Effects of Social Media Bots on the Crypto Market

Daniel Jorge Deutsch

July 22, 2022

ABSTRACT

In this research, the impacts of Stocktwits bots on the crypto market is analysed. Firstly, data corresponding to the top 50 cryptocurrencies based on market capitalization is collected and processed. This data is then subjected to two classifications: one to identify bots and other to identify the sentiment expressed in its text. The engagement rate of each twit is then calculated and aggregated into groups based on the sentiment end user type to generate four different sentiment signals. The signals obtained for BTC, ETH and DOGE are, then, compared with their corresponding price changes via correlation matrices, Granger Causality Matrices and linear regressions. The results show that twits posted by bots have no influence on the price movements on their corresponding crypto asset. It is shown, however, that previous price changes have a causality effect on the twits' sentiment for some of the analysed cryptocurrencies. The study also shows that twits posted by bots usually don't impact price changes on their corresponding assets. Finally, the study concludes that, overall, posts published on Stocktwits have little to no impact on crypto assets' price changes.

Keywords: social network, bot, crypto market.

Nomenclature

AUC Area Under the Curve.

BERT Bidirectional Encoder Representations from Transformers.

tweet A social media post published in the Twitter platform.

twit A social media post published in the Stocktwits platform.

VECM Vector Error Correction Model.

Contents

List of Figures

List of Tables

1. Introduction

The year 2021 brought multiple examples of social media impacting financial market prices. The most discussed case was GameStop's share price increasing by nearly 600% in four days as a result of a series of postings made by Reddit members on the WallStreetBets forum. However, countless more incidents may be identified, either involving public figures' comments - such as Elon Musk's tweets affecting Tesla share prices - or a community of users - such as the aforementioned GameStop event.

The same phenomenon can be seen in the crypto market, which, according to CoinGecko, has been gaining more traction in the global economy, with its entire market value increasing by about $1 trillion just in 2021. However, this segment of the financial market has shown to be highly volatile and susceptible to speculation. In December 14th 2021, Elon Musk tweeted "Tesla will make some merch buyable with Doge & see how it goes", minutes later the spot price of Doge Coin had risen about 40%. As a result of events like this, several investors base their portfolios on social media publications in an attempt to profit from other users' speculations.

Because social media is a source of information that influences many crypto investors' decision-making processes, if someone is able to control the content displayed on these platforms, he or she may be able to manipulate market prices, as the asset's demand can be affected. This manipulation, though, depends on one key factor: visibility. The higher the number of people that see the content in a platform, the more likely it is to influence someone's investment decision.

This visibility, however, may be achieved in a variety of ways. The easiest approach is to have the content published by an account that already has a lot of visibility (i.e. followers). A more subtle way to achieve that is by creating and/or interacting with several posts on the social network that reflect the same opinion about a crypto asset. This will increase the opinion's visibility by making it one of the platform's trending topics.

Betting on this later approach, many investors/developers create bot accounts on these social networks to automate the process of creating/interacting with their content. This research, thus, proposes a methodology to investigate the effects that these bots have on the crypto market. The study is divided into five main phases that are, in most cases, dependent on the previous one: (1) data collecting and processing, (2) bot detection, (3) sentiment analysis, (4) generation of sentiment signals and (5) verifying the effects of the sentiment signals on crypto assets' price changes.

One key factor of this study is the selection of a social network platform to focus the analysis on. There are several social media websites used worldwide where people express their opinions about their investments. More mainstream ones are Twitter, Reddit and Discord and, because of that, they tend to present more cases of publications affecting assets' prices. This study, however, focus its analysis on the Stocktwits platform. This decision was made because of the following reasons:

1. Stocktwits' API have no request rate limitation, allowing the collection of a greater amount of data;
2. Stocktwits is a social media platform designed for investors, so people who use it are usually the ones who operate on exchanges;
3. Stocktwits gives users the option to label their posts as either bearish or bullish, allowing the use of supervised learning algorithms for sentiment classification;

The study's results show that, overall, twits published by bots on the Stocktwits social network tend not to influence crypto assets' price movements.

2. Related Work

As previously mentioned, this research contains within its steps: bot detection, sentiment classification, and an investigation of the impacts that the sentiment behind a twit posted by a bot has on the crypto market. Each one of these phases has a variety of published papers that served as a basis to what was performed throughout this project.

This study is heavily reliant on the ability to distinguish across user categories (bot or human). The US Government's Office of Cyber and Infrastructure Analysis defines social media bots an automated program capable of performing a variety of tasks while mimicking human behavior. This definition, however, is rather broad, as it classifies scrapers as bots, which are not the focus of this study. Oentaryo et al. 2016, later on expands this definition into three different categories: broadcast, consumption, and spam bots. This research, then, proceeds its analysis considering the effects of broadcast and spam bots on the crypto market as, unlike consumption bots, traces of these bot types may be found on the data returned by Stocktwits' API.

