Bitcoin Value Prediction Using Machine Learning

Akshay Bhandare*,Priya Jha*,Dhanraj Kirgat*,Shivani Shinde*,Imran Khan*

Department of Computer Engineering

JSPM's Imperial College of Engineering and Research, wagholi, pune.

Abstract-

In this research paper, we proposed prediction of bitcoin by analyzing different parameters. By studying different research papers, we were able to propose a effective method to predict the future value of bitcoin. The objective of this research paper is to predict the future value of bitcoin by using the data of previous values of bitcoin, the real time value of bitcoin, sentiments or opinions of people that are expressed on social platform like twitter, facebook or reddit and machine learning algorithms like ARIMA. To assess the expressions of people on social platforms, we proposed a sentiment analysis module. By giving the combined input of the polarity count we got from assessing the twitter,facebook and reddit data along with bitcoin historical price recorded for a long time and the current ptice of bitcoin,we can predict the future price of bitcoin.

Introduction-

Bitcoin is a crypto money that is used all over the world for early payment or basically for speculation purposes. Bitcoin is decentralized, for example, it is not owned by anyone. The exchanges made by Bitcoins are simple, since they are not attached to any country. Speculation should be possible through different commercial centers known as "bitcoin trading". These allow people sell / buy Bitcoins using various monetary forms. The biggest Bitcoin trade is Mt Gox. Bitcoins are kept in a advanced wallet that is essentially similar to a virtual one financial balance. The record of the considerable number of exchanges, the timestamp information is saved in one place called Blockchain. Each record of a blockchain is known as a square. Each square contains a pointer to a past information square. The information about the blockchain is scrambled. During exchanges, the customer's name is not discovered, however, only your wallet id is opened. The value of Bitcoin fluctuates simply like a stock, although in a unexpected way. Various calculations are used on financial exchange information for value forecasting. However, the parameters that influence Bitcoin are extraordinary. In this way it is important to anticipate the Bitcoin estimation so that the right risk decisions can be made. The cost of Bitcoin does not depend on the business occasions or the mediating government do not resemble values at all exchange. Therefore, to anticipate the value we feel It is important to use machine learning innovation to forecast the cost of Bitcoin.

Literature Survey-

Bitcoin is a trending technology and the most complex and expensive cryptocurrency in the world. Therefore, there are many algorithms that can predict the value of bitcoin. For the development of our project, we study different research works and they are the following:
We refer to the excellent article by B. Pang, L. Lee and S. Vaithyanathan where these investigations have established standards for opinion analysis oriented to machine learning. His technique is credited as one of the first attempts to use machine learning algorithms on the subject of opinion analysis [4].

We also use historical time series price data for bitcoin value forecasting and buy / sell methodology [1].

García et al. It also appeared that increases in polarization of opinion and trade volume precede the rise in Bitcoin prices [2].

Chen and Lazer [3] determined investment methodologies by observing and ranking Twitter feeds.

From [5] a comparison is made between the MLP multilayer perceptron and the nonlinear autoregressive exogenous model (NARX). They conclude that MLP can also be used for stock market prediction, although it does not outperform the NARX model in price prediction. The authors used the MATLAB neural network toolbox to build and evaluate network performance.

According to [6] the author is analyzing what has been done to predict the US stock market. the conclusion of their work is that the mean square error of the prediction network was as large as the standard deviation of the excess performance. However, the author provides evidence that several basic financial and economic factors have predictive power for excess market performance.
In [7] instead of directly forecasting the future price of the stock, the authors predict the trend of the stock. The trend can be considered as a pattern. They made both short-term predictions for days or weekly predictions as well as long-term predictions for months and found that the latter produced better results with 79% accuracy.

Sentiment Analysis-

Sentiment analysis (or sentiment mining) is a natural language processing technique used to determine whether data is positive, negative, or neutral. Sentiment analysis is often performed on textual data to help companies monitor brand and product sentiment in customer feedback and understand customer needs.

A large number of people who invest in bitcoin or have a keen interest in bitcoin express their opinions on social media.
Hence, for the more accurate prediction we are going to perform sentiment analysis in this project by using a social platform such as twitter, facebook or reddit.

