

Predicting Bitcoin price fluctuation with Twitter sentiment analysis

Master’s in Computer Science

1825271 AYAZ SARA BINTE

1834138 CESTONARO CHRISTOPHE

1832624 DE BENEDETTI JUAN

1827435 DKMAK GHAITH

1815460 OUES NAMIR

1730455 WARNECKA KATARZYNA

**Abstract**

In the last decade, Web 2.0 services such as Twitter have been used as a communication services with spectacular results. Around the same time, Bitcoin the largest cryptocurrency, emerged as a radical phenomenon in the financial system, attracting huge number of investors. The unique feature of Bitcoin is that a price fluctuation depends on people’s opinion instead of money regulations. Increasing importance of Twitter might influence purchasing decisions, by informing users of the currency and its popularity, decisions and individual behavior are being affected. As a result, cryptocurrency user or trader can take purchasing and selling advantage, by considering the impact of tweets on price direction. This paper aims to evaluate whether Twitter sentiment relating to Bitcoin can make use of development advantageous trading strategies. This research is concerned with determining the return value in the next hour of Bitcoin using tweets associated with digital currency, by predictive examination of relationship between Bitcoin returns and tweets. The popular ARIMA model for time series forecasting is implemented as a benchmark for comparison to machine learning techniques integrating social media as an input, i.e. messages that are supposedly triggering market movements. By taking this approach, users’ opinion of the virtual currency, in the form of tweets, has been labeled as having positive, neutral or negative impact. Also, in this study we conduct practical analysis on modeling and predicting of the Bitcoin using Recurrent neural networks and long short-term memory. The findings from this work might be relevant for investors and policymakers whether to invest or no using sentiment analysis since Bitcoin is a financial asset without any connection to the measures of central banks.

Table of Contents

[1 Introduction 5](#_Toc7026441)

[1.1 Related Research 6](#_Toc7026442)

[1.1.1 Problem statement 7](#_Toc7026443)

[1.1.2 Scope 7](#_Toc7026444)

[1.1.3 Project management approach and teamwork 7](#_Toc7026445)

[1.1.4 Collaboration tools 9](#_Toc7026446)

[2 Background 9](#_Toc7026447)

[2.1 Bitcoin 9](#_Toc7026448)

[2.1.1 Forecasting Price Trends: 10](#_Toc7026449)

[2.1.2 Time Series Analysis: 11](#_Toc7026450)

[2.2 Sentiment Analysis 11](#_Toc7026451)

[2.2.1 Natural Language Processing using naïve Bayes 12](#_Toc7026452)

[2.2.2 NLP with Convolutional Neural Network 14](#_Toc7026453)

[2.2.3 NLP with Random Forest 16](#_Toc7026454)

[2.3 Algorithms comparison 18](#_Toc7026455)

[3 Method 18](#_Toc7026456)

[3.1 Data Collection 18](#_Toc7026457)

[3.1.1 Bitcoin price data 18](#_Toc7026458)

[3.2 Gathering tweets 20](#_Toc7026459)

[3.2.1 Acquiring tweets using twitter streaming API 20](#_Toc7026460)

[3.3 Sentiment analysis process 22](#_Toc7026461)

[3.3.1 Reducing noise in the twitter dataset 22](#_Toc7026462)

[3.3.2 Individual tweet sentiment analysis 24](#_Toc7026463)

[4 Model 24](#_Toc7026464)

[4.1 Recurrent Neural Network and Long Short-Term Memory 24](#_Toc7026465)

[5 Results 25](#_Toc7026466)

[5.1 Data Collection 25](#_Toc7026467)

[5.1.1 Bitcoin data-set 25](#_Toc7026468)

[5.1.2 Evaluation of the model: 26](#_Toc7026469)

[5.2 Twitter data-set 27](#_Toc7026470)

[5.2.1 Results Natural Language Processing using naïve Bayes 27](#_Toc7026471)

[5.2.2 Results for NLP with Convolutional Neural Network 28](#_Toc7026472)

[5.2.3 Results for NLP with Random Forest 31](#_Toc7026473)

[5.3 Prediction performance 31](#_Toc7026474)

[5.3.1 Multivariate Time Series Forecasting with LSTMs 31](#_Toc7026475)

[5.4 The prediction model implementation 34](#_Toc7026476)

[6 Discussion 35](#_Toc7026477)

[6.1 Prediction model 35](#_Toc7026478)

[7 Conclusion 37](#_Toc7026479)

[8 References 40](#_Toc7026480)

# Introduction

Social media capture the “wisdom of the crowd” and provide various snippets of information regarding a financial commodity (Colianni et al., 2015). Through the social media users can share ideas and thoughts about different aspects of world and gain detailed knowledge about traders’ experience, which can be used as an excellent source of learning (Stengvist, E., Lonno, J., 2017). Thus, they become new source of information that can be collected in a bigger volume and in a real time which provide more opportunity of online data than it was available before (McAtter, C., 2014). Sites like Twitter specifically has become a place that intensify word of mouth effect. Twitter is a place where consumer sentiment and public opinion about investing decision has been quickly distributed (Stengvist, E., Lonno, J., 2017). Due to a fact that Twitter is a major social platform, available worldwide it provides rapid growth and significant adoption of the information shared between businesses and individuals (Matta et al., 2014). The approach was to train dataset using direct text from Twitter, known as tweets, that has been rated positive, negative or neutral within each post (Colianni et al., 2015).

On the other hand, time series prediction is not a new phenomenon. There has been significant amount of research done predicting financial markets such as stock markets. However, Bitcoin poses different features than financial market currencies. Unlike existing currencies with central banks, Bitcoin aims to achieve complete decentralization since its creation in 2008 by Satoshi Nakamoto (Matta et al., 2014). It operates on a decentralized, peer-to-peer and trustless system in which all transactions are posted to an open ledger called Blockchain. Users of the Bitcoin market build trust relationships through the formation of Blockchain based on cryptography techniques using hash function. Hence, various researches are taken also in a field of machine learning (Jang., H., Lee, J., 2017) Because Bitcoin enables transactions of services and goods and it is not managed by government, it has been attracting a lot of attention. The power of bitcoin price relates to experts and user’s opinion expressed mostly on online social media (Mai et al., 2014). Due to the open nature of Bitcoin and high volatility in the market it is interesting time series prediction problem, especially in a market that is in a transitory stage. Traditional time series prediction such as Holt- Winters models are more suitable for forecasting data where seasonal effects are present. In case of Bitcoin, where lack of seasonality occurs, ARIMA model have shown promising results (McNally, S., 2016).

In this work we analyze whether social media activity on Twitter can be used by investment professionals. There have been several former researches predicting correlation between Bitcoin and social media. We decided to use Sentiment Analysis in order to analyze users of Twitter opinions about Bitcoin, expressed in a large-scale collections of daily Twitter posts (Matta et al., 2014). In terms of financial markets, sentiment can be seen as a positive, negative or neutral value about the investment. Our strategy applies supervised machine learning algorithms including Random Forest, Naïve Bayes and Neural Networks to determine whether the price of Bitcoin will increase or decrease within a predetermined time interval (Colianni et al., 2015).

Also, this research will be mostly beneficial for people interested in sentiment analysis of microblogging data and those who would like to invest in Bitcoin. Former studies in this area have shown that people opinions can be used with advantage as a predictor for financial markets. The importance of sentiment is especially crucial for businesses, because it provides information whether a person is willing to purchase a product or service. Hence, the risk and harness of investment in virtual currencies can be diminished (McAtter, C., 2014).

## Related Research

Research on predicting the price of Bitcoin using machine learning algorithms specifically is lacking. Due to a limited research, we wanted to explore this field.

A paper Algorithmic Trading of Cryptocurrency Based on Twitter Sentiment Analysis (Colianni et al., 2015) analyzed how tweets sentiment could impact investments decisions specifically on Bitcoin. They achieved 90% accuracy hour-by-hour and day-by-day. This was achieved through robust error analysis on the input data, which on average yielded a 25% better accuracy. Their data was labelled using online text sentiment API. Hence, their model corresponds more to how performance matches the online text sentiment API, not the accuracy in terms of predicting price fluctuations.

Also, Stenqvist and Lonno (2016) applied deep learning algorithms on every 30 minutes scale. This approach leaded to 79% accuracy in predicting Bitcoin using 2.27 tweets. None of these models used data labelled based on price fluctuation and they did not analyze the size of percentage change in the predicting models.

### Problem statement

The examination of sentiment of opinions distributed on Twitter regarding Bitcoin and its comparison with Bitcoin’s return price, has been performed in order to answer the following questions:

* Can a prediction model based on sentiment changes yield better than random accuracy?
* With what accuracy can the direction of the price of Bitcoin be predicted using sentiment on Twitter?

### Scope

Sentiment has been used only from Twitter due to its well-established position in micro-blogging sphere and easy access to programmatic interface for data collection. Tweets are publicly accessible and can be searched for or followed in real time. They provide honest and immediate opinion about users likes or dislikes. Consumer sentiment was extracted from these tweets using specific tools and techniques. Likewise, analysis has been limited only to Bitcoin, because it is the most established cryptocurrency both in age and market share. As stated previously, Twitter is free platform that could easily keep up to date users that are active or just follow the topic of their interest. Hence, Bitcoin users that are most likely to be technology savvy, so services like Twitter could reflect their opinion. However, the prediction model could be used for any cryptocurrency, by providing the underlying data collection mechanism with identifying keywords. Bitcoin transactions were collected over specific time intervals and trading volume in BTC. The key value was limited to an increase or decrease in price return. The sentiment has been analyzed in a short term and it does not correlate to macro trends in cryptocurrency.

### Project management approach and teamwork

*Project management approach*

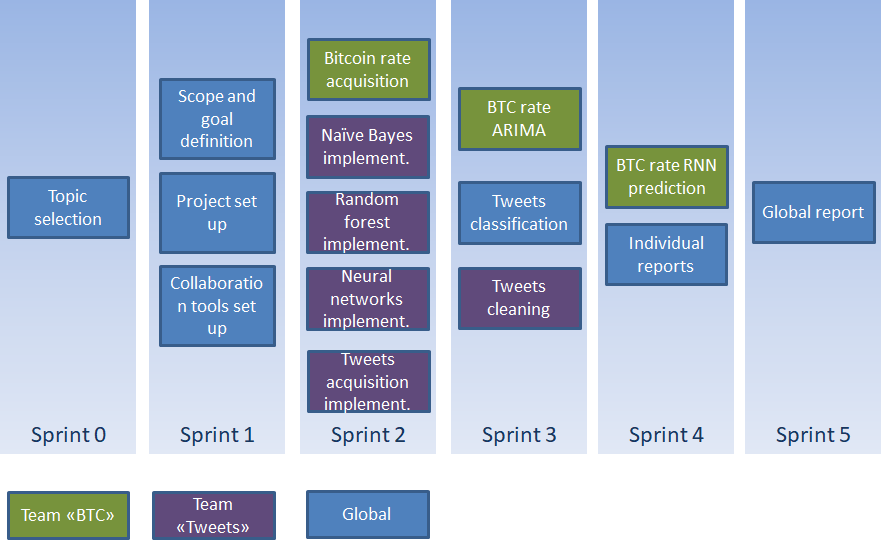
Due to a fact that the team had no prior collaboration experience and the topic was mostly new to its members, as well as the technology (python) and several ML techniques, it was clear to us that an agile project management method would help minimize evident and important risks related to it. Therefore, we decided to apply such an approach to define sprints and allocate them to each team member, so we could get a working solution as soon as possible, even if not complete functionally and work in parallel. This approach would allow us to improve the prototype if time allowed, as well as, making sure a minimal working solution would be reached.

