Bitcoin Price Prediction using Sentiment Analysis

Ketan M. Mehta
Electrical Engineering Dept.
Columbia University
kmm2304@columbia.edu

Chandra S. Narain Kappera
Computer Science Dept.
Columbia University
ck2840@columbia.edu

Venkata Sai Sriharsha Sammeta
Computer Science Dept.
Columbia University
vs2626@columbia.edu

Abstract— **In this paper we propose and implement a real-time bitcoin price prediction and alerting system. Our system is novel in two ways – firstly, it combines tweet and reddit sentiments as one of the input features and secondly, it incorporates a real-time bitcoin price prediction engine which sends out SMS alerts. We implemented two models – classic ARIMA model and LSTM. We achieved better results with LSTM model.**

Keywords- Bitcoin, Real-Time Price Prediction, Sequence-to-Sequence Model, Deep Neural Networks, LSTM, ARIMA, Real-time alerts, Tweets, Reddit.

I. Introduction

It’s 2017, where people of the world generate 2.5 million terabytes of information a day - 500 million tweets, 1.8 billion pieces of shared information on Facebook, each and every day - It’s an information revolution. Twitter specifically has become known as a location where news/opinions are expressed quickly in a concise format. It naturally makes sense to use twitter data for predictions of stock prices and cryptocurrencies, where consumer sentiment affects the prices. In our project we are using twitter data along with reddit data to predict bitcoin value.

Bitcoin, a decentralized electronic currency system, represents a radical change in financial systems after its creation in 2008 by Satoshi Nakamoto. Bitcoin stands for an IT innovation based on the advancement in peer-to-peer networks and cryptographic protocols. Due to its properties, Bitcoin is not managed by any governments or bank. Like any other currency, a peculiarity of Bitcoin is to facilitate transactions of services and goods, attracting a large number of users and a lot of media attention.

Since the Bitcoin was revealed to the world, in 2009, it quickly gained interest as an alternative to regular currencies. As such, like most things, opinions and information about Bitcoin are prevalent throughout the Social Media sphere.

Bitcoin prices are highly volatile and those cannot be accurately estimated by using standard economic theory since, the price of Bitcoins is mainly driven by the interaction of supply and demand fundamentals. In this context, the impact of mining technology which affects the production cost structure and thus supply side of the market on Bitcoin prices.

However, the supply of Bitcoins evolves according to a publicly known algorithm and the level of demand is not fully determined by the fundamentals of the underlying economy but also depends on expectations about future price movements. This is another reason why we should consider other sources of data for predicting bitcoin value.

In recent times bitcoin prices have risen from $8000 to $17000 within two months. Suddenly, this change created a buzz about bitcoin in social media, news channels, and financial blogs and so on. In this project, we want to capitalize on this sudden increase in bitcoin value by proposing business model by creating an automated bitcoin price prediction and trading system. Another advantage of trading bitcoin is that it has very high returns compared to conventional financial instruments. In simple words it requires minimum investment and maximum returns.

As explained above bitcoin prices are highly volatile and standard economy standards can’t predict its price so we are using sentiments of twitter and reddit data to predict bitcoin value.

In this project basically, we are capture real-time streaming data every minute from the twitter, reddit and coinbase, then send it to extract sentiment values and use it as an input feature along with bitcoin value to predict value for next minute. For getting more accurate predictions we are leveraging AI, by implementing ARIMA and LSTM deep learning models.

The structure of the remaining paper is as follows: section II talks about background and related work on this topic, Section III talks about the system we are using in detail including data collection and data preprocessing, Section IV discusses models we are using detail followed by their implementation in section V. In the end, results are discussed in section VI followed by the conclusion.

II. Related Works

Since the inception of bitcoin in 2009, people are interested in analyzing its value. But after increasing use of twitter and other social media people have done considerable work on finding relation between bitcoin value and social media data.

The Bitcoin represents an important new phenomenon in financial markets. Mai et al. [1] examine predictive relationships between social media and Bitcoin returns by considering the relative effect of different social media platforms and the dynamics of the resulting relationships using vector autoregressive and vector error correction models.

In the paper Trading on Twitter: Using Social Media Sentiment to Predict Stock Returns by Sul et al. [2], 2.5 million tweets about S&P 500 firms were put through the authors own sentiment classifier and compared to the stock returns. The results showed that sentiment that disseminates through a social network quickly is anticipated to be reflected in a stock price on the same trading day, while slower spreading sentiment is more likely to be reflected on future trading days. Basing a trading strategy on these predictions are prospected to yield 11-15% annual gains.

