actively traded in the market. The first metric provides a simple score for the ANN which can be used to optimize the prediction capabilities of the model. The second metric is more complex, as transaction fees and slippage must be taken into account for accurate return estimates, but is also a superior metric, as an algorithm that can accurately predict the market but is unable to profit from those predictions has no practical value.

2 Analysis

2.1 Data Exploration

The datasets for this problem consisted of raw social media posts and time series data of Bitcoin trading. Social media posts were be restricted to tweets form Twitter which were in English and contain the keyword, *bitcoin*. Raw tweets have been collected using Tweepy, an open-source Python library for accessing the Twitter API [4]. The keyword is searched in real-time, with tweets containing the token saved to a .csv file along with each tweets respective *unique ID*, *user ID* of the poster, and *time stamp*. Sentiment analysis of each individual tweet was performed using VADER (Valence Aware Dictionary and sEntiment Reasoner), a lexicon and rule-based sentiment analysis tool specifically attuned to sentiments expressed in social media to determine positivity, negativity, and neutrality of each tweet [5]. Additional sentiment analysis was performed using TextBlob, a natural language processing (NLP) library for processing textual data which determines polarity and subjectivity of each tweet [6]. The results were aggregated in one hour increments to summarize the average sentiment for each hour, while simultaneously recording the total number of tweets recorded in each one hour period.

Historical Bitcoin value history was retrieved for hourly periods from Bitcoin Charts [7]. *Open*, *High*, *Low*, *Close*, and *Volume* (OHLCV) from Kraken Bitcoin Exchange in USD were collected for all hourly periods ranging from two weeks before the start of tweet collection to the end of the study. This allowed technical indicators requiring previous information (e.g., moving average) to be calculated for all pertinent time periods. Technical stock indicators, features of stocks calculated with OHLCV data to gain insight into supply and demand of securities, were calculated for Bitcoin at each one hour time period. The resulting technical indicators were merged with the aggregated Twitter sentiment dataset to provide an input features matrix for the ANN. This matrix was split into chronological training, cross-validation, and testing sets. All features were scaled using the

mean and standard deviation of the training set. The *Close* value of Bitcoin was used to determine hourly return for each hour in the dataset. A value of 1 was assigned to hours with positive returns, while a value of zero was given to hours corresponding to negative returns. These binary values were used as the output factor variable, paired with the feature vectors for the immediately preceding one hour time periods.

2.2 Exploratory Visualization

To begin, the number of tweets per hour was plotted as a function of time. Since historical Twitter data is not free to obtain, the dataset collected for this study required a live program running for all hours of collection. Many gaps can be observed in Figure 1 due to errors within Tweepy, the computer used for collection crashing, power outages, and loss of internet connection.

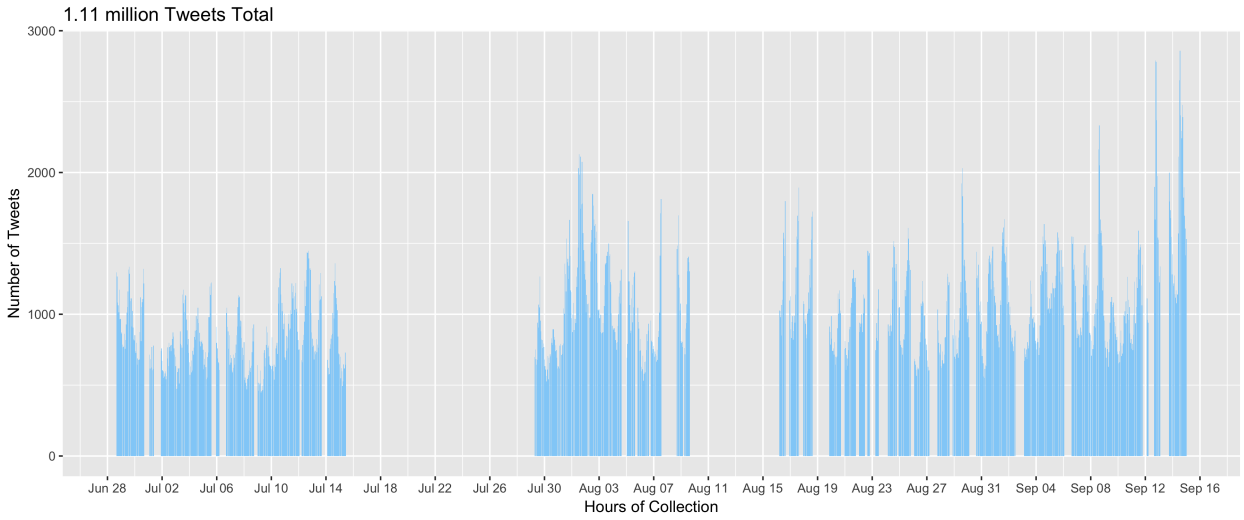


Figure 1: Number of tweets collected containing token *bitcoin* per hour from July 14, 2017 - Sept 15, 2017.

Next, a histogram of hourly returns was created to visualize the distribution. Figure 2 displays this distribution, which has a appears to be a fat-tailed distribution, which is characterized by the occurrence of low probability events greater than five standard distributions from the mean. The maximum return, 15.35%, was more than eight standard deviations away from the mean.

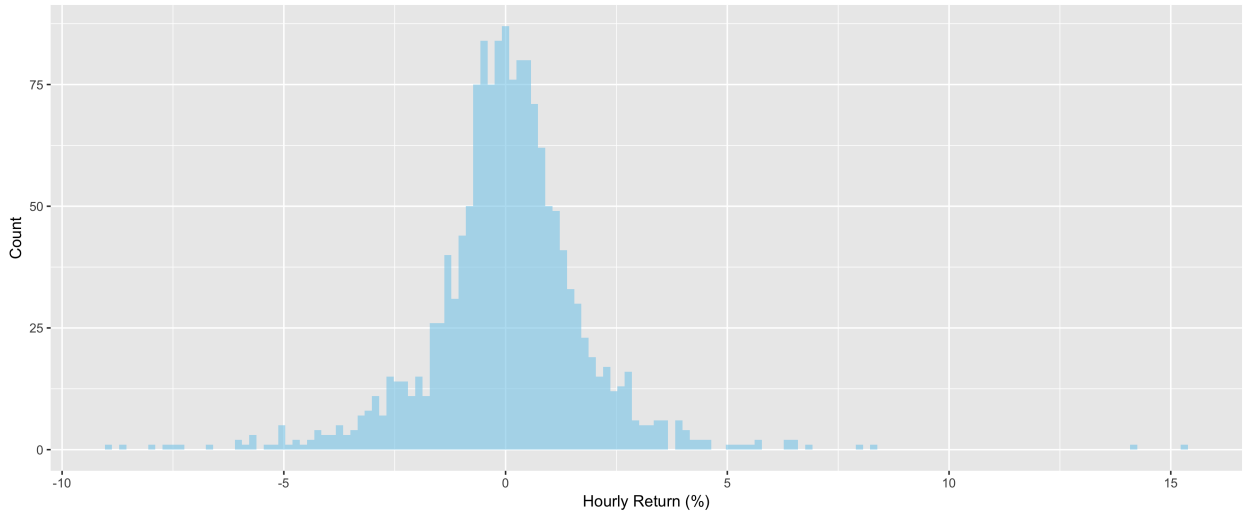


Figure 2: Distribution of hourly returns for Bitcoin from July 14, 2017 - Sept 15, 2017.

To further investigate the distribution of hourly returns, an empirical cumulative distribution function (ECDF) and a quantile-quantile (Q-Q) plot were generated. The ECDF plot allows for comparison to other well known CDF's while the Q-Q plot can be used to further compare observed returns to a theoretical normal distribution of returns. Both plots in Figure 3 show a clear skew from normal distribution, with the Q-Q plot emphasizing the extreme tails of the observed distribution compared to standard normal.

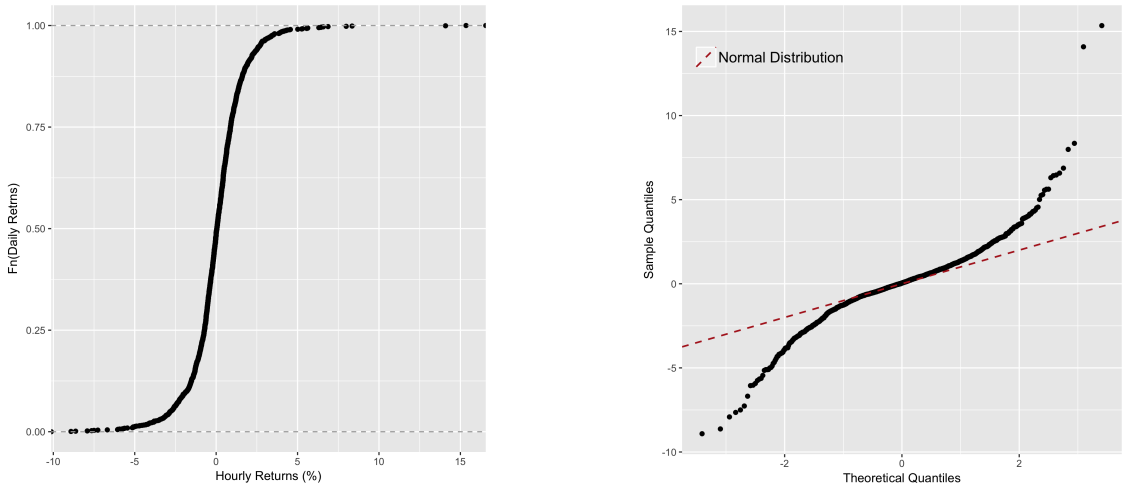


Figure 3: ECDF (left) and Q-Q normal plot (right) for hourly returns of Bitcoin from July 14, 2017 - Sept 15, 2017.