*Agile method*

The overall approach was to define “work packages” and two sub-groups to organize work. Each subgroup was assigned to one of the topics: Bitcoin rate and Twitter. Each subgroup was responsible for importing the data and implementing the appropriate classification / forecasting method. When those tasks were completed, the results could be put together to produce the expected final result. The resulting split is depicted in the image below.

The team had been meeting about once a week to have a “SCRUM” meeting, because of each member constraints. Also, constant contact was maintained through emails/instant messaging, in order to coordinate the progress and solve issues. Sprints length were not equal, as some tasks where longer in nature, such as algorithm implementation and testing.

The team used “Trello”, as an online tool, in order to define tasks and assign them to each member. No actual “user stories” were written, however tasks were formulated to be clear and concise. Overall, written documentation was kept to a minimum, as the effort has been put in developing the solutions and communicating in real time in a less formal manner.



### Collaboration tools

* Trello: was used to define tasks, assign them and keep track of progress. Tasks have been organized in a “kanban” way, i.e. “to do”, “in progress” and “done”. They have been assigned to members and a due date defined. A Gantt view has been set up, in order to get a graphical view of the task’s lengths.
* Github: was used to store code, documents and small datasets. Although most of the code was modified only by its author. This allowed to have a central repository and to keep track of changes.
* Kaggle: was used to store large datasets, which proved to be needed when working with our downloaded Tweets volume.
* Instant messaging / emails were used to communicate within the team and exchange ideas and documents.

# Background

## Bitcoin

Bitcoin released year was 2009, It is known as the world’s first cryptocurrency or digital currency that exists merely in an electronic form. The powerful aspect in the Bitcoin and other cryptocurrencies is considered to be decentralized, no central issuing authorities or political institutions controlling the amount of Bitcoin. On the contrary it is far away from anarchy.

In the time being, there are about 17 million bitcoins in circulation. The supply of the Bitcoin is utterly controlled by design and that means no authority or government controlling the supply. Thus, people believe that Bitcoin can empower them by overthrowing current currency power that is influenced by political agendas.

Bitcoin is one of many cryptocurrencies; the community has developed platforms controlled by protocols to make the process simple and organized. For example, there is a peer-to-peer network through which digital currency (Bitcoin) holders can transfer their digital money. These transfers are tracked and monitored on blockchains.

The chain responsibility is to record every bitcoin transaction has been made. Each block in the chain is built up of an encrypted structure based on Merkle Trees (Sasson et al., 2014). This approach is useful for detecting any fraud attempt or corrupted files. Any corrupt or fraudulent file in the chain, the blockchain prevents it from doing any damaging to the other blocks in the chain. Blockchain basically is backbone of the Bitcoin which ensures the transactions are accurate.

### Forecasting Price Trends:

Forecasting price fluctuation and movement is quite similar to gambling as there is always probability for winning as well as losing – nobody is always right all the time. Many traders have lost lots of money due to wrong trend forecasting or even misreading and misinterpreting patterns of previous data.

The two main approaches are known for financial forecasting, fundamental and technical analysis. While the first one is related to analyzing financial statement of business i.e. assets, earning, market and competitors, technical analysis tries to predict the direction of prices (up or down) based on historical data of trading namely volumes and prices. Trend lines, Moving average and Trading volume techniques can be used to extrapolate the previous data and make prediction on what the next move could be.

Computers and high computational processing come handy in predicting Bitcoin’s future moves. Different Machine Learning algorithms and mathematical and statistical models are being applied on such prediction. Many researchers are trying to optimize ML algorithms such as Recurrent Neural Networks (RNN), Long Short-term Memory (LSTM), Bayesian Neural Networks (BNN).

According to (McNally, S., 2016), predict the Bitcoin movements using machine learning techniques still result in a poor predictive Time Series performance comparing to other econometrics models such as ARIMA or generalized autoregressive conditional heteroskedasticity (GARCH).

### Time Series Analysis:

A time series is a sequence of numerical data in consecutive order. It tracks and records the changes and results of observations over a specified period at particular and different intervals. Time series Analysis is beneficial and effective to see how an economic or any other type of variable such as soil temperature changes over time and how these changes affect the future.

There are many methods and models that can be applied in Time Series data. The purpose of using these models varies according to the data specifications. For example, Naïve model is the simplest forecasting model for the most recent observation. It is not good at prediction, but it is normally used as a benchmark for other prediction models. Another example of Time Series analysis is exponential smoothing which is implemented to predict short term prediction. Other models such as ARIMA, GARCH, ARCH and others are more advanced than Naïve and exponential smoothing as they test the linear or non-linear relationships of dependent variables. In this research, we will use ARIMA for forecasting the return value of the next hour.

## Sentiment Analysis

Sentiment Analysis is also known as “opinion mining” and it is a process of analyzing a text in order to extract text information such as polarity through Natural Language Processing (NLP). NLP belongs to the field of Artificial Intelligence (AI) and it refers to the process where programs can process and analyze human language. Human Language is complex, thus it is challenging to implement while considering the hard to examine aspects of language, like slang and sarcasm. Computers can identify target words and determine whether a sentence has a positive or negative tone to it, for example by using a dictionary. For instance, the machine understands that “lovely” is a positive word and “awful” is negative.

The research area of sentiment analysis began to grow in 2001 when the machine learning techniques related to NLP started to develop (Pang and Lee, 2008). Nowadays, sentiment classification has an enormous contribution to the research process for products or services, as consumers take into consideration the public opinion before using or purchasing any service or product (Go et al., 2009).

Opinion mining has many applications varying from e-commerce and marketing to politics and other areas. For this analysis, NLP is used in order to find if there is any correlation between people’s opinions on twitter and fluctuations in the price of bitcoin.

There are many different algorithms using NLP to classify if an opinion is positive, negative or neutral. For this assessment, three algorithms are chosen and a comparison between them follows. The first algorithm uses Naive Bayes, the second one uses Convolutional Neural Networks and the third one a Random Forest classifier. The three algorithms are implemented with a dataset from Kaggle, that provides tweets about bitcoin and the algorithm that results in the best accuracy is chosen for the classification on recent tweets fetched via the Twitter API.

*Why Deep Learning?*

Are the received tweets positive, negative or neutral? This classification problem is somewhat different from more classical machine learning ones, in the fact that there is one input variable: the textual content. Although they have proved to be very efficient in many complex tasks, from image recognition to speech recognition, are Deep Neural Networks fit for this type of problem and this type of input data? What type of Deep Neural Network would be best suited for this task? If they are up to the task, what accuracy can they reach in our case?

Having very little prior knowledge of Deep Learning, we’ll start from a ready-made solution found on Kaggle that yields excellent prediction results. As this code is provided without any comment nor explanations, we’ll use academic research to make sense of this implementation. Thereby we’ll show why it provides a very convincing answer to the above questions.

### Natural Language Processing using naïve Bayes

The first method chosen for NLP uses a Naive Bayes classifier and a supervised Machine Learning based approach, where a classification model needs to be developed and trained using a pre-labeled dataset of positive, negative and neutral tweets about bitcoin. For text processing, the Python library Natural Language Toolkit (NLTK) is used, which provides tools needed for tokenization, parsing, classification, stemming, tagging, and semantic reasoning (Loper and Bird, 2002).

The steps followed for this approach are the following:

* Transform each review to a list of words by using tokenization.
* Remove the redundant words like stop words, remove punctuation and transform remaining words (e.g. lowercase them).
* Use the training dataset to assign a label to each group of words (positive, negative or neutral).
* Train a machine learning classifier in order to be able predict the polarity of future reviews.

According to Wang et al. (2003), the Bayesian classification used in this method classifies the content from a Statistical point of view and it follows Bayes’ Rule:

Since the denominator is independent from the class, we consider the denominator as a constant and, hence, we ignore that term. For the calculation of the numerator we write P where d is considered as a set of features and presents how often this class occurs in total. The classes for this analysis are positive, negative and neutral and we would be calculating the probability that any given review belongs to one of the three classes, without analyzing the current input document.

This simple (naive) method is used for classification and utilizes the “bag of words” representation. The bag of words model represents a sentence (a review in our case) as a group of distinct words, without considering the grammar or the word order. This is used in order to calculate the frequency of each word used as a feature for training the classifier.

For the calculation, two assumptions are made that make the process of calculating the classification probability easier.

* The bag of words assumption: the order of the words is not taken into account.
* Conditional Independence: The probabilities of the features are independent given the class c.

In sight of the above, if the assumption of independence of class features holds, the model can converge quicker than discriminative models, such as logistic regression. However, even though the conditional assumption doesn’t hold, as it is nearly impossible to find such data in real life, the method works great in practice. Not to mention, that this method is easy to understand, and the implementation can be fast.

### NLP with Convolutional Neural Network

According to Kim (2014), Convolutional Networks achieve excellent results for sentence classification and other NLP tasks on several benchmarks. They rely on vector representations of such sentences, as we will explain below.

In this context, the convolution is an operation applied to a window of n consecutive words that yields a new feature. This new feature is hence giving a condensed value of the n words in question. Just like for images, convolution allow to retain special information, in our case words sequence which is key in defining meaning.

To this new feature is applied a maximum pooling function whose role is to extract the most important features to get the right label in the training process, in other words identifying the n-grams that are most informative in the final classification, as pointed out by Jacovi et al (2018). The pooling operation also decreases the dimension of the data.

Kim compares results obtained with different topologies, layer dimensions and number of epochs. He concludes that using one layer of convolution yields remarkable results, on top of providing evidence that a correct word vector input is key to getting good results, as in a lot of NLP problems.

Intuitively, using a convolution makes a lot of sense: we’d rather consider words in a group, rather than words individually, when trying to get the meaning of a sentence, paying close attention to the words order. For example, “very satisfied” has got a very different meaning whether the preceding word is “am” or if it is “not”.

*Sentence vector representation for Neural Networks*

Neural networks only accept numeric values as input. In order to feed them with text, we could think of using a “bag of words”, i.e. representing this text as a word vector in a high dimension space. This is however not efficient computationally, as it creates sparse vectors and it not able to encode any information on the semantics of words.

On the contrary, word embedding representing word in a much lower dimension space, with dense vectors containing real numbers. Methods include co-occurrence matrix and probabilistic models, amongst others. They basically use the words in their context, i.e. the surrounding words, to measure how likely it is to find them together. The results are that vectors of words that have a similar meaning end up close together in this low dimension space, which shows that this encoding manages to capture words meaning. For example, the words “apple” and “pear” would be encoded with vectors close to each other, provided that the training text talks about fruits, not about IT. Applications of word embedding include automatic translation.

Word encoding can be made once and for all, on a very large text corpus, providing a static universal representation. However, according to Kim (2014), calibrating this encoding at the same time as training the Neural Network it feeds provides better results, since it uses the specific problem context. This is the approach we are using.

*Neural networks advantages (over other ML methods):*

* Neural networks can successfully model complex relationships between inputs, as proven with image recognition for example
* Long short-term memory layer enables to capture relationships between distant elements
* Dedicated hardware is available to meet their high computational needs

*Neural networks disadvantages (over other ML methods):*

* + As they have a lot of parameters to learn (weights and biases), they need a large amount of input data in the train process.
  + As a consequence of this, they require important computing power, to reduce training time
  + The training algorithm, usually gradient descent, can get stuck in local minima’s, there is not guarantee that a global minimum is reached
  + Also, due to this high number of parameters, they a prone to overfitting, as seen in this example
  + Hyper-parameters, such a number of layers, type and size of layers usually have to be adjusted manually, in a trial and error approach
  + They are considered as “black boxes”, i.e. results they deliver results that are difficult to explain (although a lot of efforts is currently put in “opening the black box” are make them more transparent).