In their work, Garcia et al. [3] show the interdependence between social signals and price in the Bitcoin economy, namely a social feedback cycle based on word of mouth effect and a user-driven adoption cycle. They provide evidence that Bitcoin's growing popularity causes an increasing search volume, which in turn result a higher social media activity about Bitcoin. More interest inspires the purchase of bitcoins by users, driving the prices up, which eventually feeds back on the search volumes.

The paper Algorithmic Trading of Cryptocurrency Based on Twitter Sentiment Analysis by Colianni et al. [4], similarly analyzed how tweet sentiment could be used to impact investment decisions specifically on Bitcoin. The authors used supervised machine learning techniques that yielded a final accuracy of above 90% hour-by-hour and day by day. The authors point out that the 90% accuracy was mustered through robust error analysis on the input data, which on average yielded a 25% better accuracy. Colianni et al. together with Hutto and Gilbert both mentioned levels of noise in their dataset, and the former team got a significant reduction in error rates after cleaning their dataset for noise.

In our project we are using twitter and reddit data along with machine learning and deep learning models to predict value.

III. System Overview

This section talks about our architecture, data collection and data preprocessing.

1. Architecture

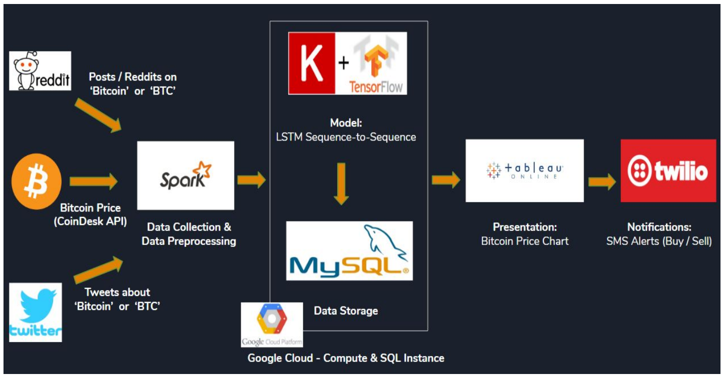

reddit
Posts / Reddits on 'Bitcoin' or 'BTC'
Bitcoin Price (CoinDesk API)
twitter
Tweets about 'Bitcoin' or 'BTC'
Spark
Data Collection & Data Preprocessing
K + TensorFlow
Model: LSTM Sequence-to-Sequence
MySQL
Data Storage
Google Cloud Platform
Google Cloud - Compute & SQL Instance
tableau ONLINE
Presentation: Bitcoin Price Chart
twilio
Notifications: SMS Alerts (Buy / Sell)

Figure 1: Architecture block diagram

In summary we can divide this model in following parts:

i) Data collection and preprocessing
ii) Prediction engines – LSTM based DNN and ARIMA.
iii) Results are displayed using Tableau and automatic alert system

Now let us take a look at each part in detail.

2. Data collection and preprocessing

For our model we required 3 types of data – bitcoin price value , tweets and reddit discussions.

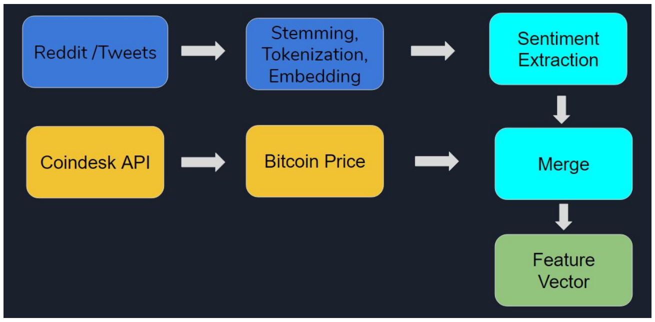


Figure 2: Data collection and preprocessing

From Figure-2 our data collection and preprocessing consist of following steps:

i) We collected bitcoin price data for every minute from Coindesk API.

ii) We collected Twitter data for every minute from Tweepy API using keywords "bitcoin", "btc".
iii) Also, we collected reddit data from reddit archives.
iv) After data collection, we did tokenization, stemming and stop words removal on twitter and reddit data using nltk library.
v) Later, we collected sentiment values for both twitter and reddit by using textblob.
vi) Then, we combined these sentiment values along with bitcoin price value to generate feature vector for that minute.