The correlation between between hourly returns was investigated to identify if price movements persisted from one hour to the next. The negative correlation of -0.0488 for successive hourly returns presented in Figure 4 indicates a negligiable tendency for a rise in the value of Bitcoin to be followed by a price fall and vice versa, but the the two are effectively uncorrelated. This further strengthens the random walk argument presented by Fama [2].

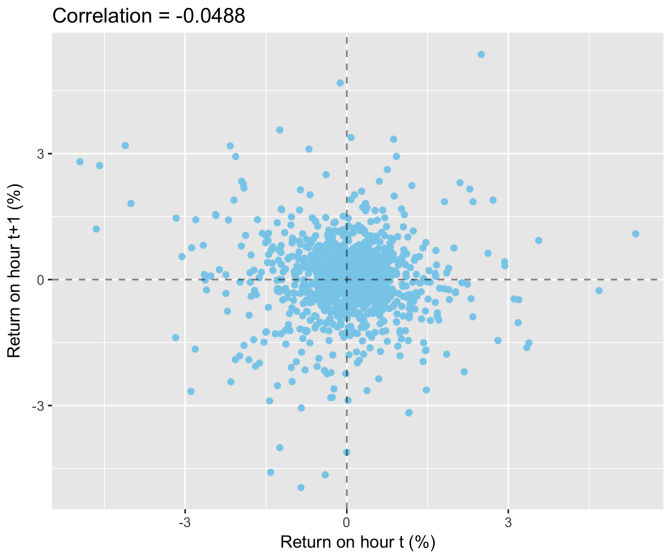


Figure 4: Scatter diagram for two successive hourly returns for Bitcoin from July 14, 2017 - Sept 15, 2017.

Examining Figure 1, it can be observed that the number of tweets collected has a cyclical nature, suggesting that the number of tweets varies for different hours of the day. Further inspection reveals that this is true not only for hour of the day but also day of the week. Figure 5 displays the average number of tweets collected for each hour of each weekday during collection. The number tweets recorded increases during afternoon hours in the United States, and a noticeable decrease in tweets is observed on weekends. The respective average value for each hour and weekday was used to normalize the raw tweet count of each hour, such that variation in tweets compared to the norm could be implemented as a feature variable into the ANN.

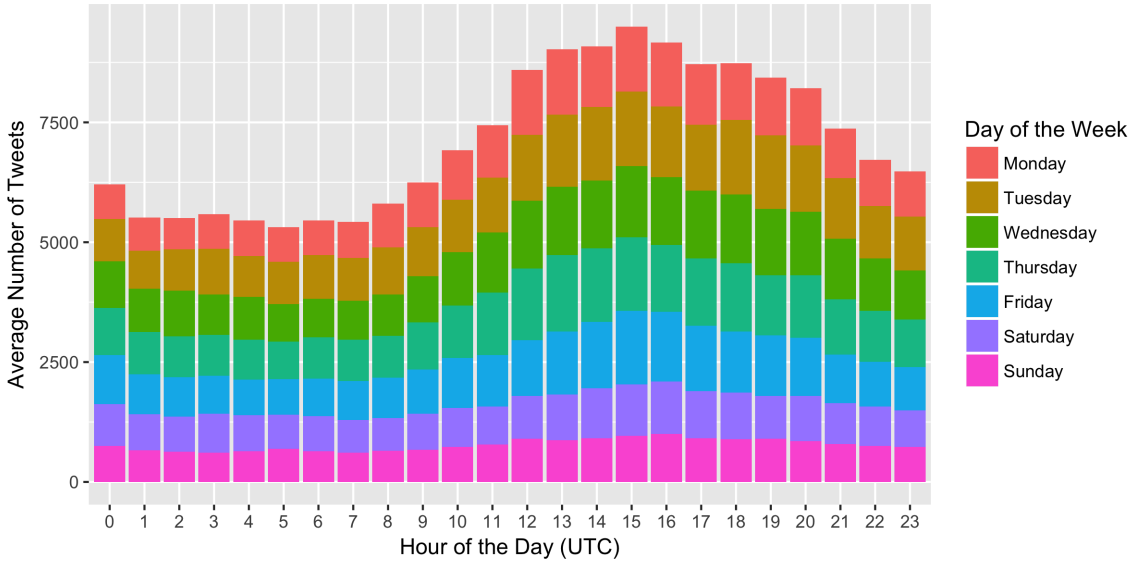
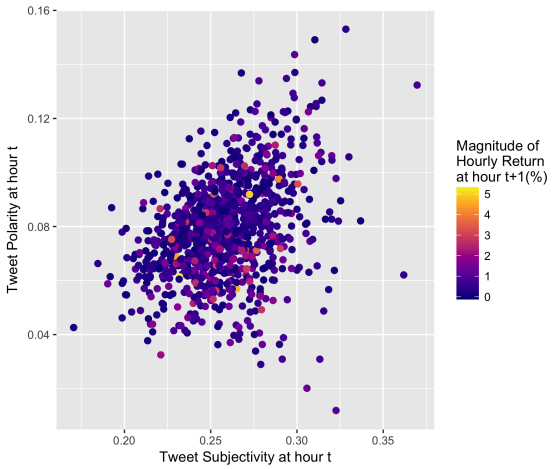
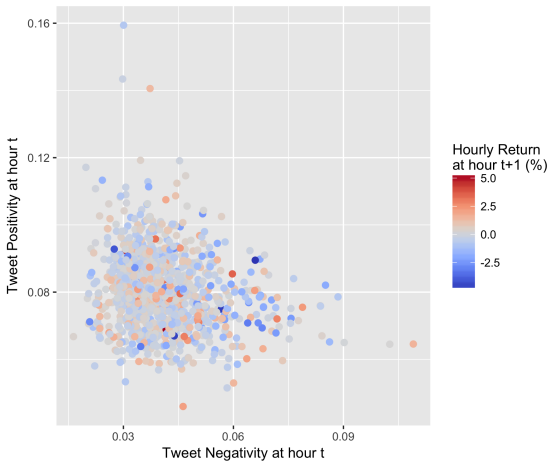


Figure 5: Average number of tweets collected containing token, *bitcoin* for each hour and weekday from July 14, 2017 - Sept 15, 2017.

Finally, the sentiment analysis results were compared to hourly returns. The average subjectivity and polarity of the tweets collected for each hour were compared to the magnitude of hourly returns, as neither metric was expected to help predict direction of movement, but may predict size of movement. Figure 6a suggests that polarity and subjectivity alone are not enough to predict magnitude, as no clear pattern is observed relating the three variables. The average positivity and negativity of tweets collected were compared to net return with the hypothesis that hours with strongly negative tweets would be followed by reduction in the value of Bitcoin, and strongly positive hours would result in a rise in value. However, as Figure 6b illustrates, the relationship between these variables is much more complex. Counterintuitively, the most positive hour was followed by a value reduction and the most negative hour was followed by a value increase, exactly the opposite of what had been hypothesized.



(a) Average polarity and subjectivity compared to the magnitude of the successive hour's return.



(b) Average positivity and negativity compared to the successive hour's return.

Figure 6: Hourly Bitcoin return as a function of sentiment analysis of Twitter.

2.3 Algorithms and Techniques

The proposed solution to this problem was a deep learning architecture. Due to the perceived non-linearity in the majority of previously studied social media causal relations with financial securities [8], a deep learning neural network was implemented. Artificial neural networks (ANN) offer a number of advantages over other machine learning models including the ability to implicitly detect complex nonlinear relationships between dependent and independent variables and the ability to investigate all possible interactions between predictor variables [9]. Specifically, a recurrent neural network (RNN) was selected to be applied in this study. Unlike the feedforward technique utilized by traditional ANNs, the connections between units in an RNN form a directed cycle, allowing for dynamic temporal behavior, which is crucial for time series data such as the studied Bitcoin returns.

2.4 Benchmark

Two benchmark models will be used for comparison with the developed algorithm using profit as the testing metric. First, a simple "buy and hold" strategy will be implemented such that the starting portfolio value will be used to purchase Bitcoin on the first day of the test period, and

the full portfolio value will be sold on the last day. This metric will compare the profits from actively trading Bitcoin using the developed algorithm with a simple value investing technique. This comparison reduces the impact of external factors such as political and macroeconomic events as both models will be impacted in an identical way. The second benchmark model will be investing the starting portfolio value in the S&P 500, an index of the 500 largest companies with common stock listed in the NYSE or NASDAQ, considered the best representation of the US stock market. This benchmark will compare the profits of trading Bitcoin to simply investing in the market, one of the most common and simplest investing strategies.