### NLP with Random Forest

Random Forests are a scheme proposed by Leo Breiman in 2001 (Breiman, L., 2001) for building a predictor ensemble with a set of decision trees that grow in randomly selected subspaces of data. Breiman demonstrated that as an ensemble method relies on combining the results of an ensemble of simpler estimators. That means that final predictions are obtained by averaging results of multiple decision trees. Thus, train set is divided into smaller samples and each of trees become independent tree. This aggregation of tree classifiers is known as a forest, as each of these trees is constructed using an injection of randomness. Hence, random forest classifier provides two types of randomness. One that comes from random sampling of training data points when building trees, second with respect to subset of features that are being considered when splitting nodes (Biau, G., 2012)

*Random sampling of training observations:*

Random Forest classifier uses the concept of Bagging and Bootstrapping. Bootstrapping draws the samples of data points with replacement. That means training each tree in a random forest on different samples multiple times. This is done in order to lower the overall variance, but not at the cost of increasing the bias. On the other hand, bagging, known as bootstrap aggregation, allows to increase accuracy in an algorithm by prevention of overfitting and lower the relatively high variance in a single tree. This method builds various instances of a black-box estimator on random subset or training set and combine their own predictions to a formal prediction.

Input for Random Forest includes number of trees, training data, number of total features and subset of features. For each tree in a forest, bootstrap sample and size of training data has to be chosen. In order to create the tree, we choose the subset of features at random from total features, then select the best among subset of features and split the node. When a tree is created, test sample will be passed to each tree and class label will be allocated based on majority of votes (Parmar, H. et al., 2016).

Two main features of Random Forest include robust and accuracy. Random Forest in considered to be robust in terms of noise, because it uses Bootstrapping, so each tree works on the subset of training data and is trained on different value. Also, it is accurate classifier because of the concept of bagging, so the output of all decision classifier is averaged (Parmar, H. et al., 2016).

Due to a fact that Random Forest is an aggregation of decision trees it deals with many hyperparameters, that either help to increase predictive power or make a model faster. They have their own importance and influence on the output prediction. Firstly, “n\_estimators” which is the number of trees that algorithm builds. The higher number of trees makes the prediction more stable but decrease the computation. Secondly, maximum number of features “max\_features” that are considered to split a node. The last one is “min\_sample\_leaf”, which determines minimum number of leaves that are needed to split an internal node.

Random Forest is popular algorithm suitable for supervised learning for both classification and regression. In a classification problem, the most popular class in each tree is chosen to be a final result. In a regression problem, the average of all the tree outputs reflect the final result (Koehersen, W., 2018). Random Forest is highly versatile machine learning method with numerous applications ranging from marketing to medical diagnosis. Also, it is fast and easy implementation, with highly accurate prediction using sentiment analysis.

The main advantage of Random Forest is the fact that it can be used both for classification and regression problem. Due to the fact that it takes an average of all the predictions, the variance decreases and there is low possibility of overfitting problem. Also, it is perceived as a highly accurate and robust method, because of the number of decision trees participating in the process. A primary disadvantage is the difficultness to interpret the results comparing to decision tree, where decision can be followed in the path of tree. Thus, Random Forest is a predictive not descriptive tool. It is difficult to understand relationship between response and independent variables. Also, the process is time consuming, as to make a prediction, all the trees in the forest have to make a prediction for a given input. In case of small dataset, random forest can be prone to overfitting (Navlani, A., 2018).

## Algorithms comparison

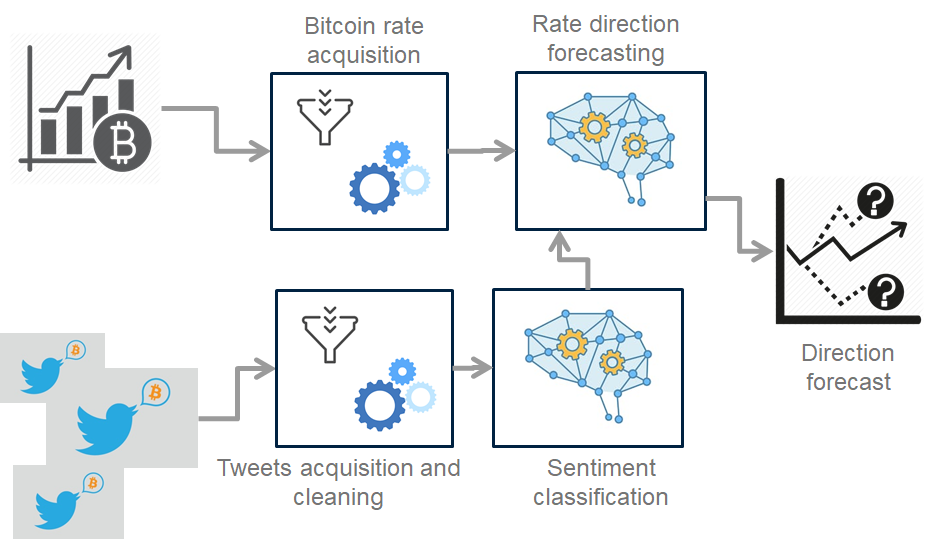
In order to choose the algorithm to implement in the definitive solution for sentiment analysis, we are looking at their results, looking not only their respective accuracy but also at other strength and weaknesses. Overall accuracy reached by each of these algorithms is high, to a level that surprised us. We can say that above 90% is satisfactory since, if done by human being, the tasks would not reach 100%, as there would be differences in interpretations. That threshold has been reached by all 3 of them.

Under those conditions, we check if the top performing algorithm, i.e. Neural Networks, presents any major drawback that would hinder its implementation. Although more complex conceptually and in terms of implementation, as its training time is acceptable and we have no need to justify classification of the tweets in detail (i.e. explain results), we come to the conclusion that this is the best solution in our context and select it as our definitive classification method. On top of this, we were lucky to find a training dataset with enough data to successfully train the Neural Network.

# Method

## Solution design / architecture

From the problem description, a design was elaborated to decompose the solution into independent modules with clear roles and interfaces, enabling to decrease the overall complexity and risks. This architecture was chosen to be aligned with the selected Agile approach, as it would allow to work in parallel on these different module.



The modules defined are:

1) **Bitcoin rate acquisition**

Role: Import bitcoin rates in the expected time granularity.

2) **Tweets acquisition and cleaning**

Role: Import Tweets related to Bitcoins in the right language. Prepare and clean them to be processed by the rest of the chain.

3) **Sentiment classification**

Role: classify Tweets according to their content, into positive or neutral.

4) **Rate direction forecasting**

Role: using the classified tweets and bitcoin rate, forecast next price movement, either up or down.

5) **Graphical dashboard**

Role: display the results in a clear and synthetic way to the user

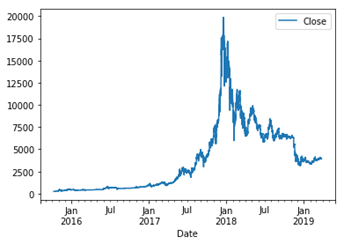
## Data Collection

### Bitcoin price data

*Autoregressive Integrated Moving Average (ARIMA) To Predict the Log Return:*

The data for bitcoin is downloaded from[Gemini](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://www.cryptodatadownload.com/%26amp;sa%3DD%26amp;ust%3D1555195900534000&sa=D&ust=1555195900547000&usg=AFQjCNF6KfC4VYfX_Qn817tDAzXToD92Cg) (Free Cryptocurrency Time Series Data, 2019) bitcoin exchange website service for the time period of Oct 2013 to Mar 2019, with hour to hour updates of Open, High, Low, Close price (OHLC), trading Volume in BTC, and Timestamps as Unix time format. The purpose here is to predict the log return of the next hour using ARIMA. Using return not prices is helpful for normalization and measuring all variables in a comparable metric. In addition, using the log return has also several practical and theoretic benefits such as *time-additivity.*

Time Series data has some characteristics Trend, Seasonality, Cyclicity and Irregularity. It is not easy to tell only by looking at the plot what characteristics the time series has. Fortunately, there is a way to decompose the four mentioned traits. It provides an abstract model for thinking about time series, problem and most importantly the best way to forecast. The data (prices) in the plot below doesn’t look stationary which means the series are not normally distributed and the mean and variance are not constant over a long time period. There is a hypothesis test of the unit root test called Dickey–Fuller checks the status of the data.



*Figure 1 Bitcoin daily price distribution*

Further, ARIMA model is made to deal with non-stationarity. It assumes that the data after differencing becomes stationary.

According to (Brockwell, 2002),ARIMA has three parameters that need to be tuned in order to get the best and optimized forecast:

* **AR**: Autoregression. A model that uses the dependent relationship and correlation between an observation and lagged observations.
* **I**: Integrated. The use of differencing; meaning subtracting an observation from an observation at the previous time step. This parameter is used in case of non-stationarity.
* **MA**: Moving Average. A model that uses the correlation between observation and residual errors from when the moving average technique used on lagged observations.

A standard notation for these parameters is ARIMA(p,d,q) where the parameters are substituted with integer values:

**p**: or lag order which is the number of lag observations.

**d**: or differencing is the number of times that the observations are differenced.

**q**:  or order of moving average is the size of the moving average window.

## Gathering tweets

### Acquiring tweets using twitter streaming API

*What is an API?*

API is technically an Application Programming Interface, which is a part of server that communicates with web browsers taking in the requests and giving back responses. The APIs are different in terms of the format of their response like HTML, JSON, XML, text etc. Most companies build APIs for their customers or for internal use. For example, while using Facebook application we send requests to their remote servers where it is interpreted and the data, as response, is returned in a readable format. This is done via API at server level.

*REST vs Streaming API*

REST API Design was defined by Dr. Roy Fielding in his 2000 doctorate dissertation. REST or RESTful (Representational State Transfer) API is architecture style, designed for distributed hyper-media systems. REST API is often referred as a web service as it uses HTTP requests to GET, POST, PUT or DELETE data. REST works in a request-response pattern i.e. client send an individual request for information and receives the data as a response from server. After the response is received, the connection closes only to be re-opened for another request.

A screenshot of a cell phone

Description automatically generated

Streaming API is different in respect of the response pattern. It builds a persistent connection to the server that continuously responds with updated data until the connection is eventually terminated. Streaming APIs are perfect when a user needs to consume a constant flow of rapidly updating live data.

A screenshot of a cell phone

Description automatically generated

*Why we used Twitter Streaming API*

We collected a huge number of tweets filtered by keywords and hash tags – Bitcoin, BTC to train our system for sentiment analysis using Twitter’s Streaming API and Tweepy library in Python. The Streaming API allows to consume large data persistently. We can also filter and get only the tweets we need by using Twitter library with Twitter’s real-time streaming API.

*Connecting to the Streaming API*

To work with Twitter’s API, we created a developer’s account and an App. We then set up an environment in the App to get the access token and secret keys to go through the authentication steps required for connection with an API.

*Authentication*

Twitter Streaming API uses OAuth authentication method. The request was first authorized according OAuth specifications using OAuth Handler and the keys in Python.