From Figure-1 after data preprocessing we feed these feature vectors as a time series to our two models – ARIMA and LSTM, where actual computation about the predictions take place.

Then we plotted several graphs to show our results like online bitcoin true vs predicted value graph, true sentiment value graph and so on. For implementing these online plots, we used tableau online tool and for that we stored all the predicted live data into MySQL database.

Then comes the main distinguishing feature of this project - implementation of automatic notification alert system using Twilio API. In simple words, we can say that these alerts notify registered users to buy or sell bitcoins. There exist two cases:

a. If bitcoin price > some fixed threshold of difference then it will notify to sell bitcoins.
b. If bitcoin price < some fixed threshold of difference then it will notify to buy bitcoins.

In a nutshell, our system fetches twitter and reddit data online every minute and predicts the bitcoin value for next minute based on sentiments of data using ARIMA and LSTM model. We demonstrated comparisons using plots for different comparisons. For bitcoin trading we implemented automatic notification alert system which notifies user to buy or sell bitcoin.

IV. Algorithm

In this section we are going to discuss two different models we used for prediction of bitcoin value in detail.

1. ARIMA model:

An autoregressive integrated moving average (ARIMA) model is a generalization of an autoregressive moving average (ARMA) model. Both of these models are fitted to time series data either to better understand the data or to predict future points in the series (forecasting).

ARIMA models are applied in some cases where data show evidence of non-stationarity, where an initial differencing step (corresponding to the "integrated" part of the model) can be applied one or more times to eliminate the non-stationarity.

The AR part of ARIMA indicates that the evolving variable of interest is regressed on its own lagged (i.e., prior) values. The MA part indicates that the regression error is actually a linear combination of error terms whose values occurred contemporaneously and at various times in the past. The I (for "integrated") indicates that the data values have been replaced with the difference between their values and the previous values (and this differencing process may have been performed more than once). The purpose of each of these features is to make the model fit the data as well as possible.

Non-seasonal ARIMA models are generally denoted ARIMA (p,d,q) where parameters p, d, and q are non-negative integers,

i) p: The number of lag observations included in the model, also called the lag order.
ii) d: The number of times that the raw observations are differences, also called the degree of differencing.
iii) q: The size of the moving average window, also called the order of moving average.

The implementation part of ARIMA model in python is covered in the 5th section in detail.

2. LSTM model:

The Long Short-Term Memory recurrent neural network has the promise of learning long sequences of observations. That's why in deep learning LSTM seems a perfect match for time series forecasting.

Long short-term memory (LSTM) block or network is a simple recurrent neural network which can be used as a building component or block (of hidden layers) for an eventually bigger recurrent neural network. The LSTM block is itself a recurrent network because it contains recurrent connections similar to connections in a conventional recurrent neural network.

An LSTM block is composed of four main components: a cell, an input gate, an output gate and a forget gate. The cell is responsible for "remembering" values over arbitrary time intervals; hence the word "memory" in LSTM.

Each of the three gates can be thought of as a "conventional" artificial neuron, as in a multi-layer (or feedforward) neural network: that is, they compute an activation (using an activation function) of a weighted sum.

Intuitively, they can be thought as regulators of the flow of values that goes through the connections of the LSTM; hence the denotation "gate".

An LSTM is well-suited to classify, process and predict time series given time lags of unknown size and duration between important events.

LSTMs were developed to deal with the exploding and vanishing gradient problem when training traditional RNNs. Relative insensitivity to gap length gives an advantage to LSTM over alternative RNNs, hidden Markov models and other sequence learning methods in numerous applications.

Implementation Details:

ARIMA Implementation:

Following are the steps for implementing ARIMA in python:

i) First, plot the data and observe trend if it is non-stationary then use differencing to make it stationary.

ii) Then, plot the autocorrelation and find lag order when autocorrelation is positive and above some threshold value say 0.5.

iii) An ARIMA model can be created using the statsmodels library as follows:

iv) Define the model by calling ARIMA () and passing in the p, d, and q parameters.

v) The model is prepared on the training data by calling the fit () function.

vi) Predictions can be made by calling the predict () function and specifying the index of the time or times to be predicted.

vii) Index of the next time step for making a prediction would be specified to the prediction function as start=101, end=101.

viii) We also would prefer the forecasted values to be in the original scale, in case we performed any differencing (d>0 when configuring the model). This can be specified by setting the typ argument to the value 'levels': typ='levels'.

ix) We can avoid all of these specifications for predict () by using the forecast () function.

x) A rolling forecast is required given the dependence on observations in prior time steps for differencing and the AR model. A crude way to perform this rolling forecast is to re-create the ARIMA model after each new observation is received. We manually keep track of all observations in a list called history that is seeded with the training data and to which new observations are appended each iteration.