To perform sentiment analysis on huge data like twitter tweets,facebook or reddit posts,we are going to use the python package called vader. i.e.
Valence Aware Dictionary and sEntiment Reasone.

What is vader?-

VADER belongs to a type of sentiment analysis that is based on lexicons of sentiment-related words. In this approach, each of the words in the lexicon is rated as to whether it is positive or negative, and in many cases, **how** positive or negative. Below you can see an excerpt from VADER's lexicon, where more positive words have higher positive ratings and more negative words have lower negative ratings.

This is how the vader sentiment analyzer works:

- VADER uses a combination of a sentiment lexicon which is a list of lexical characteristics (for example, words) that are generally labeled according to their semantic orientation as positive or negative.
- The sentiment analyzer not only reports on the positivity and negativity score, but also tells us how positive or negative a feeling is.
- Next, we use the polarity_scores () method to get the polarity indices for the given sentence.
- Then, we construct the intensity and polarity of the comment as:

```
demo = SentimentIntensityAnalyzer()
tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')
with open('kindle.txt', encoding='ISO-8859-2') as f:
    for text in f.read().split('\n'):
        print(text)
        scores = demo.polarity_scores(text)
        for key in sorted(scores):
            print('{0}: {1}, '.format(key, scores[key]), end='')
    print()
```


Let's understand what opinion code is and how VADER works in the above code output:

"I love my children."

composite: 0.6369, neg: 0.0, neu: 0.323, pos: 0.677

• The Positive (pos), Negative (neg) and Neutral (neu) scores represent the proportion of text that falls into these categories. This means that our judgment was rated 67% Positive, 32% Neutral and 0% Negative. So all of these should add up to 1.
• Composite Score is a metric that calculates the sum of all Lexical Grades that have been normalized between -1 (negative end) and +1 (positive end).
• Finally, feedback opinion scores are returned.

Sentiment analysis for facebook-

There are many ways to get Facebook comments, they are:

- Facebook graphics API
- Direct download from Facebook
- Download from sites of another dataset provider

Among the above methods, we use Facebook comment dataset download from Kaggle website, which is the best dataset provider.

We follow these important steps in our program:

- Download (retrieve) Facebook comment from Kaggle site and save as text format.
- Pre-processing the data through the SkLearn and nltk libraries. We first tokenize the data and then after tokenization we lemmatize and lemmatize.
- Analyze comments using Vader's library. Classify each comment as positive, negative, or neutral.

Twitter sentiment analysis using vader-

Twitter sentiment analysis python:

Analysis of Twitter sentiment using Python can be done through popular Python libraries such as Tweepy and TextBlob.

Tweepy- Tweepy, the Python client for the official Twitter API, supports access to Twitter through basic authentication and the newer method, OAuth. Twitter has stopped accepting Basic Authentication, so OAuth is now the only way to use the Twitter API.

Tweepy gives access to the well-documented Twitter API. Tweepy allows you to get an object and use any method offered by the official Twitter API. The main classes of models in the Twitter API are Tweets, Users, Entities, and Places. Accessing each one returns a response in JSON format and traversing the information is very easy in Python.

TextBlob- TextBlob, one of the popular Python libraries for processing textual data, is found in the NLTK. It works as a framework for almost all the necessary tasks that we need in Basic NLP (Natural Language Processing). TextBlob has some advanced features like:

- Sentiment extraction
- Spell correction

TextBlob is useful for Twitter sentiment analysis Python in the following ways:

1.Tokenization:

TextBlob can convert text blocks into different sentences and words. This makes reading between the lines much easier.

Provide all this data to VADER, for calculating polarity of data and performing sentiment analysis for twitter.

Sentiment analysis for reddit-

Let's begin with an overview of the process for this project:

1. Establish a Reddit instance via API
2. Obtain comments from a post on Reddit
3. Preprocess the comments
4. Apply a sentiment analyzer model (VADER)
5. Obtain sentiment results and create some simple visualizations, e.g. word clouds

1.Establishing A Reddit Instance with Praw

We would be working with a Python API Wrapper to obtain data in the form of comments for our analysis. For Reddit, we would be utilizing the PRAW API Wrapper. As Reddit's API uses OAuth2, a Reddit connection must be established with a client ID and client secret. These values can be found after creating your app.