*Consuming Tweets*

To connect to the Streaming API, an HTTP request was formed, and filtered stream of Tweets was consumed in JSON format continuously for 7 days. The streaming data was encoded in utf-8 and saved as CSV.

A screenshot of a social media post

Description automatically generated

Figure 1Function that gathers and filters the stream of Tweets

## Sentiment analysis process

### Reducing noise in the twitter dataset

Real-world data, which is the input of the Data Mining algorithms, are affected by several components; among them, the presence of noise is a key factor (Wang, et al., 1995). The accuracy of the classifier built for sentiment analysis highly depends on the quality of training data. Hence, classification problems such as sentiment analysis become very complex when the data contains noise.

Noisy text cannot be categorized properly by text mining software programs as it does not comply the rules used to identify and categorize words, phrases and clauses. Idiomatic phrases, acronyms, abbreviations, business-specific languages, hashtags, URLs and emojis are all considered as noise in text. Other significant problems are caused by punctuations, wrong spellings and typographical errors. All of these are prevalent in unstructured text gathered from social media, blogs or electronic conversation. In order to get rid of above-mentioned noises in Tweets, a script was written in Python using the following strategy.

The text was first converted to lower and all the white spaces and punctuations were removed. The text was then scrubbed from URLs using regular expressions. Next, it was parsed through a Python Spell Checker. All the mis-spelled words were replaced by correct ones from dictionary except the relevant influential words that cannot be found in dictionary such as “Bitcoin, BTC, PayPal, ether, Cryptocurrency” etc. After spell check we removed all the stop words from Tweets. Stop words does not have a relevant significance on sentiment analysis, example, articles and pronouns (Alfano, et al., 2015).

*Text Normalization – Stemming and Lemmatization*

In Natural Language Processing (NLP), we want to boost our algorithm’s performance by reducing the size of dictionary and making the training data dense with less dimensions. This can be achieved by Text Normalization. Stemming and Lemmatization are Text Normalization or Word Normalization techniques.

For grammatical reason, text contains different forms of word such as organize, organizes, organizing. Additionally, there are families of derivational words that are used for similar meaning in the text. These are called inflectional forms of a word. Text Normalization reduces the inflectional forms to the base form of the word known as lemma. For text normalization of Twitter data, we used Lemmatization with Python Natural Language Tool Kit (NLTK) Package.

*Why we used Lemmatization?*

Stemming is definitely simpler than Lemmatization because it uses Suffix Stripping to generate stems. The Stemming Algorithms are rules-based rather than linguistic i.e. it uses a set of phased rules to cut off the suffix or prefix from the words’ stem. Stem may or may not be a dictionary word, it is only a smaller or an equal form of the word. Therefore, stemming is often not efficient and generate words that are not in dictionary. But it is known for its simplicity and speed as it does not use a look up table and follow a simple algorithm, based on rules.

Lemmatization on the other hand, uses morphological approach to find the lemma of each word i.e. it looks up in the dictionary to find the root of the word. This root is called lemma which is its dictionary or canonical form. This requires a detailed dictionary of the language which is provided by NLTK Package and provides better results than stemming.

However, to get the best results, we had to feed a few tags to the lemmatizer, otherwise, it won’t reduce or change some words as we desire. People often shorten the words in tweets or conversational text which is not understandable by classifier and machine learning algorithms. For example, awsm for awesome, or luv for love.

The last step was to make sure all the tweets were unique and remove repetition of same words. After the cleaning process, the data was saved as CSV and sent as input to the Sentiment Analysis Algorithm.

### Individual tweet sentiment analysis

Full documentation of codes for each algorithm presented in a project is enclosed on the GitHub website: https://github.com/C3st0/CryptoCurrenciesSocialMedia

# Model

## Recurrent Neural Network and Long Short-Term Memory

Long Short-Term Memory networks (“LSTMs”) are a special type of Recurrent Neural Networks that is not only capable of processing sequences of inputs but also of learning long-term dependencies from these inputs. Here, the principal motivation to use a recurrent network is to exploit syntax in order to learn dependencies between words in a sentence.

Technically, the idea is to let every step of an RNN pick information to look at from some larger collection of information, i.e. previous inputs. In our case, this is memorizing previous words, since they can be conveying meaningful information. Actually, at this stage they are “encoded convoluted” words, since they went through the embedding and convolution layers previously. So, this memory capability in the Neural Networks enables determining the sentence meaning according to patterns in a sequence that can be as long as necessary, which has to be more efficient than looking at patterns individually.

*Dense layer*

The final layer is a fully connected layer, that is quite similar to the studied perceptron model in Alessandro’s module. Its role is to set the needed weights and biases to produce the expected outputs, i.e. one of 3 possible values used in the labelling (0 = “negative, 1 = “Neutral”, 2 = “Positive”).

*Development and execution environment*

In order to use a powerful environment that would treat the large volume of input data in an acceptable time, we used Google Collab environment, that is a cloud-based Jupiter notebook environment. It offers support to execute code with 8 core CPU’s with Nvidia Tesla K80 GPU, that are amongst the most efficient hardware for this purpose.

# Results

## Data Collection

### Bitcoin data-set

There are two ways to find these parameters, the first one is through autocorrelation factor (ACF) and partial autocorrelation factor (PACF) function. From the resulted plots, p and q can be deduce from the number of spikes outside the significance area. If there is more than 5% of the lags are outside the significance zone, then we assume that there is enough information available in the residuals. Plots of ACF and PACF of the data below.

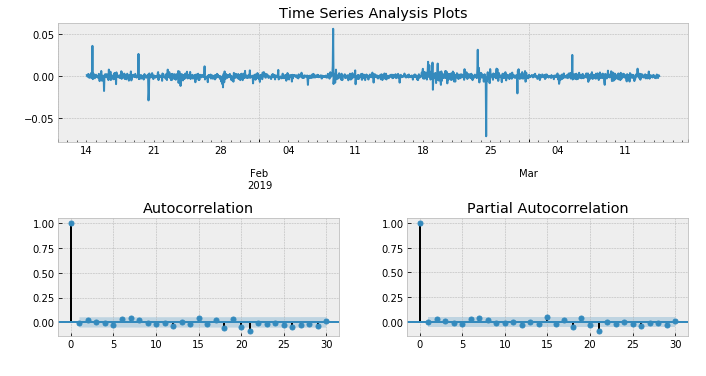


Figure 2Time Series analysis plots

The second way and the one applied in this project is by using auto.arima function which iteratively explores different combinations of (p,d,q) to find the best model that yields the lowest Akaike Information Criterion (AIC) value. AIC measures the amount of information lost after using a statistical model and the lower information lost, the better. Ljung-Box checks the if the model ARIMA(3,0,2) is good fit or not. This test yields a p-value which should be significantly larger than 0.05 and then the model has a strong evidence for having white noise residuals.

Now the predicted values of the log return for the next 168 (next 7 days) hours with two confidence intervals at 95% and 99% significance level.

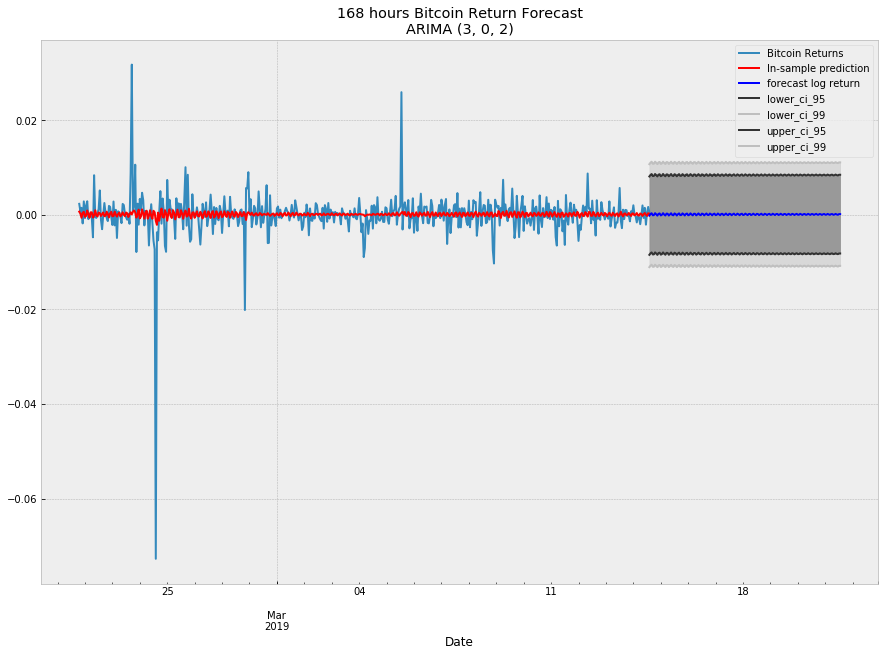


Figure 3 168 hours Bitcoin Return Forecast

Evaluation of the model:

There are some approaches to calculate the error between the forecasted values and the actual values such as Mean Absolute Error, Mean Squared Error, Root Mean Squared Error. The other approach can be done here is to calculate the percentage of the correct prediction (confusion matrix). 70% of the results are correct.



Figure 4 Prediction and Percentage of accuracy

It would have been efficient if there was a different model such as naïve to use it as a benchmark to compare the accuracy of the results.

## Twitter data-set

### Results Natural Language Processing using naïve Bayes

To begin with, accuracy testing is run several times, each time splitting the dataset into a training and testing part with a different ratio. Due to the fact that the amount of positive, negative and neutral tweets is different, each one of those classes is split individually:

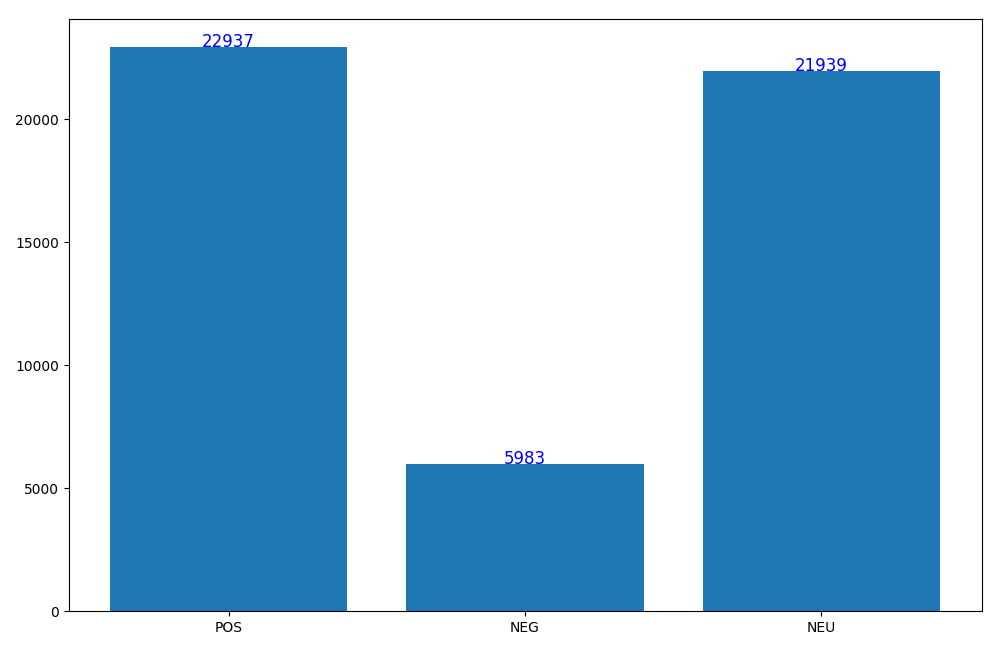


Figure 5 Classification split

The following graph shows the accuracy score for each of those runs:

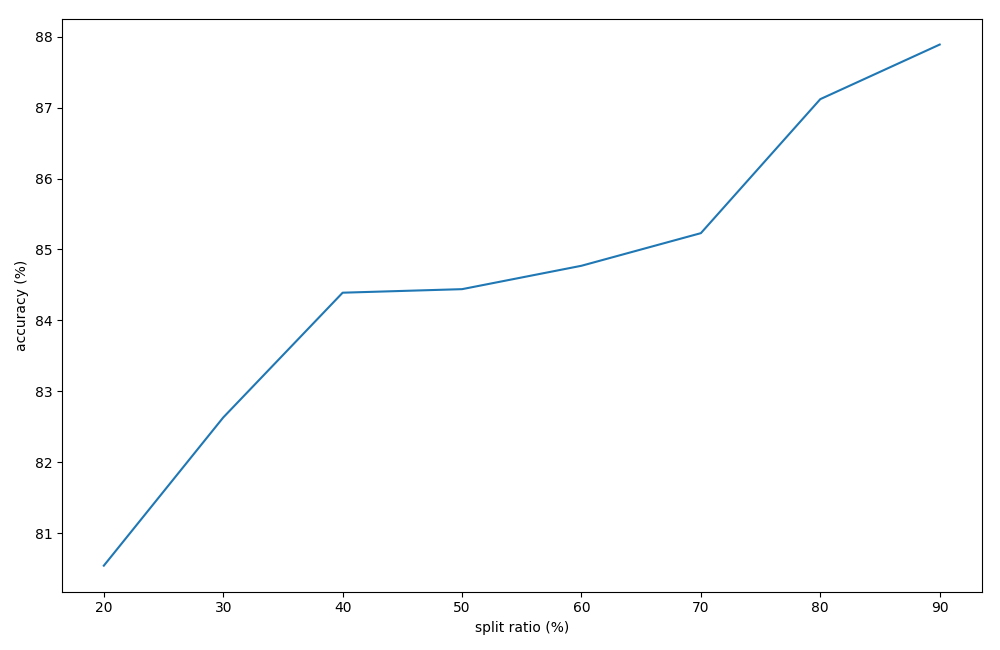
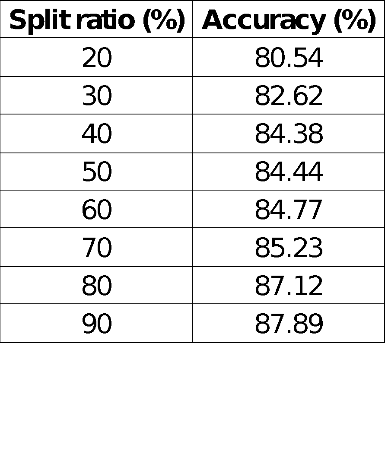


Figure 6 Accuracy score for each run

One of the benefits of using the Naive Bayes classifier is that it doesn’t require a large training set. That explains the fact that there is no dramatic change in the accuracy between 20-80% split of the data. However, because the highest accuracy score is approximately 88%, which is not satisfactory, other methods will be used in order to get a better result.

Following that, a confusion matrix is constructed in order to describe the performance of the classification model. The role of a confusion matrix or error matrix is to visualize the performance of the algorithm. It makes possible the recognition of confusion between

classes, in this case positive, negative and neutral tweets.

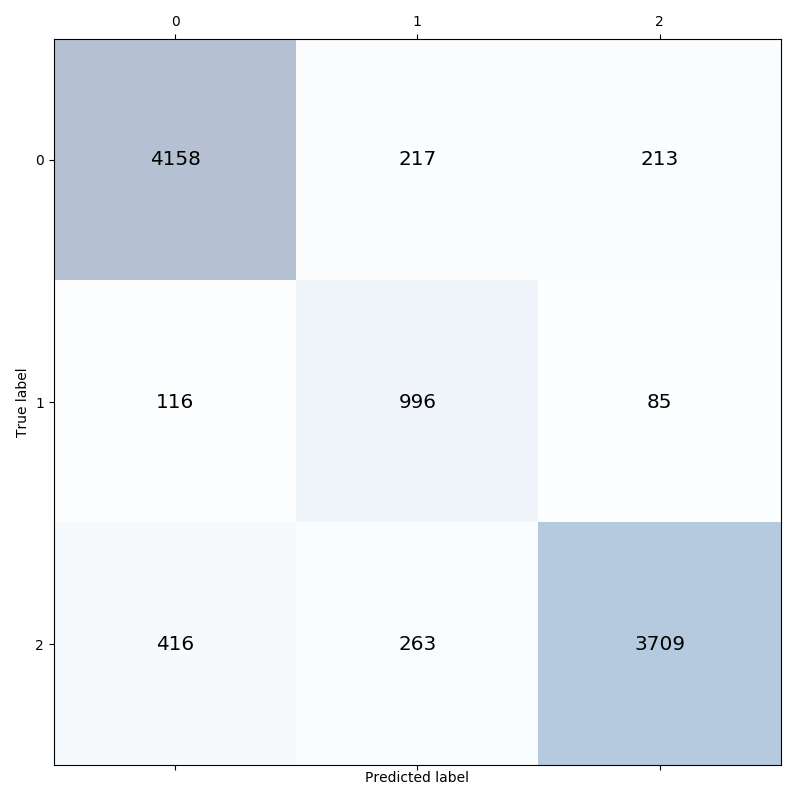


Figure 7 Confusion matrix

The matrix shows that the results are not as satisfying as expected, as there were many mislabeled tweets.

### Results for NLP with Convolutional Neural Network

Based on the above consideration, we build the complete model architecture as illustrated below. We initialize the model with Keras’ Sequential layer and added the embedding layer as the input layer. By using the embedding layer, the “bag of words” vector is turned into a dense vector of fixed size and this new representation is passed to the CNN layer. Each filter in the CNN will detect specific features or patterns and then it will be pooled to a smaller dimension in the max-pooling layer. These features are then passed into a single LSTM layer of 100 units. Then, the LSTM outputs are then fed to a Fully Connected Layer (FCL) which is built using Keras’s Dense layer. As there are three labels to be predicted, a SoftMax activation function is used at the output layer.

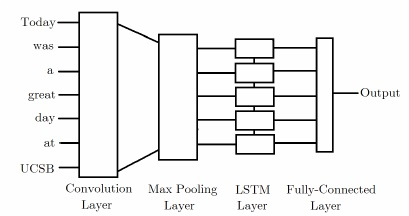


Figure 8 Sketch of Neural Networks

The actual solution uses 2 convolution-pooling layers, aa this proves to provide better results than just one.

Layer (type)                 Output Shape              Param #

=================================================================

embedding\_1 (Embedding)      (None, 30, 100)           2000000

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv1d\_1 (Conv1D)            (None, 30, 32)            9632

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling1d\_1 (MaxPooling1 (None, 15, 32)            0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv1d\_2 (Conv1D)            (None, 15, 32)            3104

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling1d\_2 (MaxPooling1 (None, 7, 32)             0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

lstm\_1 (LSTM)                (None, 100)               53200

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_1 (Dense)              (None, 3)                 303

=================================================================

Total params: 2,066,239

Trainable params: 2,066,239

Non-trainable params: 0

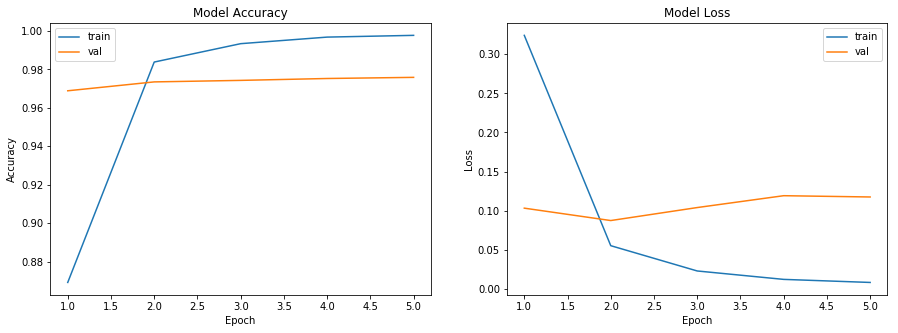


Figure 9 Model accuracy and overfitting

The above graphs show the overall model accuracy. We can see it quickly reaches a very high accuracy on the training set but quickly goes into overfitting. On the validation set, the accuracy is remarkably good (96.7%) but does not improve much with the number of epochs. The loss function is even increasing, which hints at using a smaller number of epochs (2 would be sufficient).



Figure 10 Confusion matrix

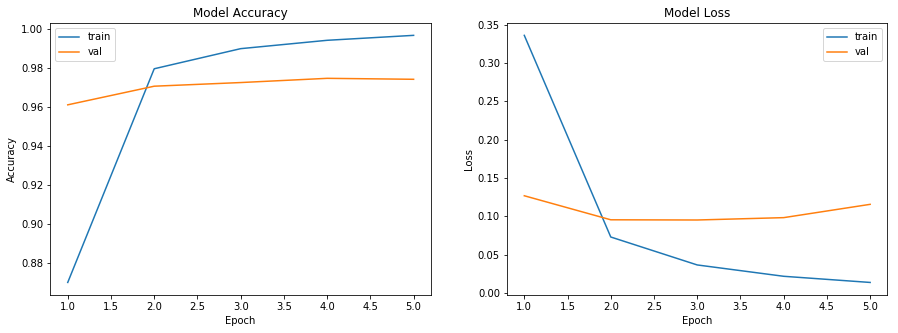
The confusion matrix tells us that the overall results quality is very good, as few tweets are mislabeled.

Topology comparison

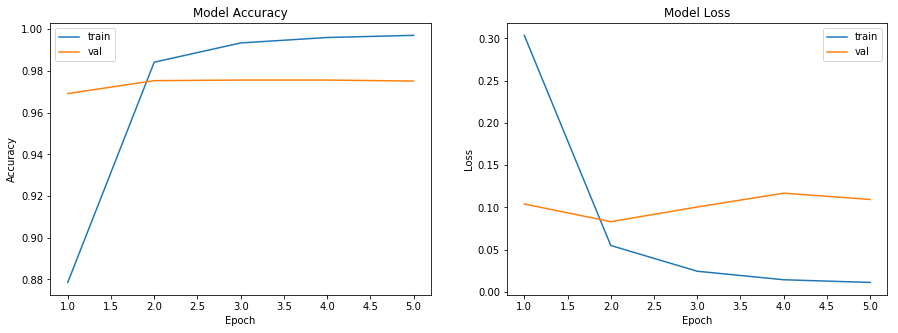
In order to confirm our understanding of the neural network behaviour and quantify the impact of each layer, we tested different topologies and compares them: removing one convolution layer, removing both, removing the Long Short Term Memory layer, adding an extra dense layer of 100 neurons. We also checked if we could be the original model, building a new one that performs better, bad enlarging the memory size in the LSTM layer.