LSTM Implementation:

We used an LSTM - based sequence to sequence model for this purpose. The encoder is a conditional probabilistic model, it learns the relation between input feature vector.

For this purpose we use the following setup:

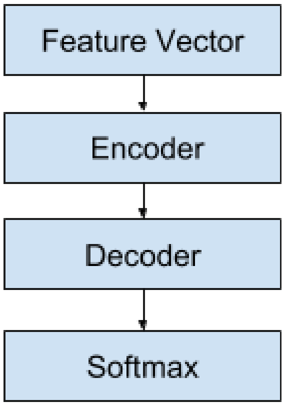


Figure 3: LSTM Model Overview

Hidden-Dimension of LSTM Encoder/Decoder: 512
Optimizer: Adam
Epochs:300
Number of LSTM layers: 2

We used Keras with Tensorflow as backend to generate the model.

V. Software Package Description

We have developed an end-to-end system to predict and alert the user about changes in bitcoin price.

Usage Details: The entire setup works on Google cloud.

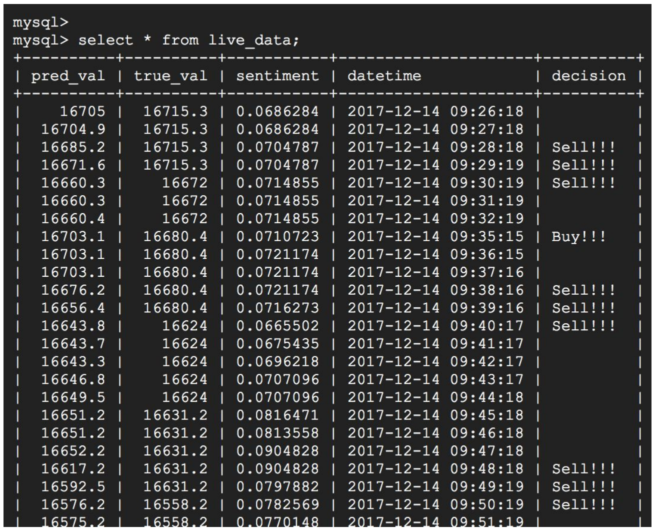
Part 1: Data Gathering:
In order to capture the real-time data, we run the following two python programs in background to continuously fetch the data.
a) Continuous_Stream_Data.py
b) Continuous_Stream_Sentiment.py

The two code do the preprocessing of data and store them in “live_tweet.csv” and “live_bitcoin.csv” files.

Part 2: Core Engine:
From the experiments we found LSTM based model to be performing better than ARIMA (discussed in detail in next section). We have set our best model parameters in engine.py file and once it is run it gathers data from the “live_bitcoin.csv” and “live_tweet.csv”, and generate features in real-time and is fed into the model.

The model outputs the next price. It also does a computation based on the threshold set in the code (this is fed from the settings file).