3 Methodology

3.1 Data Preprocessing

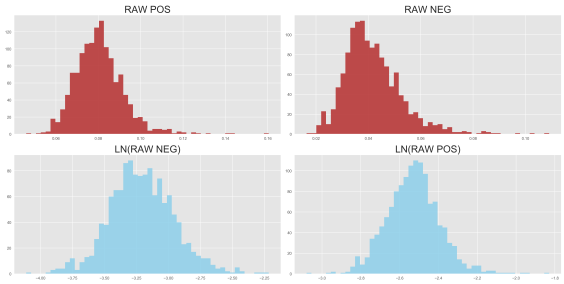
Before performing sentiment analysis, the raw tweets collected were cleaned. First the *unique ID* was used to delete all duplicate posts and "re-tweets," a Twitter mechanism to share an original post from another user. Next, tweets containing the phrase "RT @," a common tag used to copy and re-post other users' tweets similar to a re-tweet, were removed. The resulting unique, raw tweets were saved and a duplicate set was created to be cleaned further. For the next step of cleaning, all symbols were removed and all characters were made lower case. Both datasets were run through sentiment analysis, and the correlation of the raw and clean sentiments with successive hour's returns were compared. The clean neutrality and subjectivity sentiments had stronger correlation with return than their raw counterparts, and the raw positivity, negativity, and polarity sentiments had higher correlation than their clean counterparts, thus they were included as features vectors input into the ANN architecture.

An additional sentiment feature, sentiment bullishness index (SBI), introduced by Devi and Bhaskaran, was calculated using the raw positivity and raw negativity scores for individual tweets [10]:

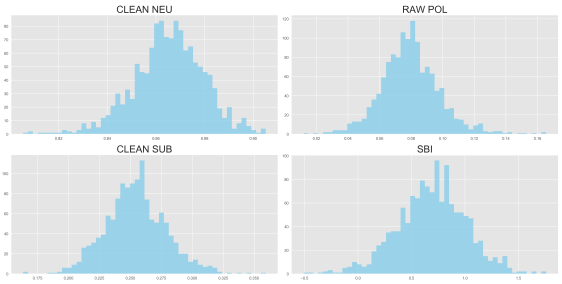
$$SBI_t = \ln\left(\frac{1 + N_t^{Pos}}{1 + N_t^{Neg}}\right) \tag{1}$$

where N_t^{Pos} is number of tweets in hour t for which the raw positivity is greater than the raw negativity, and N_t^{Neg} is the number of tweets which the raw negativity is greater than the raw positivity. This index is bullish for positive values, neutral for 0, and bearish for negative values.

Histograms of sentiments with the highest correlation were then plotted to view their respective distributions. Both raw positivity and raw negativity had a left skew, shown in Figure 7a *red*, so the natural logarithm of each feature was calculated, centering the distribution (Figure 7a, *blue*). The distributions shown in Figure 7b were considered to be sufficiently normal that they did not require further processing.



(a) Histograms of positivity and negativity (*red*) before and (*blue*) after transformation.



(b) Histograms of sentiment features which did not require transformation.

Figure 7: Histograms of sentiment features collected from Tweets containing token, *bitcoin* from July 14, 2017 - Sept 15, 2017.

Next, the hourly OHLCV data for Bitcoin and hourly returns were used to calculate common technical indicators used for stock and forex predictions. Technical indicators are frequently used by traders to apply a statistical approach to determine buy and sell signals.

First, the simple moving average (SMA) and exponential moving average (EMA) will be defined, as they are often used in calculation of other indicators. An exponential moving average finds the average of previous values, applying a weight to each value that is exponentially decreasing as are further from the present time period. The first value of the EMA is calculated from a simple moving average. Simple moving average calculates the average of values over a moving window of time periods:

$$SMA(x_t, n) = \frac{1}{n}\sum_{t=1}^{n} x_{t-n} \tag{2}$$

$$i \tag{3}$$

where n is the number of time periods in the simple moving average, which was kept as a constant 10 hours for this study, and x is the value of the parameter of interest at time period t. Once the first EMA value is calculated, the remaining values are determined recursively using Equation 4 [11]:

$$EMA(x_t, n) = \begin{cases} SMA(x_0, 10) & t = 0 \\ \frac{2}{n+1}[x_t - EMA(x_{t-1}, n)] + EMA(x_{t-1}, n) & t > 0 \end{cases} \tag{4}$$

Similar to SMA, a moving standard deviation (MSD) can be defined as the population standard deviation over a moving window of time periods:

$$MSD(x_t, n) = \sqrt{\frac{\sum_{t=1}^{n}(x_{t-n} - \bar{x})^2}{n}} \tag{5}$$

where $\bar{x}$ is the mean of parameter of interest's values over the number of time periods n in the observed window. Bollinger Bands, bands used to measure the current price relative to volatility, combine SMA with the MSD. Two bands are caluclated, an upper Bollinger Band, $upperBB$, and a lower band, $lowerBB$, given by Equation 6 and Equation 7 respectively [11]. Recently, Bollinger introduced a new indicator derived from his original Bollinger Bands, called $\%b$, which simplifies the traditional visual charting technique to one value. This value is negative when current prices

are below the lower band, 0.5 when current price is equal to the n period SMA, and greater than 1 when the current price is above the upper band. $\%b$ is calculated using Equation 8 [12]:

$$upperBB_t(n) = SMA(C_t, n) + 2MSD(C_t, n) \tag{6}$$

$$lowerBB_t(n) = SMA(C_t, n) - 2MSD(C_t, n) \tag{7}$$

$$\%b_t(n) = \frac{C_t - lowerBB_t(n)}{upperBB_t(n) - lowerBB_t(n)} \tag{8}$$

Another indicator calculated using both SMA and MSD is the commodity channel index (CCI). CCI was developed to compare current price with recent oscillations, and is frequently used for trading stock index futures and options. Values greater than 100 are considered overbought, while values below -100 are considered oversold. CCI was calculated using Equations 9 - 11 [11]:

$$TP_t = \frac{H_t + L_t + C_t}{3} \tag{9}$$

$$MD_t(n) = \frac{1}{n}\sum_{t=1}^{n}\Big|TP_{t-n} - SMA(TP_{t-n}, n)\Big| \tag{10}$$

$$CCI_t(n) = \frac{TP_t - SMA(TP_t, n)}{0.15MD_t(n)} \tag{11}$$

where TP is the typical price, H, L, and C are the respective high, low, and close values at time t, and MD is the mean deviation over the past n periods.

The momentum of Bitcoin was measured using the moving average convergence-divergence oscillator (MACD). MACD combines two moving averages into a momentum oscillator by subtracting the longer moving average from the shorter, providing both trend following and momentum signals in one indicator. Two typical periods used for MACD are (12, 26, 9) and (5, 35, 5), with the latter providing more sensitivity. Both sets of values were included as feature variables for this study, calculated using Equation 12 and Equation 13 [11]:

$$macd_t(n_1, n_2) = EMA(C_t, n_1) - EMA(C_t, n_2) \tag{12}$$

Equation 12 determines the *macd*, or the difference between two exponential moving averages of different period sizes. The exponential moving average of the *macd* is then subtracted from the current *macd* value in Equation 13 to obtain one value which can be used as an indicator, distinguished here using capital letters $MACD$. When the MACD is positive, the short period EMA is greater than the long period EMA, indicating upside momentum is increasing. When the MACD is negative, the short period EMA is less than the long period EMA, indicating that downside momentum is increasing.

$$MACD_t(n_1, n_2, n_3) = macd_t(n_1, n_2) - EMA\Big(macd_t(n_1, n_2),\ n_3\Big) \tag{13}$$

Next, the relative strength index (RSI) was calculated. RSI was developed to mitigate the effects of sharp changes in values on momentum indicators. For example, a sharp decline in value n

periods the past will cause a sharp increase in current momentum value for an indicator observing the past n periods, even if current price is relatively stable. To counteract this, RSI calculates the average of periods with both up closes (positive return) and down closes (negative return), taking the ratio as the relative strength, RS [11]:

$$RS_t(n) = -\frac{\sum_{t=1}^{n} r_{t-n}^{+}}{\sum_{t=1}^{n} r_{t-n}^{-}} \tag{14}$$

$$r_t^{+} = \begin{cases} C_t - C_{t-1} & C_t > C_{t-1} \\ 0 & otherwise \end{cases} \tag{15}$$

$$r_t^{-} = \begin{cases} C_t - C_{t-1} & C_t < C_{t-1} \\ 0 & otherwise \end{cases} \tag{16}$$

where r_t^{+} is the sum of gains and r_t^{-} is the sum of losses over the last n periods. The relative strength is then turned into an index using equation 17 [11]. Values range between 0-100, with values greater than 70 considered overbought and oversold below 30.