Wang et al. 2021 propose an algorithm to detect bots in social medias that is followed throughout this study. The algorithm assumes that, since bots represent a small portion of users in the social media, anomaly detection algorithms can be used to spot them. The proposed framework consists of three steps: encoding, decoding and clustering. The encoding and decoding steps are performed by a variational autoencoder neural network that receives as input user's information. The encoder proceeds to represent all the user feature's in a lower dimensional space (latent space) that is later on passed as input to the decoder. In turn, the decoder attempts to recreate the user's information in the original dimensional space. The theory is that because the model is primarily trained on regular data, when an outlier is introduced, the decoder would fail to recover the original information. The decoder's output is then combined with the original data and delivered to the kNN classification algorithm, which returns the distance between the query user and k of its neighbors. After calculating the average distance of each query user, a threshold is set in order to determine which user is a bot and which is human.

Li et al. 2021 propose a deep neural network architecture to classify the sentiment (positive or negative) underlying each post in the Weibo text dataset acquired during the COVID-19 pandemic. The performance of numerous neural network architectures is compared in the study, and

the model provided by the authors is shown to yield very accurate results. Their model's design includes a BERT embedding layer, an embedding structure proposed by Devlin et al. 2018 that leverages transformer blocks and attention heads and has been shown to improve the performance of text classification models. BERT is described as a context-aware encoding technique, meaning that each word's vector representation outputted by BERT depends on the context in which the word was used in the sentence.

Before training text classification models, it is good practice to perform text preprocessing. Alzahrani et al. 2021 shows that BERT embedding layers tend to provide better word representations when little to no noise reduction techniques are applied to the original text. Nguyen et al. 2020, at the same time, shows that their pre-trained BERT model (used to classify tweets' sentiment), provides better results when light noise reduction techniques are applied. Since the text data used in this study (twits) is very similar to the one used by Nguyen et al. 2020 (tweets), light noise reduction techniques were applied to texts in this study.

Finally, Mai et al. 2018 fits Twitter and internet forum activities as well as BTC market value along with some control variables into a VECM to show that Twitter publications' sentiment is an important predictor in determining bitcoin's valuation. However, their study also shows that the silent majority (the 95 percent of users who are less active and whose contributions amount to less than 40 percent of total messages) has a more significant effect on bitcoin's valuation. Their data is then subjected to the Granger Causality Test, error variance decomposition, and out-of-sample forecasting, with the findings indicating that forum sentiment has a strong predictive capacity for bitcoin's value.

3. Data Methodology

This research aims at understanding the impacts of social media bots on the crypto market. To do so, it is necessary to obtain and manage data related to the subject, which can be done through a series of steps.

3.1 Cryptocurrency Sample Space

Establishing a representative sample space for the cryptocurrency market is the first and most crucial step. This is accomplished by defining two parameters: a list of cryptocurrencies and a time period. To ensure that the conclusions obtained are without much loss of generality, these parameters should be picked in such a manner that the assets that comprise it reflect a large share of the crypto market. As they control the number of social media posts that will be considered in this research, their values should also take into account the time and hardware constraints for this study.

Taking that into consideration, the list of cryptocurrencies analyzed in this study is comprised of the top 50 cryptocurrencies by market capitalization that are tradable on the Binance exchange, and the time period considered is limited to dates from 2019-06-01 to 2022-06-01. Having these parameters set, it is possible to cross-reference the data provided by the Binance API with the crypto data provided by the CoinMarketCap's API, this yields the following list of crypto assets:

Table 3.1: Top 50 cryptocurrencies by market capitalization that are tradable on the Binance exchange

Cryptocurrency Sample Space				
BTC	LTC	XRP	DOGE	XMR
XLM	ETH	WAVES	ETC	ZEC
MKR	EOS	BCH	BNB	TRX
MANA	LINK	ADA	XTZ	FIL
THETA	TUSD	VET	USDP	USDC
FTM	ATOM	MATIC	ALGO	RUNE
FTT	KLAY	FLOW	HBAR	BUSD
SOL	HNT	AVAX	SHIB	SAND
NEAR	DOT	GRT	AXS	EGLD
UNI	AAVE	ICP	XEC	APE

3.2 Data Collection

After establishing the cryptocurrency sample space, the following step is to collect market and social media data for each of the assets on the list across the time period of interest.