The information about the time stamp, predicted price, current real price and buy/sell decision is then written into a mysql database



```
mysql>
mysql> select * from live_data;
+----------+----------+-----------+---------------------+----------+
| pred_val | true_val | sentiment | datetime            | decision |
+----------+----------+-----------+---------------------+----------+
|    16705 |  16715.3 | 0.0686284 | 2017-12-14 09:26:18 |          |
|  16704.9 |  16715.3 | 0.0686284 | 2017-12-14 09:27:18 |          |
|  16685.2 |  16715.3 | 0.0704787 | 2017-12-14 09:28:18 | Sell!!!  |
|  16671.6 |  16715.3 | 0.0704787 | 2017-12-14 09:29:19 | Sell!!!  |
|  16660.3 |    16672 | 0.0714855 | 2017-12-14 09:30:19 | Sell!!!  |
|  16660.3 |    16672 | 0.0714855 | 2017-12-14 09:31:19 |          |
|  16660.4 |    16672 | 0.0714855 | 2017-12-14 09:32:19 |          |
|  16703.1 |  16680.4 | 0.0710723 | 2017-12-14 09:35:15 | Buy!!!   |
|  16703.1 |  16680.4 | 0.0721174 | 2017-12-14 09:36:15 |          |
|  16703.1 |  16680.4 | 0.0721174 | 2017-12-14 09:37:16 |          |
|  16676.2 |  16680.4 | 0.0721174 | 2017-12-14 09:38:16 | Sell!!!  |
|  16656.4 |  16680.4 | 0.0716273 | 2017-12-14 09:39:16 | Sell!!!  |
|  16643.8 |    16624 | 0.0665502 | 2017-12-14 09:40:17 | Sell!!!  |
|  16643.7 |    16624 | 0.0675435 | 2017-12-14 09:41:17 |          |
|  16643.3 |    16624 | 0.0696218 | 2017-12-14 09:42:17 |          |
|  16646.8 |    16624 | 0.0707096 | 2017-12-14 09:43:17 |          |
|  16649.5 |    16624 | 0.0707096 | 2017-12-14 09:44:18 |          |
|  16651.2 |  16631.2 | 0.0816471 | 2017-12-14 09:45:18 |          |
|  16651.2 |  16631.2 | 0.0813558 | 2017-12-14 09:46:18 |          |
|  16652.2 |  16631.2 | 0.0904828 | 2017-12-14 09:47:18 |          |
|  16617.2 |  16631.2 | 0.0904828 | 2017-12-14 09:48:18 | Sell!!!  |
|  16592.5 |  16631.2 | 0.0797882 | 2017-12-14 09:49:19 | Sell!!!  |
|  16576.2 |  16558.2 | 0.0782569 | 2017-12-14 09:50:19 | Sell!!!  |
|  16575.2 |  16558.2 | 0.0770148 | 2017-12-14 09:51:19 |          |
```

Figure 4: Data storage in mysql

Part 3: Tableau and Notification system:

We have used Tableau to generate plots in real-time form the sql-database mentioned in previous section.

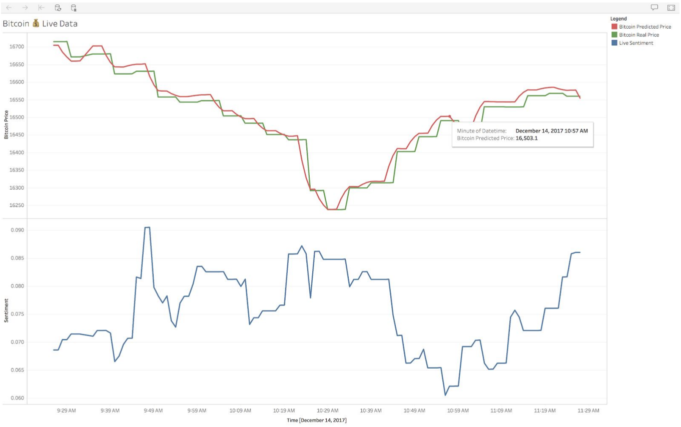
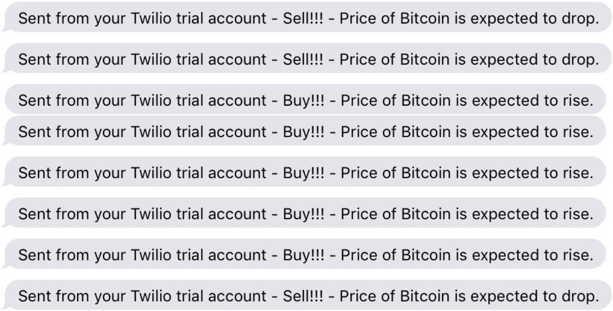


Figure 5: Data storage in mysql

Also, the engine sends out real time alerts in the following format:



Sent from your Twilio trial account - Sell!!! - Price of Bitcoin is expected to drop.

Sent from your Twilio trial account - Sell!!! - Price of Bitcoin is expected to drop.

Sent from your Twilio trial account - Buy!!! - Price of Bitcoin is expected to rise.

Sent from your Twilio trial account - Buy!!! - Price of Bitcoin is expected to rise.

Sent from your Twilio trial account - Buy!!! - Price of Bitcoin is expected to rise.

Sent from your Twilio trial account - Buy!!! - Price of Bitcoin is expected to rise.

Sent from your Twilio trial account - Buy!!! - Price of Bitcoin is expected to rise.

Sent from your Twilio trial account - Sell!!! - Price of Bitcoin is expected to drop.

Figure 6: Realtime SMS notifications using Twilio

VI. Experiment Results

Experiment 1: ARIMA vs LSTM

In this experiment we observed that for ARIMA model we recorded 104 mean squared error. Then we thought that this is pretty high and to get better performance we implemented LSTM model and got 42 mean squared error.