2.Obtaining Comments From A Post

Next, we have to create a submission object before extracting the comments from a Reddit subreddit post.

3. Preprocess The Comments

When dealing with raw or uncleaned data, we would first have to preprocess or "clean" it before modelling or in our case, to apply a sentiment analysis.

In our instance, we would include the following steps for our preprocessing:

- **Removing Emojis**
- **Tokenizing, removing links etc.**
- **Removing stopwords**
- **Normalizing words via lemmatizing**

We first begin by converting our list into a string object before passing them through each preprocessing process.

Preprocessing String - Convert To A String Object

```
In [10]: List1 = Comments_All
         List1 = [str(i) for i in List1]    # map to a list of strings
         string_uncleaned = ' , '.join(List1)           # join all the strings separated by a comma
         string_uncleaned
```

Next, we would remove Emojis that are in our string object.

Preprocessing String - Removing Emojis

```
In [11]: string_emojiless = emoji.get_emoji_regexp().sub(u'', string_uncleaned)
```

Followed by tokenizing and cleaning the string via a regular expression tokenizer. Tokenizing breaks apart every word in the string into an individual word which would then carry it's own "positive" or "negative" sentiment based on our sentiment analyzer later.

Preprocessing String - Tokenizing & Cleaning Strings

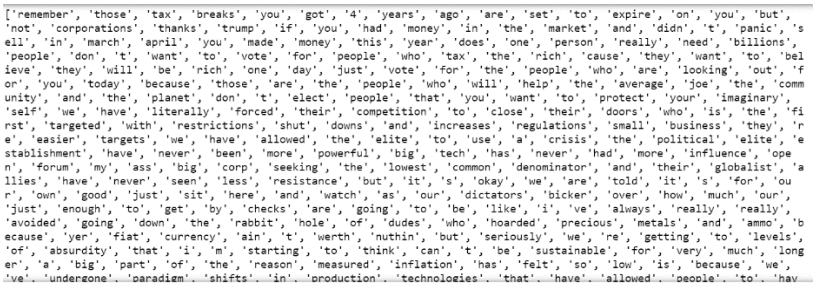
```
In [14]: tokenizer = RegexpTokenizer('\w+|\$[\d\.]+|http\S+')
         tokenized_string = tokenizer.tokenize(string_emojiless)
         print(tokenized_string)
```


As you can see, some tokens are uppercased, and some are lowercased. To standardise this, we would convert all tokens to lower case and store it into a list.

```
['Remember', 'those', 'tax', 'breaks', 'you', 'got', '4', 'years', 'ago', 'are', 'set', 'to', 'expire', 'ON', 'YOU', 'but',
'not', 'corporations', 'THANKS', 'TRUMP', 'If', 'you', 'had', 'money', 'in', 'the', 'market', 'and', 'didn', 't', 'panic', 's
ell', 'in', 'March', 'April', 'you', 'made', 'money', 'this', 'year', 'Does', 'one', 'person', 'really', 'need', 'billions',
'People', 'don', 't', 'want', 'to', 'vote', 'for', 'people', 'who', 'tax', 'the', 'rich', 'cause', 'they', 'want', 'to', 'bel
ieve', 'they', 'will', 'be', 'rich', 'one', 'day', 'Just', 'vote', 'for', 'the', 'people', 'who', 'are', 'looking', 'out', 'f
or', 'YOU', 'today', 'Because', 'those', 'are', 'the', 'people', 'who', 'will', 'help', 'the', 'average', 'Joe', 'the', 'comm
unity', 'and', 'the', 'planet', 'Don', 't', 'elect', 'people', 'that', 'you', 'want', 'to', 'protect', 'your', 'imaginary',
'self', 'We', 'have', 'literally', 'forced', 'their', 'competition', 'to', 'close', 'their', 'doors', 'who', 'is', 'the', 'fi
rst', 'targeted', 'with', 'restrictions', 'shut', 'downs', 'and', 'increases', 'regulations', 'Small', 'business', 'They', 'r
e', 'easier', 'targets', 'We', 'have', 'allowed', 'the', 'elite', 'to', 'use', 'a', 'crisis', 'The', 'political', 'elite', 'e
stablishment', 'have', 'never', 'been', 'more', 'powerful', 'Big', 'tech', 'has', 'never', 'had', 'more', 'influence', 'ope
n', 'forum', 'my', 'ass', 'Big', 'Corp', 'seeking', 'the', 'lowest', 'common', 'denominator', 'and', 'their', 'globalist', 'a
llies', 'have', 'never', 'seen', 'less', 'resistance', 'But', 'it', 's', 'okay', 'we', 'are', 'told', 'it', 's', 'for', 'ou
r', 'own', 'good', 'Just', 'sit', 'here', 'and', 'watch', 'as', 'our', 'dictators', 'bicker', 'over', 'how', 'much', 'our',
'just', 'enough', 'to', 'get', 'by', 'checks', 'are', 'going', 'to', 'be', 'Like', 'I', 've', 'always', 'really', 'really',
'avoided', 'going', 'down', 'the', 'rabbit', 'hole', 'of', 'dudes', 'who', 'hoarded', 'precious', 'metals', 'and', 'ammo', 'b
ecause', 'yer', 'fiat', 'currency', 'ain', 't', 'werth', 'nuthin', 'but', 'seriously', 'we', 're', 'getting', 'to', 'levels',
'of', 'absurdity', 'that', 'I', 'm', 'starting', 'to', 'think', 'can', 't', 'be', 'sustainable', 'for', 'very', 'much', 'long
er', 'A', 'big', 'part', 'of', 'the', 'reason', 'measured', 'inflation', 'has', 'felt', 'so', 'low', 'is', 'because', 'we',
've', 'undergone', 'paradigm', 'shifts', 'in', 'production', 'technologies', 'that', 'have', 'allowed', 'people', 'to', 'hav
```