3.2.1 Market Data

The goal here is to collect data that represents the market conditions for the cryptocurrency sample space. This is achieved by using Binance's API to collect hourly close prices for each one of the cryptocurrencies listed in Table 3.1 over the established time period. Figure 3.1 illustrates what this data looks like for the BTC and ETH cryptocurrencies.

Figure 3.1: Hourly close prices of BTC and ETH over time

Figure 3.2 illustrates a heatmap that displays missing data for each cryptocurrency over time. It displays in black the absence of data and in white its presence. Consequently, whereas older cryptocurrencies such as BTC and ETH have a completely white bar, newer coins such as SHIB and APE have their bars predominately dark.

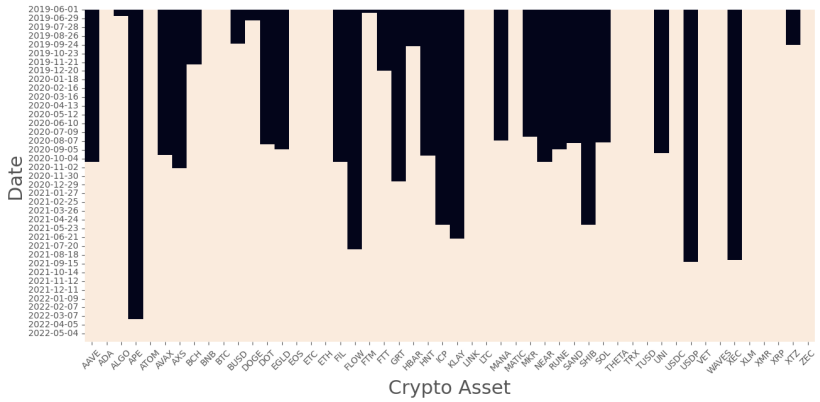

Figure 3.2: Heatmap of missing market data in the cryptocurrency sample space

It is worth mentioning that what seems to be missing market data in the dataset is simply because Binance began to enable operations on some of the cryptocurrencies listed in Table 3.1 later in time.

3.2.2 Social Media Data

As stated in the introduction, the social media sentiment signal calculated in this research is based on Stocktwits posts; consequently, the objective here collect all the twits linked to the cryptocurrency sample space via Stocktwits' API. It is important to mention, however, that while it was possible to gather all the twits from forty nine of the the cryptocurrencies listed in 3.1, this was not the case for the DOGE cryptocurrency, which had its data partially collected - from 2021-05-03 until 2022-06-01 - due to a relatively big latency on the response from Stocktwits' API and an abnormally large volume of twits. Table 3.2 presents some descriptive statistics over the collected data.

Table 3.2: Number of twits collected per label

	Number of Samples
Bearish	465,004
Bullish	3,884,532
NaN	2,326,433
TOTAL	6,675,969

When analysing Table 3.2, two problems should be noted: the considerate amount of unlabeled twits - represented by *NaN* - and the imbalance of the dataset.

When posting a twit on Stocktwits, users may choose whether to designate its content as bullish or bearish. This tagging, however, is not mandatory, leading many users to publish untagged twits. Furthermore, in an attempt to mimic human behaviour, several bots tend to publish a certain percentage of untagged twits. The combination of these two phenomena result in a high proportion of unlabeled twits on the website and, consequently, on the collected dataset.

As the goal of this study is to investigate the effects of bots on the crypto market, having unlabeled twits is a big concern as they may also convey market effects. This implicates the necessity of manufacturing a classification model capable of identifying bullish and bearish content on twits. This model, however, should be carefully designed and trained since, as previously mentioned, the collected data is very imbalanced - 90% of it is tagged as bullish. Imbalanced datasets are a serious issue when training the classification models as it leads to biased and hence unrepresentative results.

Figure 3.3 displays the label representativity for each crypto asset in the twits dataset. When training the model, it is also important to ensure that the training data is a good representation of the cryptocurrency sample space, so that it provides more accurate results.