$$RSI_t(n) = 100 - \frac{100}{1 + RS_t(n)} \tag{17}$$

Another commonly used momentum indicator is Williams %R. Williams %R reflects the level of latest close in relation to the highest high and lowest low over the past n periods. Values have traditionally been bound to a range between -100 and 0 for charting in order to overlay the indicator onto plots of current price in a visually effective manner. Using this range, values below -80 are considered oversold and values greater than -20 are considered overbought. Equation 18 preserves this standard notation [11]:

$$\%R_t(n) = 100 \frac{\max\limits_{H_{t-n}, \dots, H_t} - \quad C_t}{\max\limits_{H_{t-n}, \dots, H_t} - \min\limits_{L_{t-n}, \dots, L_t}} \tag{18}$$

Bitcoin is notorious for its extreme volatility. In order to utilize this volatility for predictions, the average true range (ATR) was calculated as a feature. ATR is an indicator which is commonly used to measure volatility in commodities, which are frequently more volatile than stocks. In order to calculate ATR, the true range, tr is first determined from the greatest distance from the current period's high to low, previous period's close to current period's high, or previous close to current low [11]:

$$TR_t = \max\left\{H_t - L_t,\ |H_t - C_{t-1}|,\ |L_t - C_{t-1}|\right\} \tag{19}$$

The true range can then be transformed to ATR using Equation 20. ATR reflects the degree of interest or disinterest in a move (i.e., strong moves in either direction are often accompanied by large ATR values). The initial 14 ATR values are calculated with a 14 period SMA of true range values, with remaining values determined recursively [11]:

$$ATR_t = \begin{cases} SMA(TR_t, 14) & t < 15 \\ \frac{13ATR_{t-1} + TR_t}{14} & t \geq 15 \end{cases} \tag{20}$$

In order to incorporate volume into the model prediction, on balance volume (OBV) was added as a technical indicator. OBV measures buying and selling pressure as a cumulative indicator,

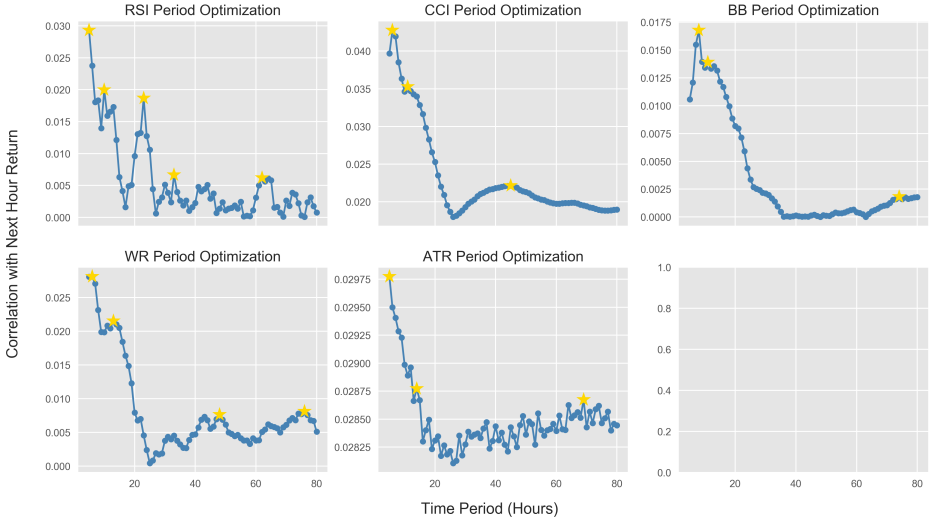
adding or subtracting the current volume in alignment with the sign of the return for the period. In order to turn OBV into a technical indicator, a short period SMA of cumulative volume, V^* is subtracted by a long period SMA of cumulative volume, [11]:

$$V_t^* = V_{t-1}^* + sgn(C_t - C_{t-1}) \times V_t \tag{21}$$

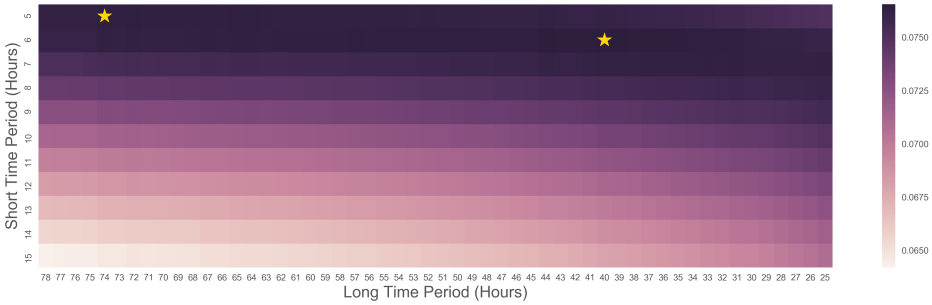
where V_t is the volume of trades in time period t. OBV is then calculated using Equation 22 [11]:

$$OBV_t(n_1, n_2) = EMA(V_t^*, n_1) - EMA(V_t^*, n_2) \tag{22}$$

In order to determine optimal time periods for technical indicators, the correlation of each indicator with the successive hour's return was calculated and plotted as a function of length of time period, n, used for each indicator. Each was calcualted a time period ranging from the oldest hourly OHLCV Bitcoin data available through the end of the training period, such that correlation information for the validation and testing sets did not cause leakage into the selection of indicators. This period ranged from February 25, 2017 through August 30, 2017. Figure 8 displays the results of this optimization approach.



(a) Indicators with one input time period.



(b) OBV optimization with both a long and short time period.

Figure 8: Optimization of time periods for technical indicators. Gold stars represent time periods selected as input features for ANN model.

Multiple time periods were selected for each indicator in order to include the maximum amount of possible variance into the model. In general, shorter periods increase sensitivity to current trends

while longer periods allow long term trends to monitored. For example, a long period Williams %R gauges long term trends in XBT, and periods of up to six months are commonly used on less volatile securities. In contrast, short term periods such as five hours provide a more instantaneous momentum indicator, capable of detecting sharp changes in value. By including a range of time periods for which correlation with future returns are at a peak, it is assumed that each peak represents a unique, relevant trend which will provide supplementary knowledge to the model.

Following the selection of technical indicator time periods, histograms of each indicator were created to visualize distributions. Figure 9 displays technical indicators which required transformation to reduce skewness.

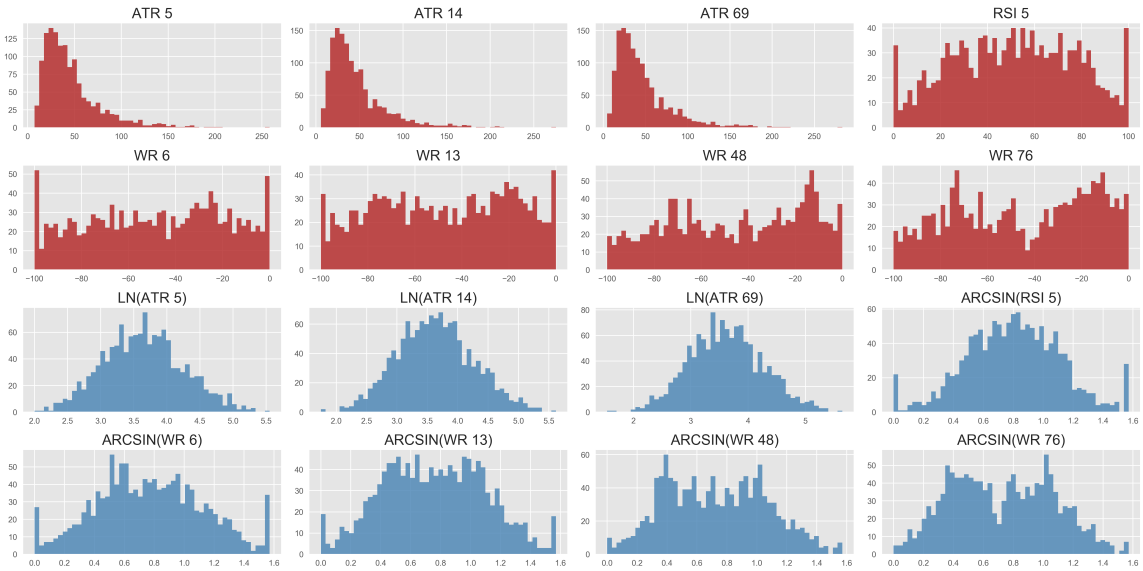


Figure 9: Histograms of technical indicators (*red*) before and (*blue*) after feature transformation.

ATR values were transformed using the natural logarithm to remove a left skew. RSI with a 5 hour period and Williams %R values were modified using an arcsine transformation. The arcsine transformation is used in multivariate studies with percentage data. Since RSI ranged from 0 to 100 and %R from -100 to 0, the values were first normalized by 100 and -100 respectively to put them in the correct 0 to 1 range of percentage data. Next the $arcsin\sqrt{x}$ was performed with x referring to the scaled values for either RSI or %R. Figure 9 shows the result of these transformations, reducing the skewness of the technical indicators, creating more normal, centered distributions to be input into the ANN model. The remaining technical indicators, with distributions show in Figure 10, were considered to be sufficiently normal that they did not require further processing.