Preprocessing String - Converting Tokens Into Lowercase

```
In [15]: lower_string_tokenized = [word.lower() for word in tokenized_string]
         print(lower_string_tokenized)
```



```
['remember', 'those', 'tax', 'breaks', 'you', 'got', '4', 'years', 'ago', 'are', 'set', 'to', 'expire', 'on', 'you', 'but',
'not', 'corporations', 'thanks', 'trump', 'if', 'you', 'had', 'money', 'in', 'the', 'market', 'and', 'didn', 't', 'panic', 's
ell', 'in', 'march', 'april', 'you', 'made', 'money', 'this', 'year', 'does', 'one', 'person', 'really', 'need', 'billions',
'people', 'don', 't', 'want', 'to', 'vote', 'for', 'people', 'who', 'tax', 'the', 'rich', 'cause', 'they', 'want', 'to', 'bel
ieve', 'they', 'will', 'be', 'rich', 'one', 'day', 'just', 'vote', 'for', 'the', 'people', 'who', 'are', 'looking', 'out', 'f
or', 'you', 'today', 'because', 'those', 'are', 'the', 'people', 'who', 'will', 'help', 'the', 'average', 'joe', 'the', 'comm
unity', 'and', 'the', 'planet', 'don', 't', 'elect', 'people', 'that', 'you', 'want', 'to', 'protect', 'your', 'imaginary',
'self', 'we', 'have', 'literally', 'forced', 'their', 'competition', 'to', 'close', 'their', 'doors', 'who', 'is', 'the', 'fi
rst', 'targeted', 'with', 'restrictions', 'shut', 'downs', 'and', 'increases', 'regulations', 'small', 'business', 'they', 'r
e', 'easier', 'targets', 'we', 'have', 'allowed', 'the', 'elite', 'to', 'use', 'a', 'crisis', 'the', 'political', 'elite', 'e
stablishment', 'have', 'never', 'been', 'more', 'powerful', 'big', 'tech', 'has', 'never', 'had', 'more', 'influence', 'ope
n', 'forum', 'my', 'ass', 'big', 'corp', 'seeking', 'the', 'lowest', 'common', 'denominator', 'and', 'their', 'globalist', 'a
llies', 'have', 'never', 'seen', 'less', 'resistance', 'but', 'it', 's', 'okay', 'we', 'are', 'told', 'it', 's', 'for', 'ou
r', 'own', 'good', 'just', 'sit', 'here', 'and', 'watch', 'as', 'our', 'dictators', 'bicker', 'over', 'how', 'much', 'our',
'just', 'enough', 'to', 'get', 'by', 'checks', 'are', 'going', 'to', 'be', 'like', 'i', 've', 'always', 'really', 'really',
'avoided', 'going', 'down', 'the', 'rabbit', 'hole', 'of', 'dudes', 'who', 'hoarded', 'precious', 'metals', 'and', 'ammo', 'b
ecause', 'yer', 'fiat', 'currency', 'ain', 't', 'werth', 'nuthin', 'but', 'seriously', 'we', 're', 'getting', 'to', 'levels',
'of', 'absurdity', 'that', 'i', 'm', 'starting', 'to', 'think', 'can', 't', 'be', 'sustainable', 'for', 'very', 'much', 'long
er', 'a', 'big', 'part', 'of', 'the', 'reason', 'measured', 'inflation', 'has', 'felt', 'so', 'low', 'is', 'because', 'we',
've', 'undergone', 'paradigm', 'shifts', 'in', 'production', 'technologies', 'that', 'have', 'allowed', 'people', 'to', 'hav
```