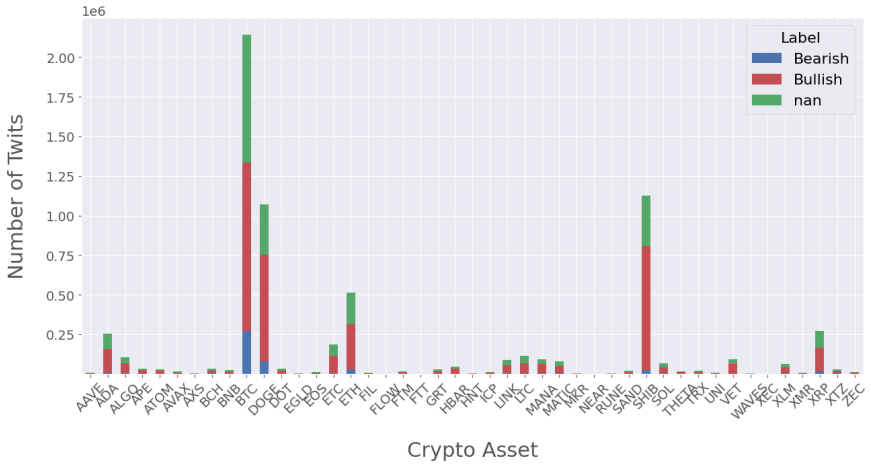

Figure 3.3: Number of collected twits per crypto asset

As expected, the more well-known cryptocurrencies have a larger volume of twits, as it is the case of BTC, ETH, SHIB and DOGE, that together sum up to 4,847,364 twit samples (73% of the collected dataset). It is also worth noting that the imbalanced label proportion remains across all cryptocurrencies, with a predominance of bullish and unlabeled twits.

Because the cryptocurrency sample space is also time dependent, it is critical to understand how the volume of twits changes over time. As demonstrated in Figure 3.4, there is a rise in the number of twits during the fourth quarter of 2021 that lasts until May 2022. Additionally, April 2021 and September 2021 stand out as anomalies due to the enormous amount of twits in those months. It is also noticeable that the uneven label proportion persists over time.

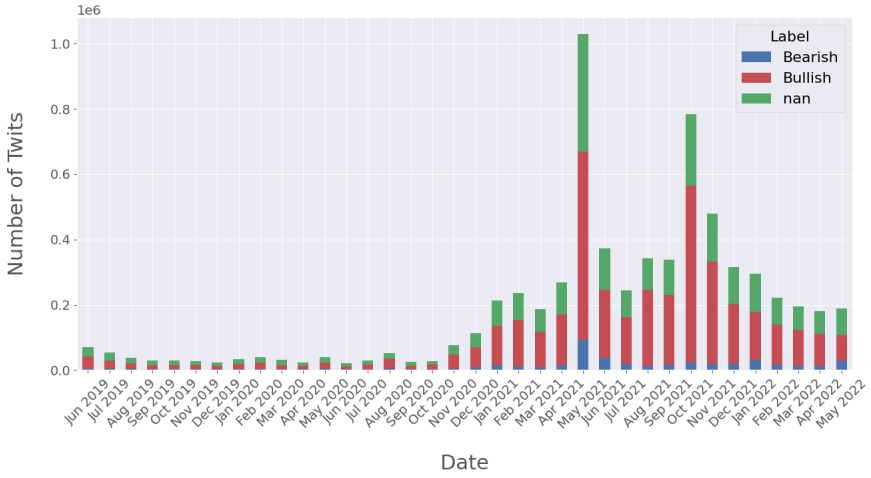

Figure 3.4: Number of collected twits over time

Finally, because one of the steps in this study requires the creation of an hourly aggregated twit sentiment signal for cryptocurrencies, it is critical to ensure that there aren't many gaps in the hourly aggregated twits, i.e., the

time interval between twits that mention the cryptocurrency of matter isn't frequently greater than one hour.

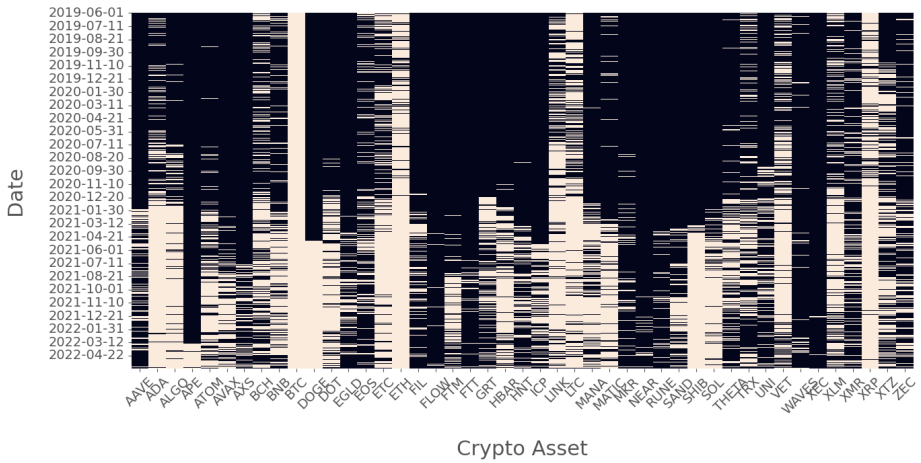

Figure 3.5: Heatmap of hourly gaps for each cryptocurrency in the twits dataset

Figure 3.5 illustrates the hourly gaps for each crypto asset. It is observed that BTC, ETH and DOGE are among the cryptocurrencies with the fewest gaps since they have the least interpolation between white and black stripes. This implies that the hourly sentiment signal for these coins would have the least amount of gaps. Consequently, the research should concentrate on these coins to investigate the effects of bots on the crypto market, as they will provide a more accurate results.