Model accuracy and confusion matrices for networks provide the following results:

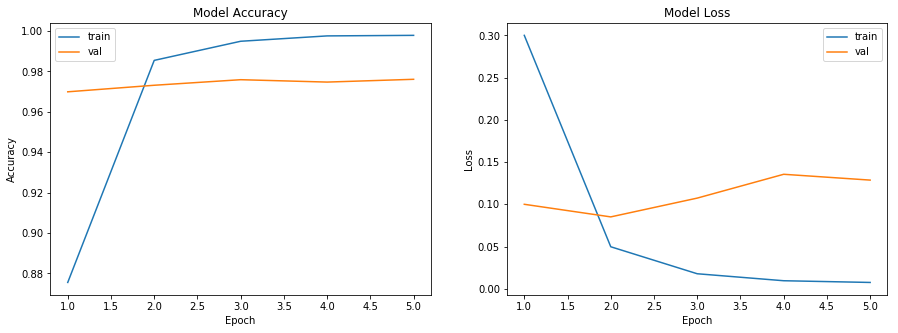
1) no convolution : Accuracy = 97.4%



2) no LSTM : Accuracy = 90.2%



3) Added dense layer : Accuracy = 97.4%



We draw the following conclusion:

1) The Long Short Term Layer plays a major role in the performance: without it, the performance drops by about 10%. Increasing the memory size does not improve the results, however, showing that only a subset of terms play can really influence the classification (which makes sense to us).

2) The two convolution layers play a marginal role here. But they can make the difference between a very good result and a top one (in the Kaggle sense), as they allow to improve the accuracy by 0.2%: this top 0.2% that is the hardest to reach. Additionally, removing them make the training time longer, as the operation of dimension reduction they implemented is now missing.

3) Adding one more dense layer does not improve accuracy on the testing set but it does on the training set, which is a sign of overfitting.

We come conclusion that we can’t find a topology that brings significant improvement over the original one. However tests confirm our understanding of neural network configuration and the fact is it very well suited to the problem in question.

#### *Hyper parameters*

The model hyper parameters have not been modified much. With more time, one could think of trying to optimize those parameters: size of layers, size of layers, activation functions for convolutions, size of pooling, etc. This could lead to some accuracy improvements, applying a trial and error strategy.

Worth trying would be to introduce a dropout layer, in order to reduce overfitting. There as well, drop out percentage would need to be tweaked manually to reach an improvement.

#### *Conclusion*

In a few lines of code, a complex Deep Neural Network has been built, that involves several advanced concepts. The result is for us very convincing. As far as Machine Learning is concerned this is very satisfactory: the model can learn with very high accuracy and it generalizes enough so it can be applied to new data with an accuracy this is still very high.

It is questionable if such an accuracy is due to some luck or some improbable uniformity in data set, where the training and validation datasets would have exceptional similarities. With more time, we would like to clarify this and make sure we can trust these numbers.

### Results for NLP with Random Forest

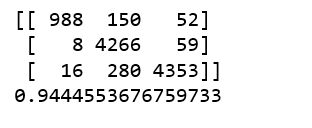


Figure 11 Confusion Matrix

The model accuracy (94%) is very high, but this might suggest overfitting. This is not the best result out of three methods used, so it will not be proceed in further analysis. The confusion matrix shows identification of confusion between labels of tweets. Overall result is relatively good, as just few tweets are mislabelled.

## Prediction performance

### Multivariate Time Series Forecasting with LSTMs

*Why Long Short-Term Memory in Time Series?*

A time series can be seen as a data signal where each data is a succession of the previous observations. The main objective is to predict the future based on the past and the properties of such time series. LSTMs allows predicting the future values of a time series using multiple time series.

In order to understand the concept of Long Medium- and Short-term trend, the figure below shows the decomposition of the signal three components. Where it can be seen that the long term, is the baseline that adds an offset to the signal. The frequency range of the long-term components (low frequencies) are between https://www.google.com/chart?cht=tx&chf=bg,s,FFFFFF00&chco=000000&chl=0%5Cleq%7B%7Df%3C%5Cfrac%7B%5Cpi%7B%7D%7D%7B4%7D  while the medium components are between https://www.google.com/chart?cht=tx&chf=bg,s,FFFFFF00&chco=000000&chl=%5Cfrac%7B%5Cpi%7B%7D%7D%7B4%7D%5Cleq%7B%7Df%3C%5Cfrac%7B3%5Cpi%7B%7D%7D%7B4%7D finally the short term is associated to rapid changes in the series and their components are in the range of high frequencies https://www.google.com/chart?cht=tx&chf=bg,s,FFFFFF00&chco=000000&chl=%5Cfrac%7B3%5Cpi%7B%7D%7D%7B4%7D%5Cleq%7B%7Df%3C%5Cfrac%7B%5Cpi%7B%7D%7D%7B%5C+%7D.

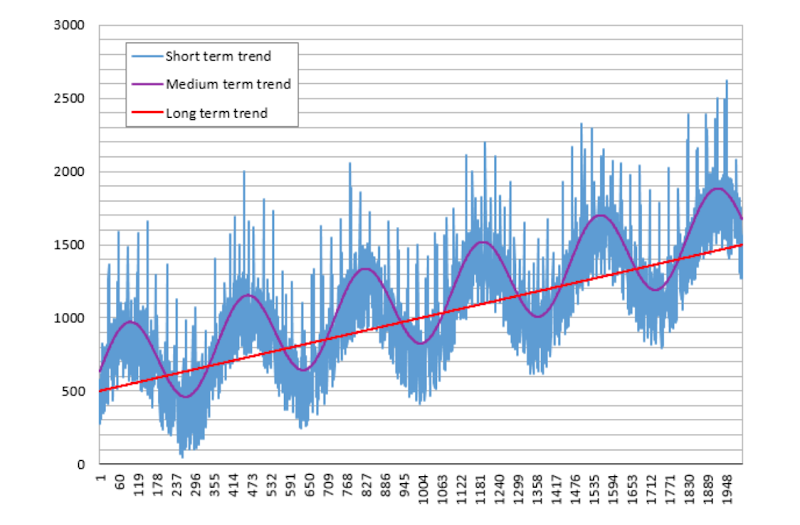


Figure 12 Trend of Bitcoin

In our case, as the bitcoin data is sampled with a period of 1-hour According to Nyquist

(Aliasing Bruno (2000)), the maximum frequency component (https://www.google.com/chart?cht=tx&chf=bg,s,FFFFFF00&chco=000000&chl=%5Cpi%7B%7D) that we can appreciate

will be https://www.google.com/chart?cht=tx&chf=bg,s,FFFFFF00&chco=000000&chl=%7Bf%7D_%7Banalog%7D%3D%5C+%5Cfrac%7B1%7D%7B2%5C+x%5C+%283600sec%29%7D that is 2 hours. This means that any change that is higher

than this max component will not be appreciated.

LSTM Networks

Long Short Term Memory networks are a particular case of recurrent neural networks and they hey were introduced by [Hochreiter & Schmidhuber (1997)](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttp://www.bioinf.jku.at/publications/older/2604.pdf%26amp;sa%3DD%26amp;ust%3D1555201298457000&sa=D&ust=1555201298474000&usg=AFQjCNGkdu2ollTmn7USrU0jPpA82PB5AQ).

This type of neural network was created to deal with long-term dependencies problems where they can retain information for long periods of time.

The main difference with a standard neural network is that its LSTM can be considered a complex system with 4 neural networks where each NN has a different role. The figure below shows the task of each network.

Starting from the bottom:

The first NN will predict give as output a range of possible predicted values without considering the internal memory. However, it has a component of memory due it receives the new information concatenated with the prediction (all the NN of this system receives this same input).

Then the second NN “ignoring” is connected to the main branch with a multiplier. The function of this NN is to remove possible values that are impossible to happen. To achieve this, the output of this neural network will give a vector with 0 and 1. This output vector is multiplied with the possibilities (the output of the first NN), therefore every value equal to  zero of the ignoring NN output  will remove the possible value that the first NN block gave as output.

 The vector with the remaining possible values after this multiplication will be added to the internal memory vector. Which was calculated by the forgetting NN.

Finally, the selection NN will choose one result from the collected possibilities and this result will be the prediction that will be also part of the next input.

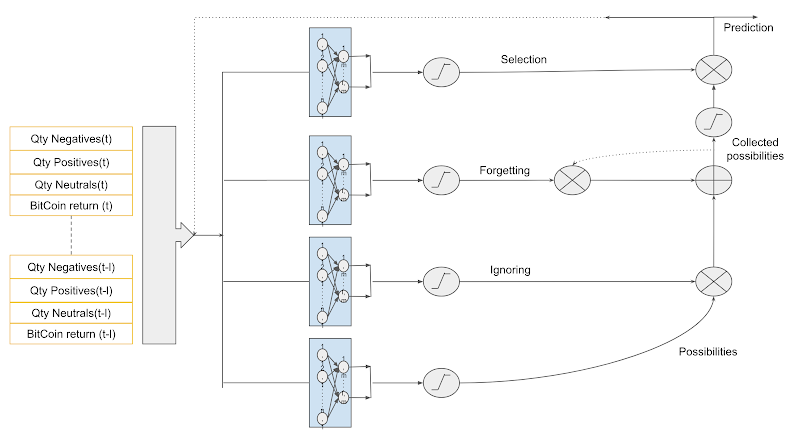
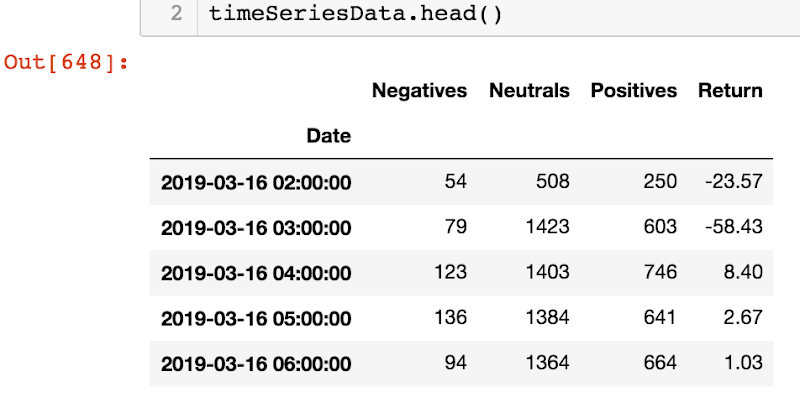


Figure 13 The repeating module in an LSTM contains four interacting layers

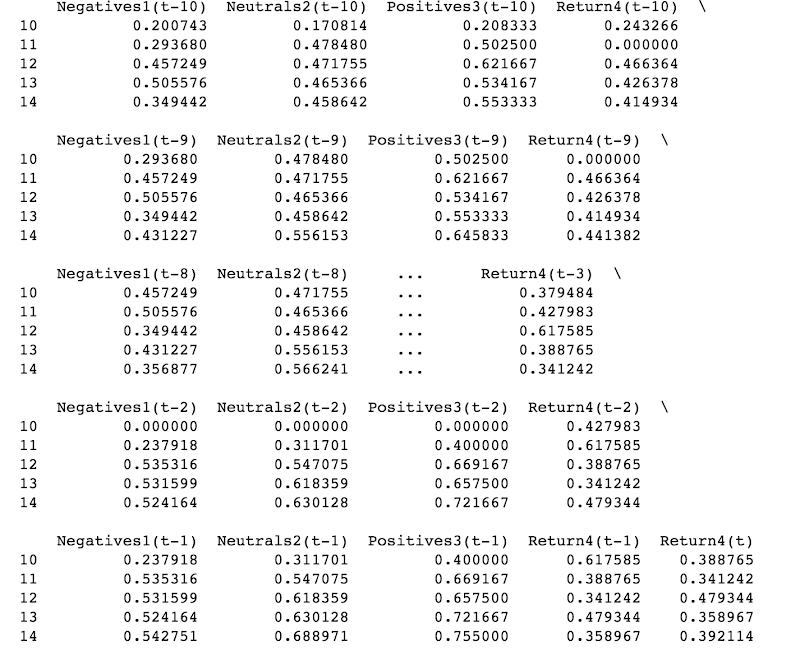
## The prediction model implementation

This implementation will lead with four time-series concatenated in the same matrix. Three of these four time-series were generated by the output of the sentimental analysis (the series is Qty Negatives(t), Qty Positives(t), Neutral(t)). Due that the period of bitcoins is one record every one hour, the output of the twitter sentiment analysis was grouped as total by hours.