Next, we would remove stop words in our list object.

Stop words are words that do not add much information to a sentence. As such, they can safely be ignored without sacrificing the meaning of the sentence.

Preprocessing String - Removing Stopwords

```
In [17]: nlp = en_core_web_sm.load()

         all_stopwords = nlp.Defaults.stop_words

         text = lower_string_tokenized
         tokens_without_sw = [word for word in text if not word in all_stopwords]

         print(tokens_without_sw)
```

Lastly, we would normalize the words via a process called: Lemmatizing or Stemming.
Both processes are used to trim words down to their root words. However, stemming might return a root word that is not an actual word whereas, lemmatizing returns a root word that is an actual language word.
As such, we would only be applying the lemmatizing process to our list object.

Preprocessing String - Normalizing Words via Lemmatizing

```
In [25]: lemmatizer = WordNetLemmatizer()

         lemmatized_tokens = ([lemmatizer.lemmatize(w) for w in tokens_without_sw])
         print(lemmatized_tokens)
```

Once we obtain our cleaned output, we would calculate each tokenized word's polarity scores using the VADER

Prediction Using ARIMA-

ARIMA stands for Autoregressive Integrated Moving Average.

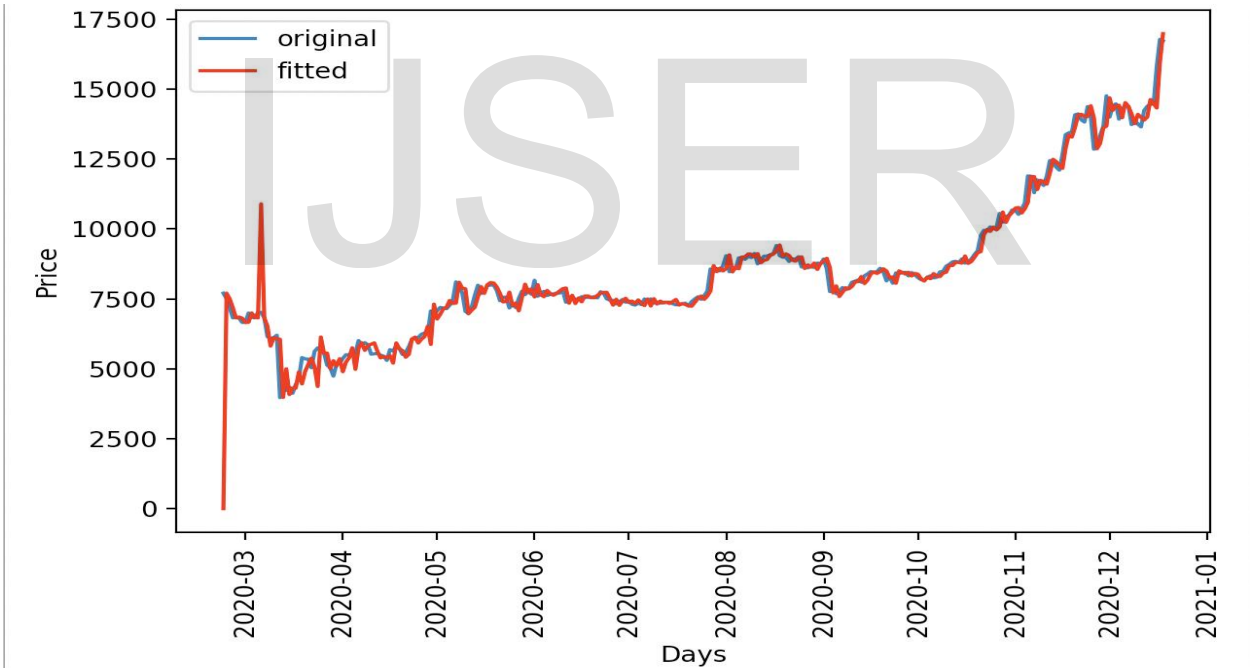
An Autoregressive Integrated Moving Average, or ARIMA, is a statistical analysis model that uses time series data to better understand the data set or to predict future trends.