3.3 Data Processing

After gathering the raw data returned by Stocktwits' API, the next step is to process it in order to facilitate its manipulation and, hence, analysis.

The raw data follows the structure indicated in Appendix A. There are two factors to notice when observing the ingested data structure: (1) it contains information about the user who published the twit and (2) there are several columns with nested relevant information.

Since this research relies on Stocktwits' user information - which is required to distinguish different user types (bot or human) - having easy access to it is very convenient. Additionally, the API offers a multitude of relevant information about the twits, such as the number of likes and reshares, in a nested JSON structure that is difficult to manipulate.

As a result of (1), the collected twits dataset was divided into two: a processed user dataset - containing information about the users who published the collected twits (with 162,430 users) - and a processed twit dataset - with information regarding the twit itself. In the later one, due to (2) its contents were processed into a structure that facilitates the access to the twit's attributes. The structures of the processed datasets can be verified on Appendixes B and C.

4. Bot Detection

The US Government's Office of Cyber and Infrastructure Analysis defines social media bots as:

> *"Programs that vary in size depending on their function, capability, and design; and can be used on social media platforms to do various useful and malicious tasks while simulating human behaviour".*

Oentaryo et al. 2016 expands this definition into three different categories: (1) broadcast, (2) consumption and (3) spam bots.

1. **Broadcast Bot:** disseminates information to a broad audience by giving, for example, innocuous links to news, blogs, or websites. Such bots are frequently operated by an organization;
2. **Consumption Bot:** aims at gathering material from several sources and/or providing update services (e.g., stock changes, relevant news) for personal consumption or usage;
3. **Spam Bot:** aggressively promotes innocent but invalid (or irrelevant) material or uploads harmful content;

As aforementioned, consumption bots are often programs designed for personal use. Because of that, they leave no trace in the social media database, making it hard to analyze their influence on the market having only access to Stocktwits' API. As a consequence, this research examines the effects of broadcast and spam bots on the cryptocurrency market.

With information on the author of each collected twit, it is feasible to estimate whether a twit was posted by a human or a bot. There are numerous techniques to infer an author's user type from its user data; more sophisticated solutions, such as the one proposed by Ping et al. 2018, employ deep learning algorithms to do this task. However, because the processed user dataset is unlabeled, that is, there is no feature in the dataset explicitly indicating whether a user is a bot or not, supervised learning techniques cannot be used to categorize the data. Consequently, the use of unsupervised learning techniques make itself necessary in this classification task.

4.1 Anomaly Detection

Varol et al. 2017 estimates that bot accounts represent between 9% and 15% of all Twitter's users. Given the similarities between the two social media platforms, it is reasonable to assume that Stocktwits has a similar percentage of bots. Being 15% a small portion of users, it is possible to describe bots on Stocktwits as an anomaly (i.e. an exception to the norm) on the platform, and therefore anomaly detection methods can be leveraged to spot bot accounts in the dataset.

With the intense use of data over the last decade, several anomaly detection algorithms have emerged. Nonetheless, Wang et al. 2021 suggests an approach that leverages a variational autoencoder neural network whose output is passed to a K Nearest Neighbors (KNN) model and, based on the distances outputed by the later model, classifies the user as being either an anomaly (bot) or not (human). Since this approach is shown very accurate and can be can be easily adapted to the Stocktwits context, the bot detection section of this study is based on it.

4.1.1 Data Enhancement

Besides the user's information retrieved by Stocktwits API, it is beneficial to enhance the users dataset by adding information over the content of the twits posted by each user. Wang et al. 2021 suggests the usage of of several content-related attributes for the Twitter user that also fits the Stocktwits' context. Based on Wang et al. 2021's proposed attributes, the following features were added to the users dataset:

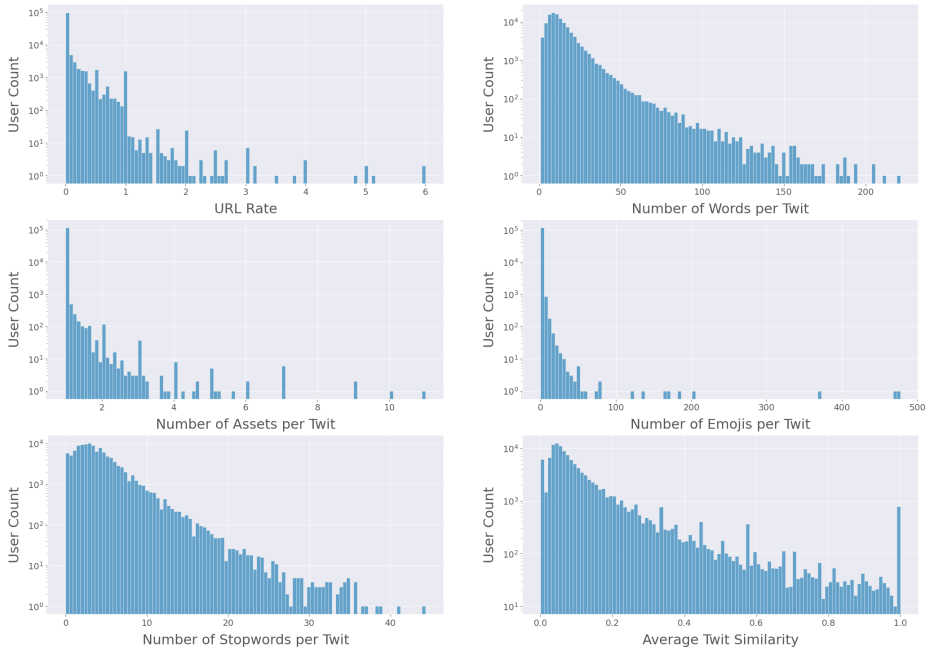

Figure 4.1: Distribution of the user's content-related features

- **URL Rate:** average number of links (URLs) per twit posted. As previously mentioned, bots tend to post content with a specific goal in mind, which could be spreading harmful links or information to other users. Thus, it is natural to assume that bots shall have a higher URL rate than humans.

- **Number of words per twit:** average number of words in the twits posted by the user. To achieve their distribution goal, social media bots typically include a lot of unnecessary material to their twits, resulting in a higher number of words in it.

- **Number of assets per twit:** average number of assets tagged (using the Stocktwits' syntax, e.g. $BTC.X) in the twit. To have their content shown in more search results, bots tend to tag a higher number of assets on the same twit.

- **Number of emojis per twit:** average number of emojis in the twits posted by the user. The writing style of social media bots differs greatly from that of regular people. Bots' twits are either overflowing with emojis or lack any facial emojis at all. Humans, on the other hand, do not employ emojis to such an extent.

- **Number of stopwords per twit:** average number of stopwords in the twits posted by the user. Humans tend to use more consistent amount of stopwords (set of commonly used words in the English language) than bots in their twits.

- **Average twit similarity:** a measurement of how similar the content of twits posted by each user is. As bots normally have an artificial intelligence model behind the generation of the content of their posts, it is normal to assume that there is a higher similarity between twits posted by bots. To obtain this measure, the Term Frequency Inverse Document Frequency (TF-IDF) is used to weight each word, then a similarity matrix (with dimensions equal to the number of twits posted by the user) is obtained by calculating cosine similarity between the weights of each pair of twits. Finally, the average of the upper triangle of the matrix is calculated, giving the average similarity between each pair of twits posted by the user.

- **Number of punctuation per twit:** average number of the seven kinds of punctuations. This is not represented as one feature per se, but as seven different features, each one containing the average number of times the punctuation symbol of matter is used in the user's twits. The symbols considered are: ",", ".", ";", """, "!", "(", and ")".

Once all these features were added, it was also necessary translate some of the user attributes provided by Stocktwits' API into numbers. The final form of the enhanced users dataset can be verified on Appendix D. It is also important to mention that users with only one twit were dropped from the

enhanced dataset because the similarity measure, in these cases, provide no useful information. This deletion of users left the enhanced user dataset with 116,890 samples.

4.1.2 Variational Autoencoder (VAE)

Reconstruction methods are a prominent approach to anomaly detection that has gained traction as deep learning has become more widely available. The basic premise is that if a model can learn a function that compresses and reconstructs normal data, it would fail to do so when confronted with anomalous data since the function was trained on normal data. Failure to reconstruct data, or, more precisely, the range of reconstruction error that it implies, might thereby indicate the presence of anomalous data.