On the other hand, for the Bitcoin, we transformed the close values of each other to the return https://www.google.com/chart?cht=tx&chf=bg,s,FFFFFF00&chco=000000&chl=%7Br%7D_%7Bt%7D%3D%5Cfrac%7B%5C+%7BP%7D_%7Bt%7D-%7BP%7D_%7Bt-1%7D%7D%7B%7BP%7D_%7Bt-1%7D%7D because this signal is stationary, and the stock values are not. It is better to work with a stationary series in stochastic signal processing.



In order to prepare the input of the data in the NN, it was necessary to build a matrix where each row has the data in the time t with all the delays. In the figure below the time windows length, L is equal to 10.



# Discussion

## Prediction model

*Results:*

Unfortunately, the results did not present a good performance between the actual values. The correlation between the predicted value (y Hat) and the Actual bitcoin trend was near Zero.



The scatter plot of predicted value (y Hat) and the Actual bitcoin trend.

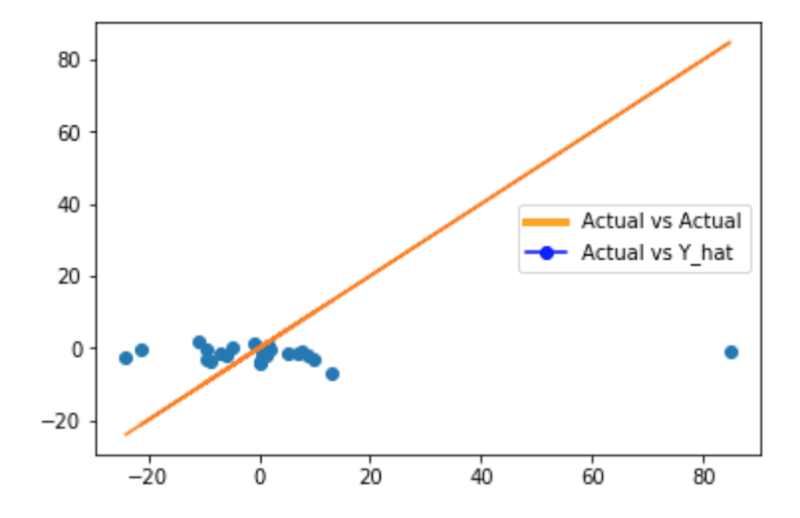


Figure 14 The correlation between the predicted value (y Hat) and the Actual bitcoin trend

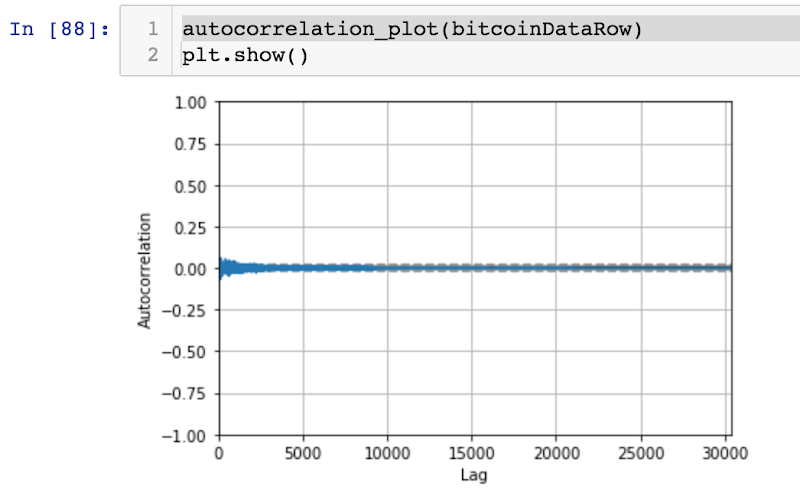
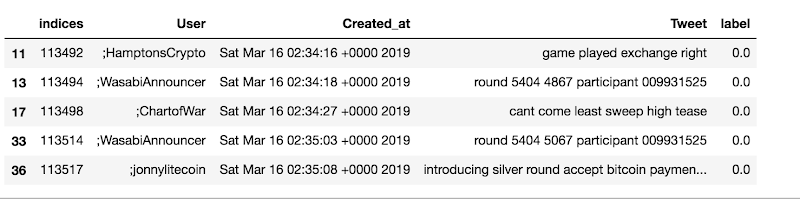


Figure 15 Autocorrelation plot

The figure 12 illustrates with the blue dots the scatter plot between the actual values (horizontal axis) and the predicted values (vertical axis). The orange line is the actual values vs actual values line. It is worth noting that the value predicted is the mean of the returns (equal to zero). We can assume that this prediction was because the neural network did not have accurate inputs to predict the values. The figure 14 illustrates the inputs of the LSTM of the estimated positive and negatives tweets. On the other hand, as it can be appreciated in the figure 14, the bitcoin has an autocorrelation equal to zero for every lag. this means (L. Cohen 2002) that it might be very hard to predict the trend of future values using only this time series (this was why it was added to the analysis of the time series of twitters messages). We were expecting to improve the performance with this time series. Before showing the results that are part of the input of the final prediction, it is worth explaining that the performance of 96% shown, was using a dataset download form Kaggle that seem to be cleaned of spam and different sources of noise.  On the other hand, the results we show in the next two figures, belong the output of the same algorithm but using data download from the API twitter.  The first figure below shows an example of the first 5 values of the sentimental analysis result with negative prediction (label = 0). It can be appreciated that the results not easy to predict as negative (even labing manually by human beings). In the same way, the next figure shows some “positive results” (label = 2), where again, it is not clear if those messages are positive.



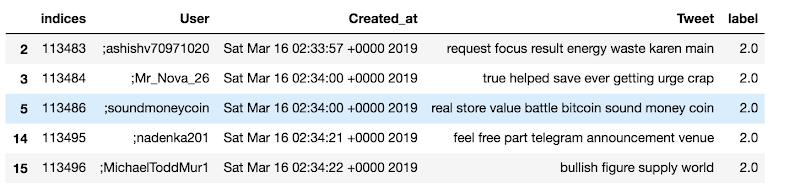


Figure 16 Tweets classified as negative (labeled as 0) and positive (labeled as 2)

# Project outcome and reflexions

**Big data**

Working with big data is time consuming and puts constraints not only in storage solutions for the large data volumes, but also on bandwidth and algorithm performance. This is even more true when working as a team in different locations, where data has to be moved around from one person to another many times.

In order to have more efficient collaboration, a dedicated workspace with high bandwidth connection would be needed.

**Results quality and reflexions**

As demonstrated, the predictive power of our solution is close to none. This is in contradiction to several previous studies that showed the complete opposite. There are several reasons to be investigated for this, in our view.

Based on our findings, the input data from Twitter is of bad quality, i.e. it contains an important volume of “spam”: messages that don’t convey any useful information but aim at doing advertisement and when they not just scams. Many other messages contain only web links or no clear positive nor negative message, even for human to interpret. Our ML classifier confirms this. One should note that Twitter has been infested by bots, posting automatic messages, that could not be shut down completely despite important efforts by the firm, which would explains the amount of spam. This is rather recent and it is why previous studies did not face this (or at least in the same extent). Here is for the technical explanation we can provide.

This raises the question if the mentioned studies, stating Twitter could predict bitcoin rates accurately, might have created the problem: it would make sense that people would try to use this principle at their advantage, so they would flood Twitter with information they made up in order to manipulate the rate in the direction they were expecting. This would explain the bots, or at least the important volume of “infomercial” messages.

From a “business” point of view, two phenomenons were at play on top of this:

1) Bitcoin lost in interest: after reaching highs in December 2017, its rate dropped and never reach the same levels. This is either a cause or a result of the clear decreasing interest momentum in Bitcoin, possibly to the profit of other cryptocurrencies.



Figure 16. Bitcoin Price in $ chart 2010-2019 (monthly rate).

2) Twitter lost in popularity: in February 2019, Twitter had its lowest number of users in two years : 321 million monthly users, down from 326 million the prior quarter (The Verge, 2019). At the same time, other social networks are gaining importance, new ones are emerging. So using Twitter as the sole source to measure trends may not be as reliable as it used to be, as it may not be as representative for user segments. It could that younger people are using more instagram for example (ranking provided below).

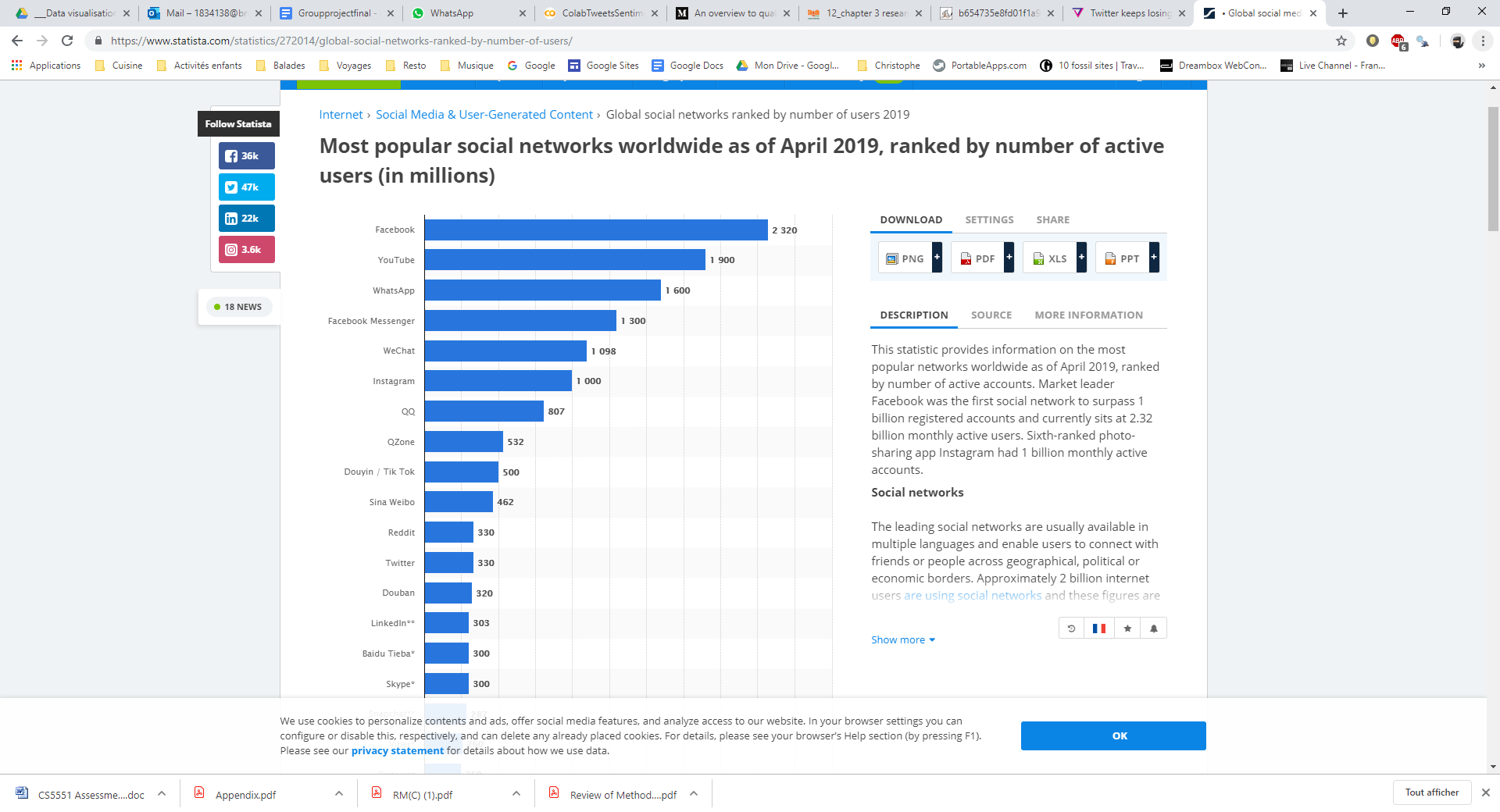


Figure 17 Most popular social networks worldwide as of April 2019, ranked by number of active users (in millions)

All these elements added together really put in question the fact that Twitter can be predictive at all, even if all spam was removed. There seems to be less people interested in Bitcoins, less Twitter users (marginally but still) and much more spam marked with the Bitcoin hashtag. If we were investors, we would definitely read other sources of information to make up our mind, it is likely that other investors could be thinking the same in 2019.