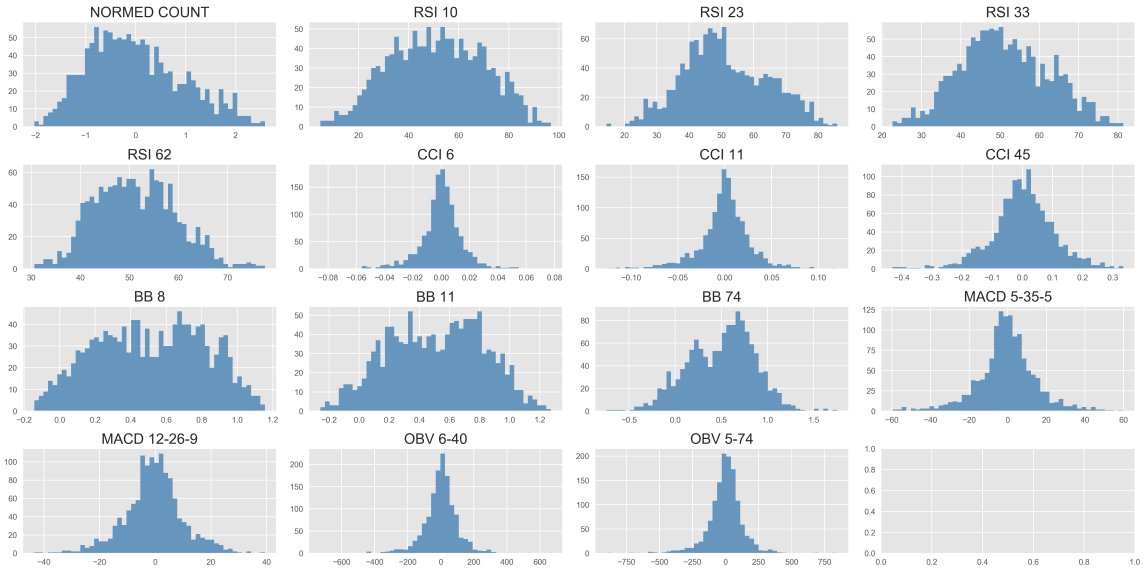


Figure 10: Histograms of technical indicators which did not require transformation.

Figure 11 displays a heat map of the correlation between each of the input features vectors for the ANN model. Many of the technical indicators were correlated with one another, and some of the sentiment indicators were correlated with each other as well. Interestingly, the normalized number of tweets each hour was moderately correlated (~0.5) with the ATR of Bitcoin, but ATR was not correlated with any of the other technical indicators. None of the indicators were strongly correlated with future returns.

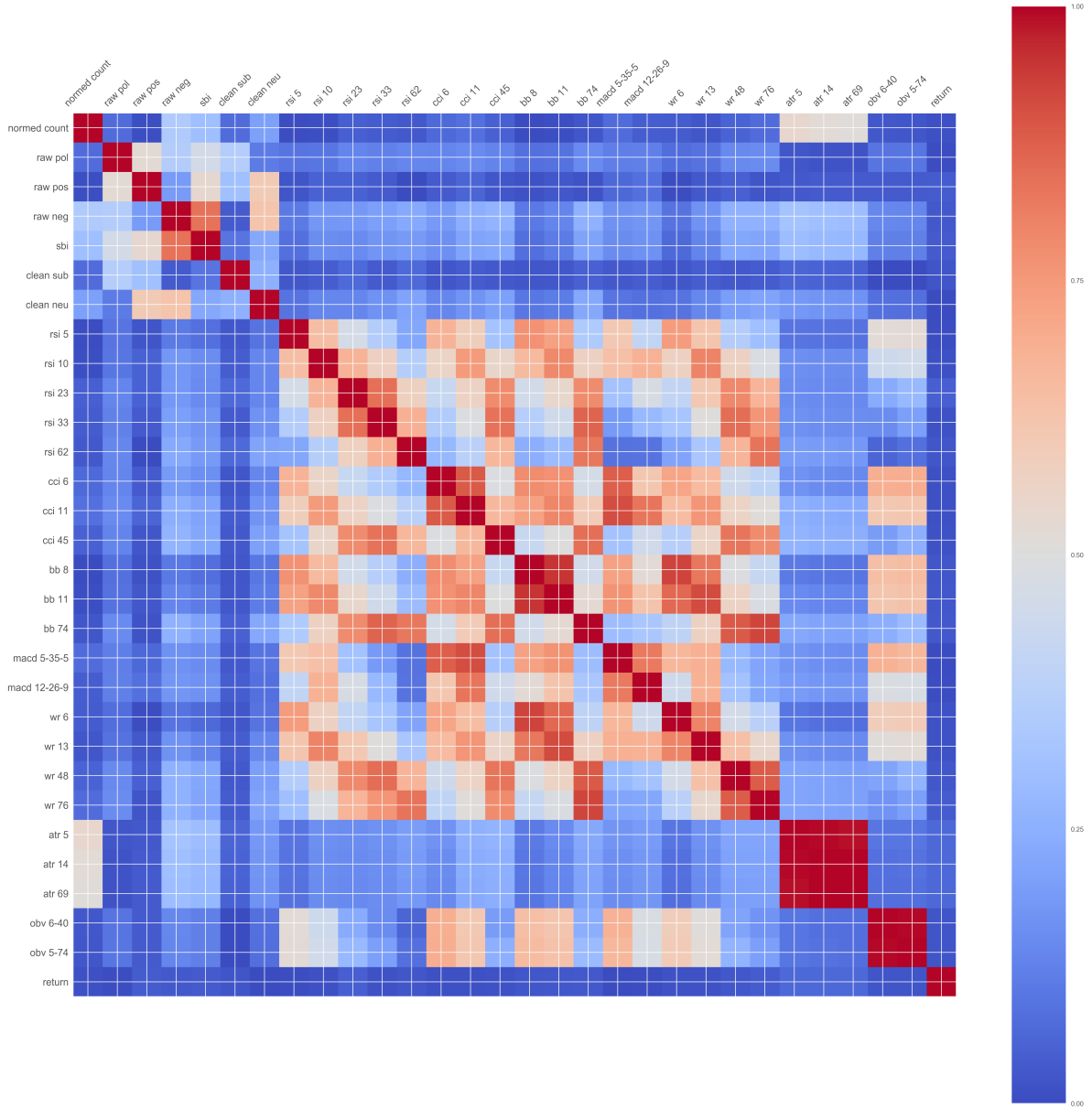


Figure 11: Heat map of correlations between input feature vectors.

The final preprocessing step was scaling and centering the input features. Each feature was standardized using Equation 23:

$$x' = \frac{x - \bar{x}}{\sigma_x} \tag{23}$$

where x' is the standardized feature, x is feature before standardization, $\bar{x}$ is the average of the feature, and σ_x is the standard deviation of the feature. Centering and scaling each feature forces the mean to be 0 with most of the data in the range -1 to 1 and nearly all data between -3 and 3. This allows features of different scales and sizes to be compared to one another, and decreases the training time of ANNs.

3.2 Implementation

Initially, an RNN was to be implemented as the architecture for this study, as the unique ability to preserve and incorporate time series information in predictions provided an optimal model for the given dataset. However, due to gaps in data collection, shown in Figure 1, an RNN was no longer suitable. RNNs require continuous time series inputs, where each successive time step is equidistant from the previous step, such that there are no breaks in the data. As previously mentioned, the necessity to live stream and scrape from Twitter and the associated limitations did not provide a perfectly continuous dataset of Tweets.

In order to overcome this complication, a conventional ANN was implemented as the model architecture for this study. A conventional ANN was selected over other machine learning architectures due to its ability to implicitly detect complex nonlinear relationships between dependent and independent variables and the ability to investigate all possible interactions between predictor variables [9].

During the time period studied, 748 hours had positive returns while 741 hours had negative returns. Given the nearly perfectly balanced outcomes, simple accuracy was chosen as the evaluation metric for the ANN model during refinement. The dataset was split into three chronological groups, with the oldest 72% used for training the architecture, the next 13% used for cross-validation of the model, and the most recently collected 15% saved for testing.

In order to draw stronger conclusions about the usefulness and practicality of the model however, a market simulator was designed to simulate the performance of the algorithm in real life. The testing dataset was not seen by the model until it was used for predictions in the market simulator, such that no information was able to leak into the model and the simulation would accurately represent an algorithms application to the market.

The market simulator was modeled using Kraken Bitcoin Exchange [13]. The fee for each trade was calculated on a per-trade basis as a percentage of the size of each order in USD. The fee percentage on Kraken is not stagnant, but a function of trades made by each account over the past 30 days. The fee percentages used during the simulation are as follows: [13].

Table 1: Simulated transaction fees for orders on Kraken Bitcoin Exchange.