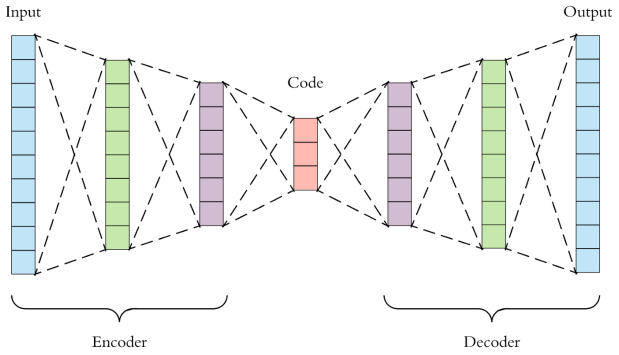

Figure 4.2: Illustration of a generic autoencoder (Dertat 2017)

An autoencoder - illustrated in Figure 4.2 - is a deep learning model with two primary components: an encoder that learns a lower-dimensional representation of input data and a decoder that attempts to replicate the input data in its original dimension using the encoder's lower-dimensional representation. The theory behind this architecture is quite similar to that of image compression: a well-trained encoder learns to encode the input data in such a manner that it captures the most relevant information it contains and is therefore adequate (or as close to being sufficient) for the decoder to recreate.

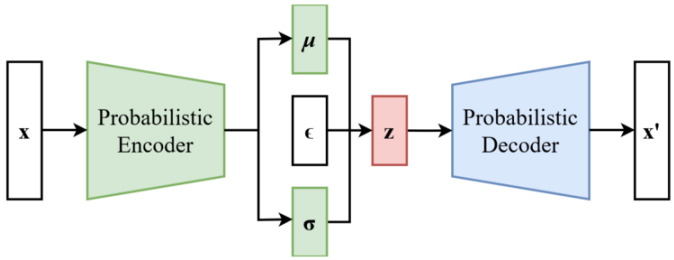

Figure 4.3: Illustration of a generic variational autoencoder (EugenioTL 2021)

In a variational autoencoder (VAE) - illustrated in Figure 4.3 - the encoder similarly learns a function that takes as its input a vector of size n. However, instead of learning how to generate a latent vector that the decoder function can reproduce, as traditional autoencoders do, a VAE learns to generate two vectors (of size m) that represent the parameters (mean and variance) of a distribution from which the latent vector is sampled, and which the decoder function can transform back to the original input vector. Simply put, the autoencoder's learning task is to learn a function that transforms data into a latent vector that a decoder can easily reproduce, whereas the VAE's learning task is to learn a function that generates distribution parameters from which a latent vector that a decoder can easily reproduce can be sampled.

The VAE model implemented in this study is used to encode and decode the enhanced user's features. The encoder consists of four layers: one input layer with 35 neurons (the number of the user's numerical features), one fully-connected hidden layer with 17 neurons, one fully-connected hidden layer with 22 neurons (11 of them represent the mean and the other 11 represent the variance of the distribution) and one output layer with 11 neurons. The encoder's output layer contains the latent vector obtained from the sample of the distribution with parameters obtained from the hidden layer. This latent vector is, then, passed as the input to the decoder, which follows a mirrored structure of the encoder (one input layer with 11 neurons, one fully connected layer with 17 neurons and one output layer with 35 neurons).

After establishing the architecture for the VAE model, the following step is to build and train the neural network. To accomplish so, the network training parameters must be defined: its optimizer and loss function.

When working with VAEs, it is typically ideal to work with adaptive optimizers with a low learning rate; consequently, the Adam optimizer with a learning rate of 1E-4 was employed to train the network. As for the loss function, VAEs are usually trained with a loss that consists of a reconstruction term (which makes the encoding-decoding method efficient) and a regularization term (that makes the latent space regular). The loss function used in this study is given by the sum of the Mean Squared Error (MSE) and the Kullback-Leibler Divergence, which can be expressed as:

$$\text{Loss} = ||x - x'|| + KL[\mathcal{N}(\mu_x, \sigma_x),\ \mathcal{N}(0, 1)]$$

Where x is the encoder's input, x' is the decoder's output, μ_x is the mean of the sampled distribution and σ_x is the square root of the variance of the sampled distribution.

In addition to the conventional training settings, the model was also trained with callback functions: Model-Checkpoints (to save the best weights at each epoch) and Early-Stopping (with patience of 5 based on the loss value, to avoid overfitting).

The model, is then, set to be trained for 500 epochs with a batch size of 1024 user samples. Figure 4.4 displays the training loss over the epochs.