An autoregressive integrated moving average model is a form of regression analysis that measures the strength of a dependent variable relative to other changing variables. The objective of the model is to predict future movements in the financial or securities market by examining the differences between the values in the series rather than through the actual values.

An ARIMA model can be understood by describing each of its components as follows:
• Autoregression (AR) refers to a model that shows a changing variable that regresses on its own lagging or previous values.
• Integrated (I) represents the differentiation of raw observations to allow the time series to become stationary, that is, the data values are replaced by the difference between the data values and the previous values.
• The moving average (MA) incorporates the dependence between an observation and a residual error of a moving average model applied to lagged observations.

Each component works as a parameter with a standard notation. For ARIMA models, a standard notation would be ARIMA with p, d, and q, where integer values substitute for parameters to indicate the type of ARIMA model used. Parameters can be defined as:
• p: the number of lag observations in the model; also known as delay order.
• d: the number of times the raw observations are differentiated; also known as degree of differentiation.
• q: the size of the moving average window; also known as the order of the moving average.

In a linear regression model, for example, the number and type of terms are included. A value of 0, which can be used as a parameter, would mean that that particular component should not be used in the model. In this way, the ARIMA model can be built to perform the function of an ARMA model, or even simple AR, I or MA models.



Result of ARIMA prediction

Conclusion-

The study focuses on the current sentiments of people and prediction of bitcoin price. In this paper we conclude that survey report will be just introducing modules of Bitcoin price prediction and machine algorithms.We also concluded that ARIMA is outperforming the LSTM algorithm.The sentiment analysis module had a huge part in development of this project.The data from facebook,twitter and reddit were very helpful for us along with the real time data of bitcoin.The prediction of bitcoin price using sentiment analysis and the ARIMA module were mostly accurate.

References-

[1] M. Amjad and D. Shah, "Trading Bitcoin and Online Time Series Prediction," in NIPS 2016 Time Series Workshop, 2017.

[2] D. Garcia and F. Schweitzer, "Social signals and algorithmic trading of Bitcoin," Royal Society Open Science, vol. 2, no. 9, 2015.

[3] R. Chen and M. Lazer, "Sentiment Analysis of Twitter Feeds for the Prediction of Stock Market Movement," Stanford Computer Science, no. 229, 2011, p. 15. [6] A. Go, L. Huang and R. Bhayani, "Twitter Sentiment Classification using Distant Supervision," Stanford Computer Science, 2009.

[4] B. Pang, L. Lee and S. Vaithyanathan, "Thumbs up: sentiment classification using machine learning techniques," in ACL-02 conference on Empirical methods in natural language processing, Philadelphia, PA, USA, 2002.

[5]M. Daniela and A. BUTOI, "Data mining on Romanian stock market using neural networks for price prediction," Informatica Economica, vol. 17, no. 3, 2013.

[6] H. Jang and J. Lee, "An Empirical Study on Modelling and Prediction of Bitcoin Prices with Bayesian Neural Networks based on Blockchain Information," in IEEE Early Access Articles, 2017.

[7]F. A. d. Oliveira, L. E. Zarate, M. d. A. Reis and C. N. Nobre, "The use of artificial neural networks in the analysis and prediction of stock prices," in IEEE International Conference on Systems, Man, and Cybernetics, 2011.