# Conclusion

This study investigates the predictive power of social media metrics for bitcoin return price in an hour advance. It has been successful in acquiring the needed input data and extracting indicators out of it. Technically, this is a success. However, the result suggest that sentiment analysis of tweets is a poor indicator of future bitcoin returns, in our context.

We have found too many spam in the bitcoin’s tweets. It was also difficult to deal with the sarcasm in the comments that were not spam. We have learn also that the data set used to learn is a critical aspect to have success. In our project, we have used in a first stage a dataset from Kaggle that had a significant better quality of information regarding our objective than the twits we could gather from the API. Finally, we have learned that it is essential for this types of projects to contemplate that the stage cleaning data will take a lot of effort and time. And without success in this initial stage, the results will not have the performance expected.

As the result is not convincing, we do not advise to use Twitter to predict Bitcoins, not at least without improving the preprocessing stage of tweets and detecting positive and negatives more accurate. We also suggest that Bitcoins trends’ time series signal very hard to predict without any other useful data. It may be interesting to work instead of twits with articles in online newspapers that talk about bitcoins and cryptocurrency in general.

*Learning outcomes from team project*

In our view, as a team, the key takeaway from this project is what we gained from in terms of collaboration experience and applied theory. From the “theory” perspective, an important self-learning effort has been produced by each of us, in order to implement his/her part of the solution. But this has then been shared with the rest of team, enabling all members to gain important and valuable background and understanding in many different areas, technical and related to the subject matter. Overall this project has constituted a unique experience to learn and apply data science theory and methods, resulting in a great amount of knowledge gained:

* **Teamwork / collaboration**, using an agile method, with different profiles (ages, origins, backgrounds) and different personal constraints we had to manage.
* **Development / Python,** as most of us had little Python experience, if any at all.
* **Crypto currencies / trading**, as only one of us was proficient in this area, having owned a little crypto-portfolio
* **Social media** and related concepts, methods to access (API), role as a thermometer for many societal topics
* **Machine learning techniques**, especially neural networks, as we only had benefited from one lesson on “perceptron” during this project
* **Time Series Statistical techniques,** which is not a subject this is part of curriculum.
* **Data visualization,** to a lesser extent, as our solution is fairly straightforward in this area

*Possible improvements to solution*

In order to produce a solution that is produces better forecast, still using the same principles we could propose following steps:

* Build a dedicated spam filter with high efficiency, to increase the input quality (naive bayes or artificial neural networks)
* Take into account the number of retweets
* Consider other social media as well. Some research have used Google trends, such as, Instagram, linkedIn,...
* Training the model with combination of news articles and twitter data so the classification and the prediction would be more robust
* Modification to training set that provides equal number of words in each class (more balanced dataset)
* Reduce the limitation of considering tweets only in English. Would require important efforts, to find or produce training dataset in each language. Start with most common and promising ones, to maximize work/benefit ratio.
* Integrate other fundamental data, such as Gold rate, after checking their correlation level
* Once a sufficient predictive power is achieved, build a fully integrated solution, taking input and delivering results close to real time. Deploy in on a cloud infrastructure.

# References

Alfano, S. J., Feuerriegel, S. & Neumann, D., 2015. *Is News Sentiment More Than Just Noise?.* Freiburg, ECIS 2015 Completed Research Papers.

Anon., (2018) *rest-apis-streaming-apis.* [Online]   
Available at: https://snowflakesoftware.com  
(Accessed 10 April 2019).

Anon., (2019) *what-is-rest-api-design.* [Online]   
Available at: https://www.mulesoft.com  
(Accessed 10 April 2019).

Anon., n.d. *connecting.* [Online]   
Available at: https://developer.twitter.com  
(Accessed 10 January 2019).

Biau, G., (2012) ‘Analysis of Random Forest Model’, Journal of Machine Learning Research, 13, pp. 1-5.

Breiman, L., (2001) *Random forests*. Machine Learning, 45, pp. 5–32.

Brockwell, P.J, Davis, R.A., Calder M.V. (2002) *Introduction to time series and forecasting*, Springer

Cohen, L., (2002) *Autocorrelation function and the Wiener-Khinchin theorem*. Available at:<https://www.itp.tu-berlin.de/fileadmin/a3233/grk/pototskyLectures2012/pototsky_lectures_part1.pdf>. (Accessed: 14.04.2019)

Colianni, S., Rosales, S., Signorotti, M. (2015) Algorithmic *Trading of Cryptocurrency Based on Twitter Sentiment Analysis*. Available at: <http://cs229.stanford.edu/proj2015/029_report.pdf> (Accessed:14.04.2019).

Deering, S.,(2012). *developers-rest-api.* [Online]   
Available at: https://www.sitepoint.com  
(Accessed 10 April 2019).

Free Cryptocurrency Time Series Data (2019). Available at: <https://www.cryptodatadownload.com/> (Accessed: 14.04.2019).

Go, A., Bhayani, R., & Huang, L. (2009). “*Twitter sentiment classification using distant supervision.”* CS224N Project Report, Stanford, 1(12).

Heidenreich, H., 2018. *stemming-lemmatization.* [Online]   
Available at: https://towardsdatascience.com  
(Accessed 10 April 2019).

Hochreiter, S., Schmidhuber, J., (1997) *Long Short-Term Memory,* Available at: <http://www.bioinf.jku.at/publications/older/2604.pdf> (Accesed: 14.04.2019)

Hoffman, C., 2018. *what-is-an-api.* [Online]   
Available at: https://www.howtogeek.com  
(Accessed 9 April 2019).

Jabeen, H., 2018. *stemming-lemmatization-python.* [Online]   
Available at: https://www.datacamp.com  
(Accessed 11 April 2019).

Jang, H., Lee, J. (2017) ‘An Empirical Study on Modeling and Prediction of Bitcoin Prices With Bayesian Neural Networks Based on Blockchain Information’, IEEE, 6, pp. 2-4. Doi: [10.1109/ACCESS.2017.2779181](https://doi.org/10.1109/ACCESS.2017.2779181)

Koehersen, W., (2018) *An Implementation and Explanation of the Random Forest in Python*. Available at: <https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76> (Accessed: 14.04.2019).

Lamon, C., Nielsen, E., Redondo, E., (2016) Cryptovurrency Price Prediction Using News and Social Media Sentiment. Available at: [https://pdfs.semanticscholar.org/c3b8/0de058596cee95beb20a2d087dbcf8be01ea.pdf. (Accessed: 14.04.2019).](https://pdfs.semanticscholar.org/c3b8/0de058596cee95beb20a2d087dbcf8be01ea.pdf)

#### Loper, E. and Bird, S. (2002). *NLTK: “The Natural Language Toolkit*.” [online] arXiv.org. Available at: https://arxiv.org/abs/cs/0205028 [Accessed 8 Apr. 2019].

Mai, F., Bai, Q., Shan, Z., Wang, X. (2014) *From Bitcoin to Big Coin: The Impact of Social Media on Bitcoin Performance*. Available at: http://www.fmaconferences.org/Orlando/Papers/Bitcoin\_FMA.pdf (Accessed: 14.04.2019).

Margaret, R., 2018. *stemming.* [Online]   
Available at: https://searchenterpriseai.techtarget.com  
(Accessed 10 April 2019).

Matta, M., Lunesu, I., Marchesi, M. (2014) *Bitcoin Spread Prediction Using Social and Web Search Media.* Available at: <https://pdfs.semanticscholar.org/1345/a50edee28418900e2c1a4292ccc51138e1eb.pdf> (Accessed: 14.04.2019).

McAteer, C. (2014) *Twitter Sentiment Analysis to Predict Bitcoin Exchange Rate*. Available at: <https://www.scss.tcd.ie/publications/theses/diss/2014/TCD-SCSS-DISSERTATION-2014-055.pdf> (Accessed: 14.04.2019).

McNally, S., (2016) *Predicting the price of Bitcoin using Machine Learning*. Available at: <http://trap.ncirl.ie/2496/1/seanmcnally.pdf> (Accessed 14.04.2019).

Navlani, A., (2018) *Understanding Random Forests Classifiers in Python*. Available at: <https://www.datacamp.com/community/tutorials/random-forests-classifier-python> (Accessed: 14.04.2019).

Olshausen, B., (2000) *Aliasing PSC 129- Sensory Processes*. Available at: <http://www.rctn.org/bruno/npb261/aliasing.pdf> (Accessed: 14.04.2019)

Pang, B. and Lee, L. (2008): “Opinion mining and sentiment analysis.” Foundations and Trends in Information Retrieval, 2.1–2 : 1-135.

Parmar, H., Bhanderi, S., Shah, G., (2016*) Sentiment Mining of Movie Reviews using Random Forest with Tuned Hyperparameters*. Available at: <https://www.academia.edu/9434689/Sentiment_Mining_of_Movie_Reviews_using_Random_Forest_with_Tuned_Hyperparameters> (Accessed: 14.04.2019).

Rouse, M., 2012. *noisy-text.* [Online]   
Available at: https://searchbusinessanalytics.techtarget.com  
(Accessed 10 April 2019).

Rouse, M., 2019. *RESTful-API.* [Online]   
Available at: https://searchmicroservices.techtarget.com  
(Accessed 10 April 2019).

*Sasson, E.B., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E. and Virza, M., 2014, May. Zerocash: Decentralized anonymous payments from bitcoin. In 2014 IEEE Symposium on Security and Privacy (pp. 459-474). IEEE.*

Stenqvist, E., Lonno, J. (2017) *Predicting Bitcoin price fluctuation with Twitter sentiment analysis*. Available at: http://www.diva-portal.org/smash/get/diva2:1110776/FULLTEXT01.pdf (Accessed: 14.04.2019).

Wang, Y., Hodges, J., & Tang, B., (2003) “*Classification of web documents using a naive Bayes method*.” Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence.

Wang, R. Y., Storey, V. C. & Firth, C. P., 1995. A framework for analysis of data quality research. *IEEE Transactions on Knowledge and Data Engineering,* 7(4), pp. 623-640.