30 Day Volume (USD)	Fee	Slippage	Total Transaction Cost
$< 50{,}000$	0.16%	0.10%	0.26%
$< 100{,}000$	0.14%	0.10%	0.24%
$< 250{,}000$	0.12%	0.10%	0.22%
$< 500{,}000$	0.10%	0.10%	0.20%
$< 1{,}000{,}000$	0.08%	0.10%	0.18%

Additionally, slippage of 10 basis points (10 bp or 0.10%) was added to the transaction cost for each order placed. Slippage refers to the price difference between where a market trade is placed and where it is executed. When trades are executed, the exchange is made at the most favorable price available. For a trade of \$10,000 placed when the XBT/USD exchange rate is \$3910.0, the first \$5000 may trade at the listed exchange rate, but all book orders at that rate have been filled, the next most favorable exchange rate may be \$3918.0, which is the rate at which the remaining \$5000 would be exchanged. In this example, there was a 20 bp change in exchange rate over the duration

of the trade, with the average price paid being ~10 bp from the listed value. Since price moves up when buying and down when selling, this form of slippage always works against the trader. Another possible source of slippage comes from another trader making a large move between the time an order is placed and executed, which could affect the price in either direction. While it is impossible to perfectly replicate this phenomenon in a simulator, the value of 10 bp was chosen as a conservatively large value from monitoring the bid-ask spread on Kraken Bitcoin Exchange, specifically noting movement resulting from orders of ~$10,000. Summing the fee from Kraken Bitcoin Exchange with the average assumed slippage yields the total transaction cost, shown in Table 1.

3.3 Refinement

Initially, a simple network of five layers with $\{5, 20, 20, 5, 1\}$ units in each respective layer was implemented, using the hyperbolic tangent activation function (Equation 24) for the hidden layers and the logistic sigmoid function (Equation 25) for the output node.

$$tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{24}$$

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{25}$$

The hyperbolic tangent function was chosen as it allows for values to range between -1 and 1, matching the scaled and centered data input into the network. The sigmoid function was selected for the output layer as it scales the final output to a value between 0 and 1, providing the probability that the input features result in a classification of factor 1. Using the sigmoid output, values greater than 0.5 are predicted to be 1 (*buy*) and values below 0.5 are predicted to be 0 (*sell*).

The initial architecture provided a prediction equivalent to guessing, with ~50.3% training accuracy. Other activation functions including rectified linear unit (ReLU), scaled exponential linear unit (SELU), and S-shaped rectified linear activation unit (SReLU) were tested in place of the hyperbolic tangent, with each performing poorly in comparison. Additional units were subsequently added to each layer in an attempt to increase training accuracy. Moderate improvements in training accuracy were achieved through the addition of units. Next, more layers were added to the network to further increase training accuracy. The number of layers and units was adjusted until 80% training accuracy was achieved, with a network of seven layers, $\{64, 64, 128, 256, 256, 256, 1\}$.

Next, batch normalization was included at each layer to increase learning rate, boosting the speed of training. Similar to the final preprocessing step before inputting features into the ANN, batch normalization normalizes and centers the weights and parameters at each layer of the network, avoiding complications that may arise from values diverging from 0. Further, batch normalization increased both the testing accuracy of the network. The validation accuracy at this stage was ~40%, indicating strong overfitting of the training data.

In order to overcome overfitting, dropout regularization was applied to the network. Dropout regularization randomly selects a percentage of neurons to ignore from the specified layer during training, so contributions to the activation of downstream neurons is temporarily interrupted on the feed forward step, and weight updates are not applied to the neurons during the backwards pass. This increases co-dependence between neurons such that the network as a whole is less sensitive to specific weights of individual neurons, resulting in multiple independent internal representations

being learned. This in turn improves the ability of the network to generalize and mitigates the probability of overfitting. Initially, dropout rates of 0.2 (20%) were applied to each layer, as this is the default recommended value by Keras [14]. Dropout rates were experimented with on each layer. Binary cross entropy loss was used as criterion for selecting an optimal model. The model which minimized validation loss had the following structure:

$$\{64,\ ^*64,\ (0.2)^*128,\ (0.3)^*256,\ (0.4)^*256,\ (0.4)^*256,\ ^*1\}$$

where the dropout rates are in () and Batch normalization is represented with *, with hyperbolic tangent activation function for hidden layers and sigmoid activation for the output layer. 200 epochs were used to determine the optimal weights for the network. The final training accuracy achieved was 57.3% with an equal testing accuracy of 57.3%, indicating no overfitting and a model capable of generalizing.

4 Results

4.1 Model Evaluation and Validation

The final model used for this study was an artificial neural network, chosen for its ability to find underlying, non-linear connections between input features. The final model architecture consisted of seven layers with batch normalization applied at each to improve accuracy and training time, and dropout regularization utilized at four of the hidden layers to mitigate overfitting. The input feature vectors consisted of two datasets. First, raw tweets containing the token, *bitcoin* were collected over a three month period. These raw tweets were cleaned, and analyzed using sentiment analysis. The results of this analysis were averaged into one hour batches to be used as feature vectors, including normalized counts per hour, positivity, negativity, neutrality, subjectivity, polarity, and constructed feature sentiment bullishness index. The second data set consisted of hourly OHLCV data for Bitcoin over an identical period. Technical indicators commonly used for evaluation of securities were applied to this set to form feature vectors, including Bollinger bands, commodity channel index, moving average convergence-divergence, relative strength index, Williams %R, average true range, and on balance volume. Multiple time periods were applied for each indicator to maximize variance in the model, yielding twenty-two technical indicators, and a total of twenty-nine feature vectors for the ANN architecture. Before introduction into the model, features were transformed to reduce skewness and improve homoscedasticity, centered to zero, and normalized by their respective standard deviations, such that each feature was approximately standard normal.

The final model architecture selected was chosen after rigorous optimization of model parameters including activation function, number of hidden layers, number of neurons per layer, use of normalization, use of regularization, and regularization dropout rate. Binary cross entropy loss was applied as cross validation criterion for selecting an optimal model. The resulting architecture had 57.3% accuracy on on both the training and testing datasets, demonstrating the ability to generalize well to new data.

Next, the accuracy results from the test period were merged with the raw hourly Bitcoin data to compare hourly returns with prediction accuracy. Figure 12 shows the quartiles for both correct and incorrect predictions made by the algorithm during the test period. The average return for correct predictions (0.88% arithmetic, 0.52% geometric) was higher than the average return for incorrect predictions (0.79% arithmetic, 0.51% geometric). For returns with magnitude below 1%, the algorithm had prediction accuracy of 55%, which increased to 66% for returns between 1-3%.

For returns with magnitude greater than 3%, the algorithm correctly predicted 33% of the periods, but more data is needed in each regime to draw stronger conclusions.

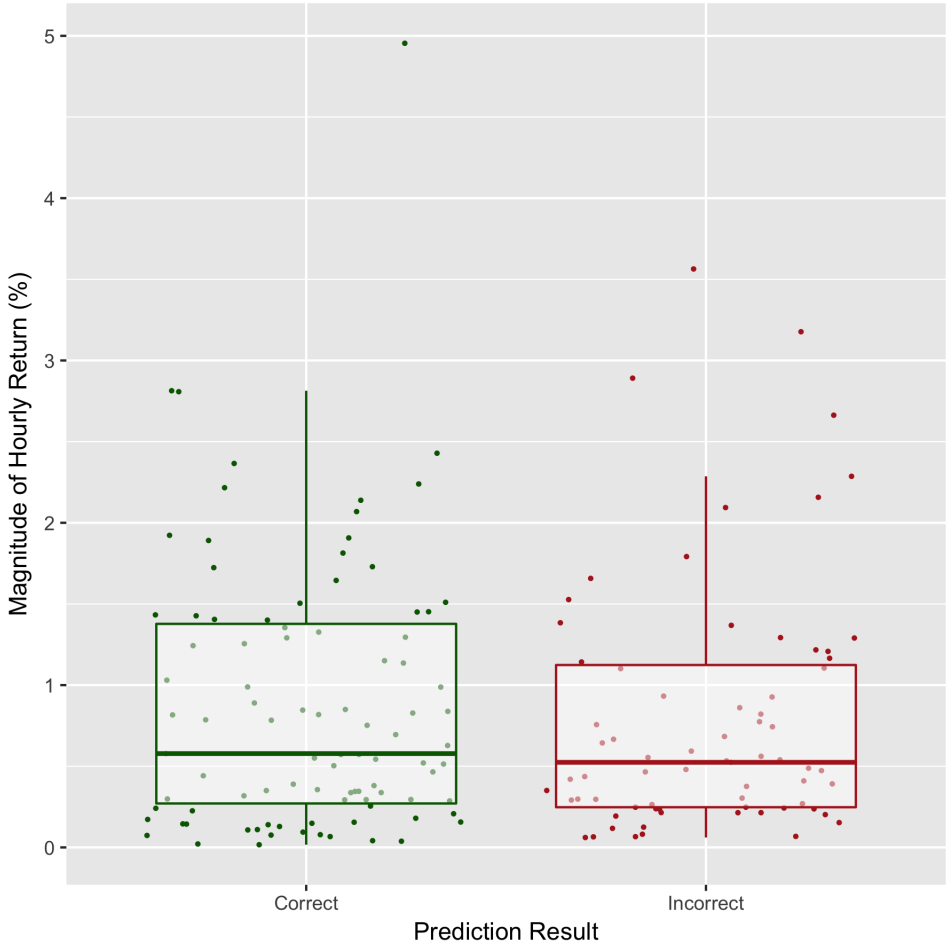


Figure 12: Box plot depicting quartiles and mean magnitude of predictions made by ANN algorithm over test period from Sept 9, 2017 - Sept 15, 2017.