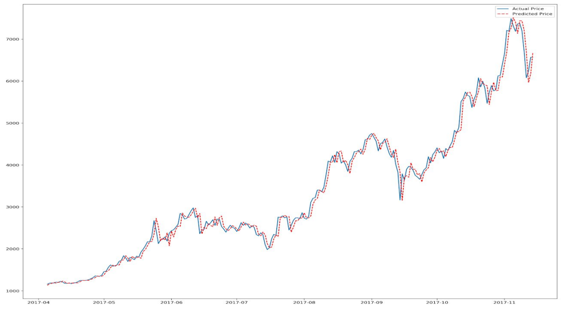


Figure 7: ARIMA model output

Experiment 2: Exploring look-backs in LSTM:
Lookback is like Markov chain, it tells us on how many previous states the current state depends on. Since we are dealing with time-series data - we would be using the past time-steps information to predict the next step. For this purpose we have done the following experiments and results are tabulated.

Input Sequence Length	RMSE
Lookback = 1	85
Lookback = 2	42
Lookback = 3	65
Lookback = 2 + Sentiment	32

Table 1: Results

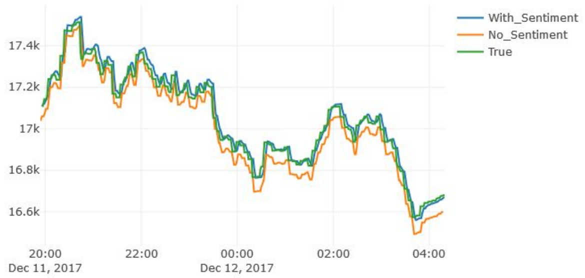


Figure 8: LSTM based model predictions

VII. Conclusion

In this project based on the results we can conclude that AI models with sentiment analysis predicted values which are much closer to true value compared to values without sentiment.

Also, based on our experiments we can conclude that deep learning approaches such as LSTM model are more accurate for forecasting time series than our traditional machine learning approach ARIMA model.

Based on feedback from the presentation we have incorporated a virtual trading feature in our code: we have entered a virtual currency of $16000 and traded it based on our buy/sell notifications form the system. In 2 days we have observed a profit of $30. From this we conclude that our approach is in right direction.

VIII. Future-work

Coming to the future works there is lot of scope to expand our project in several ways:

1. Another way is to extract twitter and reddit data based on the geographical locations say China or Korea which are leaders in the most transactions done for bitcoin.

2. We can try other information resources like news API or financial magazines or blogs for getting more accurate predicted value.

3. Another way which Prof. Lin suggested is that to apply this same approach for other crypto currencies like lightcoin or ethereum and so on.

IX. Acknowledgment

The Authors would like to thank Professor Ching Yung Lin and the TA’s who have been very helpful and making this course a wonderful learning experience.

X. References

[1] Mai, Feng and Bai, Qing and Shan, Zhe and Wang, Xin (Shane) and Chiang, Roger H.L., “From Bitcoin to Big Coin: The Impacts of Social Media on Bitcoin Performance,” (January 6, 2015).

[2] Hong Kee Sul, Alan R Dennis, and Lingyao Ivy Yuan.“Trading on twitter: Using social media sentiment to predict stock returns,”, Decision Sciences, 2016.

[3] Garcia D, Tessone CJ, Mavrodiev P, Perony N. “The digital traces of bubbles: feedback cycles between socio-economic signals in the Bitcoin economy,”, 2014

[4] Stuart Colianni, Stephanie Rosales, and Michael Signorotti. “Algorithmic trading of cryptocurrency based on twitter sentiment analysis,”, 2015.

[5] http://tweepy.readthedocs.io/en/v3.5.0/

[6] http://www.redditarchive.com/

[7] https://www.coindesk.com/api/

XI. Appendix

Contributions:

Name	Contributions	Total Percentage
Chandra S Narain Kappera (ck2840)	Data Collection code, LSTM based model development and Experiments, Nvidia-based cloud setup, real-time engine development, Notification system, Presentation Slides and Report, Video Report	33%
Venkata Sai Sriharsha Sammeta (vs2626)	Data Collection, Google Cloud based Setup and services, mysql Database development, ARIMA model development, Notification system, Presentation Slides and Report, Video Report.	33%
Ketan M Mehta (kmm2304)	Spark and Hadoop Exploration, ARIMA model development, Presentation Slides and Report.	33%

Table 2: Individual Contributions