Additionally, a market simulator was created to approximate the outcome of applying the developed algorithm in the real market, accounting for transaction costs and estimated slippage. Figures 13 and 14 display the results of the ANN algorithm trading XBT compared to a portfolio consisting of buying and holding XBT and a portfolio consisting of the S&P 500.

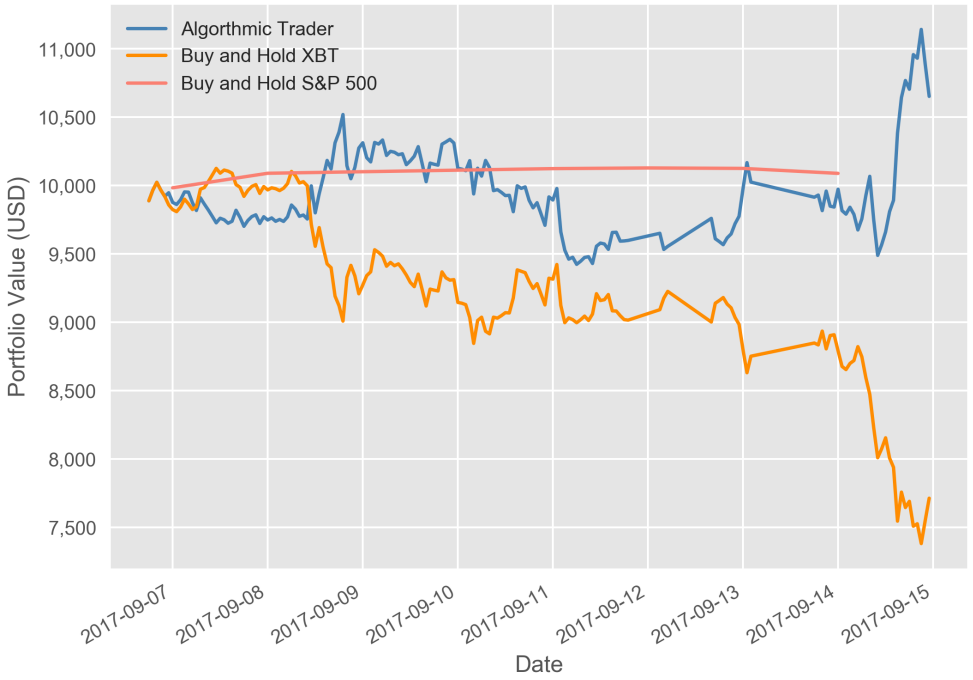


Figure 13: Comparison of ANN algorithm trading XBT, buying and holding XBT, and buying and holding the S&P 500 over test period of the dataset from Sept 9, 2017 - Sept 15, 2017. XBT values are hourly close, while S&P 500 values are daily close.

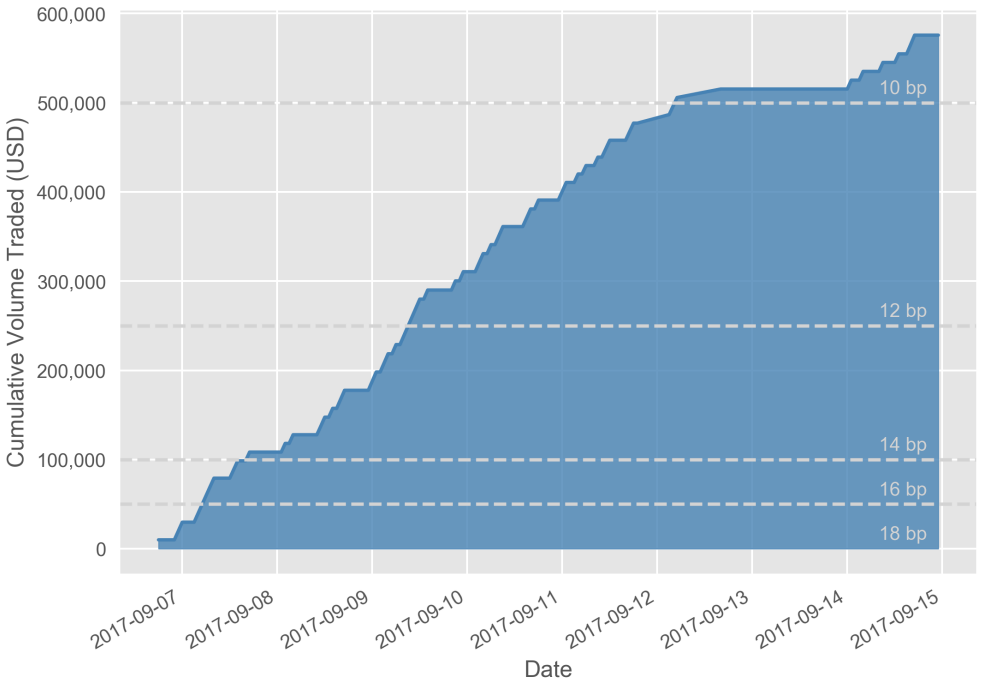


Figure 14: Cumulative volume traded during market simulation. Dashed lines indicate ranges where trading fees are reduced.

Bitcoin values were the hourly closing values from Kraken Bitcoin Exchange obtained from Bitcoin Charts [7], while S&P 500 values were daily closing values obtain using Pandas Datareader API to pull Yahoo! Finance data[15, 16]. Standard and Poor's Depositary Receipt's S&P 500 Trust ETF, (ticker 'SPY'), was used to approximate S&P 500 value. only daily values of SPY are freely available, limiting the visual comparison to Bitcoin over the testing period. Bitcoin's immense volatility is evident in the eight day period, with hourly changes in value peaking at 5%. For comparison, the S&P 500's largest change in value occurred on Black Monday in 1987 when its value fell 22.6% over the course of the entire market day.

4.2 Justification

From Figure 13 it can be seen that the algorithm performed much better than simply buying and holding Bitcoin, and was more profitable than holding the S&P 500, but still suffered from sever volatility. Multiple metrics have been developed to combine these factors using portfolio analysis, a systematic method for relating risk and reward in investments. The most common metric is the Sharpe ratio, which measurers risk adjusted return compared to the risk-free rate, or theoretical rate of return on an investment with zero risk. The Sharpe ratio is given by Equation 26 [17]:

$$Sharpe\ Ratio = \frac{\bar{r_p} - r_f}{\sigma_p} \tag{26}$$

where $\bar{r_p}$ is the mean return of the portfolio, r_f is the risk-free rate over an identical period, and σ_p is the standard deviation of returns. The risk-free rate used in this study was the yield on a ten-year US treasury note, interpolated for the length of the study.

One criticism of the Sharpe ratio is that it punishes volatility in both directions, even though volatility that results in positive returns is welcomed by investors. In order to account for this, the Sortino ratio was developed. The Sortino ratio is similar to the Sharpe ratio, but uses only downside deviation in the denominator [17]:

$$Sortino\ Ratio = \frac{\bar{r_p} - r_f}{\sigma_d} \tag{27}$$

where the downside deviation, σ_d, is given by:

$$\sigma_d = \sqrt{\frac{1}{n} \sum_{r_i < r_f}^{n} (r_i - r_f)^2} \tag{28}$$

Both the Sharpe and Sortino ratios were calculated for each tested portfolio in the study, with results presented in Table 2. Negative values for both ratios for Bitcoin and the algorithm indicate that the return on the investment is not enough to justify the risk of the portfolio. Values for both metrics are considered good at 1, very good at 2, and great at 3. The S&P 500 values should be near 1 in both cases, showing that the sample size used in this study is not sufficient to draw definitive conclusions about any of the three portfolios studied. In order to draw such conclusions, the length of the study would likely need to increase tenfold, and data collection should be continuous, without gaps.

Table 2: Portfolio Analysis for ANN algorithm, buying and holding Bitcoin, and buying and holding S&P 500.

Portfolio Metric	Algorithm	Hold XBT	Hold S&P 500
Sharpe Ratio	-0.193	-1.269	1.714
Sortino Ratio	-0.006	-0.084	0.009

As a final sanity check, the study was repeated using logistic regression in place of a neural network. Logistic regression does not benefit from the detection of nonlinear relationships between input feature variables, and assumes that features are linearly independent, a claim shown to be false in Figure 11. Hyperparameter optimization of the model was performed for regularization strength and the solving algorithm. Three solvers were tested, limited-memory BFGS (lbfgs), liblinear, and newton conjugate gradient algorithm (newton-cg). The optimal model achieved a training accuracy of 56.6% and test accuracy of 49.2%. The optimal model performed essentially the same as simply guessing, demonstrating both the non-linearity between features and overall complexity of predicting Bitcoin value. Logistic regression's inferior performance validates the use of a complex model such as the ANN used in this study.

5 Conclusion

5.1 Free-Form Visualization

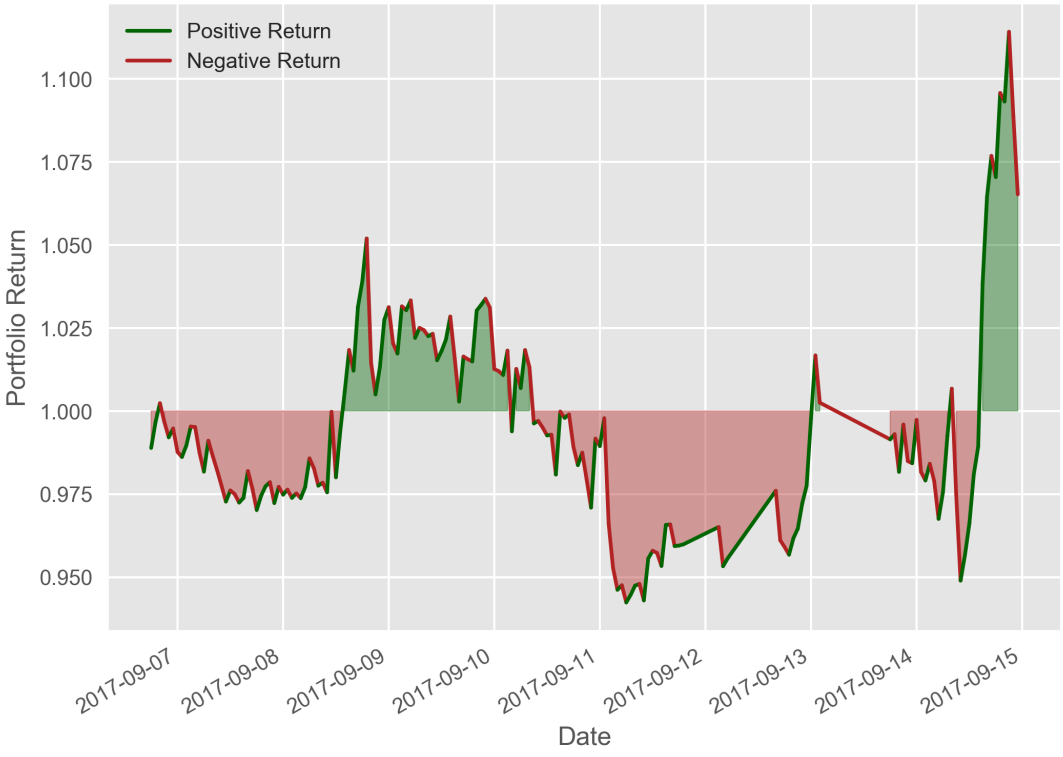


Figure 15: Return on portfolio over test period from Sept 9, 2017 - Sept 15, 2017.

Figure 15 displays the positive and negative returns earned by the ANN model during the market simulation, both of individual hourly trades and overall return since implementation to the market. The algorithm fluctuated between an overall positive and negative return nine times, ending with an overall positive return of 0.65% over eight days. This would extrapolate to 17.7% return over the course of one year, but the sample size is not large enough, nor representative of such a long time period, to make such claim with confidence. Although the test period is too small to elicit confidence, it does show promise. The cumulative volume traded continuously increases, so transaction fees, which are inversely proportional to cumulative volume traded over the past 30 days, would decrease substantially beyond the rates observed in the 8 day test period. In total \$366 or 0.36% of the starting portfolio value was lost to fees, which is over half of the total return earned. As future fees decrease, the hurdle to be profitable decreases and the algorithm's returns receive a non-negligible incremental hike. This indicates that an identical accuracy (for all return magnitudes) would result in superior performance in the future, with larger returns.

5.2 Reflection

This study consisted of three main segments, data collection, data preprocessing, and deep learning analysis. Each section contained its own unique set of challenges. The data collection process was complicated by the necessity to live stream data, causing gaps in collection due to extraneous factors. Data preprocessing was the most time intensive step. An in depth literature review was conducted to determine technical indicators suitable for Bitcoin before calculation and optimization procedures began. Several sentiment analysis techniques were investigated to extract the maximum information from raw tweets. The requirement of an advanced machine learning technique such as ANN used in the study added complexity to the determination of an optimal model. Unlike techniques such as logistic regression or random forests which have only a few hyperparameters to optimize allowing for a grid-search approach, neural networks have numerous parameters that are not capable of a simple grid-search approach. This requires intuition and experience on behalf of the architecture's designer and requires much more development time in addition to computation time.

One interesting aspect of this project was finding that a correlation between Twitter posts and the value of Bitcoin not only exists but was more predominant than some of the most commonly used technical indicators. This aligns with the hypothesis that many Bitcoin investors are tech-savvy and have social media presence, such that Twitter sentiment would be more significant in predicting Bitcoin's value than a Fiat currency or large cap stock. The ability of the ANN algorithm to perform better than randomly guessing matched my expectations and affirms the use of such an architecture in similar applications. It was expected that the neural network would perform better than guessing, as ANNs with only technical indicators are currently successfully employed in the market, but an accuracy 7% better than guessing surpassed expectations. The profitability of the developed algorithm during the testing period which accounted for trading fees and slippage was a pleasant surprise, given the historical complexity of predicting the value of financial securities. However, this profit was from only an eight day sample, much too small to extrapolate expected profits using the algorithm in the real market for an extended period of time.

5.3 Improvement

The two sections which improvement may have the largest impact on the final end goal, increased profit, are the data collection process and the ANN architecture. Twitter sells historical data, so full datasets for a time period spanning years can be created for a fee. This would exponentially

increase the sample sizes used in the study, improving the accuracy of the model and increasing the strength of conclusions that can be drawn as a result of the model. With a larger dataset from Twitter, additional, more complex sentiment analysis techniques could be explored and potentially included in the deep learning model. Further, a continuous, fully connected dataset would allow for the use of an RNN, as originally intended for this study. RNNs use a directed cycle architecture as opposed to the feedforward technique utilized by ANNs, allowing for dynamic temporal behavior, which may be crucial for optimal predictions for complex time series data. With a larger dataset and implementation of an RNN, the 57.3% accuracy achieved in this study can reasonably be expected to be surpassed. While this study forms a basic framework for using sentiment analysis to predict Bitcoin value, there is an immense amount of potential improvements which can be made to increase prediction capability and ultimately profit.

References

[1] Bollen, Johan, and Huina Mao. (2010) "Twitter Mood Predicts the Stock Market," *Journal of Computational Science*. 2(1), 1-8

[2] Fama, Eugene F. (1991) "Efficient Capital Markets: II," *The Journal of Finance*. 46(5):1575.

[3] Qian, Bo, and Khaled Rasheed. (2006) "Stock Market Prediction with Multiple Classifiers," *Applied Intelligence*. 26(1):25-33.

[4] *Tweepy*. `http://www.tweepy.org/` Web. Sept 16, 2017.

[5] *Vader*. `https://github.com/cjhutto/vaderSentiment` Web. Sept 16, 2017.

[6] *TextBlob* `https://textblob.readthedocs.io/en/dev/` Web. Sept 16, 2017.

[7] *Bitcoinchartts*. `https://bitcoincharts.com/charts/krakenUSD` Web. Sept 16, 2017.

[8] Tharsis T.P. Souza, Tomaso Aste. (2016) "A Nonlinear Impact: Evidences of Causal Effects of Social Media on Market Prices," UCL.

[9] Tu, J.V. (1996) "Advantages and Disadvantages of using Artificial Neural Networks versus Logistic Regression for Predicting Medical Outcomes," *J Clin Epidemiol*. 49(11):1225-31

[10] Devi, K.N. and Bhaskaran, V.M. (2015) "Semantic Enhanced Social Media Sentiments for Stock Market Prediction," *International Science Index, Economics and Management Engineering*. 9(2)

[11] Murphy, J.J. *Technical Analysis of the Financial Markets*; New York Institute of Finance: Paramus, New Jersey, 1999.

[12] *Bollinger Bands* `https://stockcharts.com` Web. Sept 22, 2017.

[13] *Kraken Bitcoin Exchange* `https://www.kraken.com` Web. Sept 26. 2017.

[14] *Keras* `https://github.com/fchollet/keras` Web. Sept 26, 2017.

[15] *Pandas Datareader* `https://github.com/pydata/pandas-datareader` Web. Sept 26, 2017.

[16] *Yahoo! Finance* `https://finance.yahoo.com` Web. Sept 16, 2017.

[17] *Investopedia* `http://www.investopedia.com` Web. Sept 16, 2017.