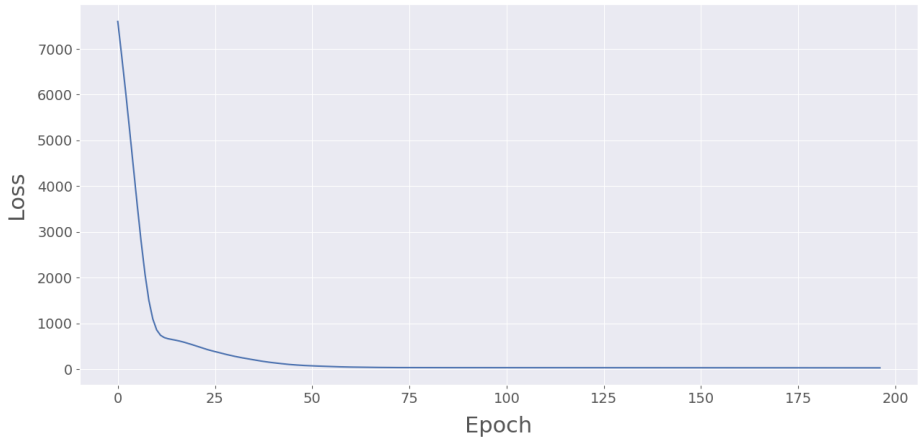

Figure 4.4: Variational autoencoder's training loss over the epochs

The first thing to note is that the model did not train for the 500 epochs that were specified, implying that the Early-Stopping criterion was met at the 197th epoch. The figure also shows that the loss converges and reaches a plateau at the 50th epoch.

After calculating the neural network weights, the samples from the enhanced user dataset are sent to the VAE and their decoded values are stored.

4.1.3 k-Nearest Neighbors (kNN)

k-Nearest Neighbors (kNN) is a non-parametric supervised learning algorithm frequently used for classification problems that works on the assumption that similar data points should be found close to one another. Because of its premises, kNN is also broadly used in anomaly detection problems as, by definition, outliers (i.e. anomalies) are data points that are distant from normal data. When it comes to anomaly detection the kNN algorithm takes an unsupervised approach. This is because there is no actual predetermined labeling of "outlier" or "not-outlier" in the dataset, instead, it is entirely based upon threshold values.

The k parameter in the kNN algorithm specifies how many neighbors will be searched to determine the classification of a single query point. Different values of k might lead to overfitting or underfitting, therefore defining it can be a balancing act. Lower k values usually leads to high variance and low bias, while higher k values tend to result in high bias and low variance. Wang et al. 2021 uses cross-validation techniques to show that, in the dataset used by them, k=6 results provides a better classification accuracy. However, because the enhanced user dataset contains no information about the user type (bot or human), performing cross-validation across values of k is not feasible because there is no way to verify whether the predicted labels are correct and therefore compare the accuracy achieved from different values of k. Thus, the value k=6 is adopted in this study.

Wang et al. 2021 also shows that a higher accuracy is obtained when the distance between two data points is computed using both the decoded and

original values of the user's features. Following this result, to determine the distance between users, the original values of the user's features are combined with the VAE output, yielding a dataset with 70 dimensions. Once the distances are calculated, the average distance for each query point is taken, providing the distribution depicted in Figure 4.5.

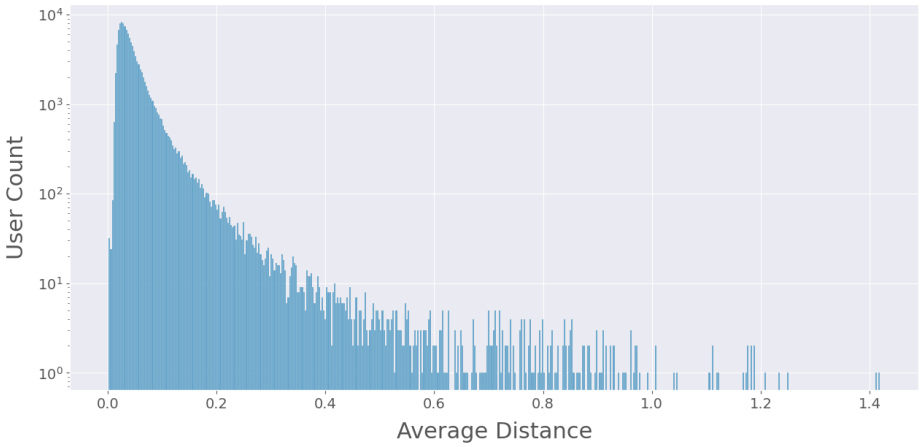

Figure 4.5: Distribution of the average distance between each user and six of its neighbors

The establishment of a distance threshold is the final stage of this classification problem. Every user whose average distance is greater than the threshold is regarded as an anomaly (bot), whereas those whose average distance is less or equal to the threshold are considered regular users (human). According to Varol et al. 2017, bots account for 9% to 15% of Twitter users, and the similar amount may be anticipated for Stocktwits. Thus, the distance threshold is set to be 0.09 based on the distribution displayed in Figure 4.5 and the proportion of bot representation on Stocktwits.

4.1.4 Bot Detection Results

With the average distance threshold set, it is possible to properly classify users as either being bots or humans.Table 4.1 shows the number of users classified as each type. It is notable that around 11.18% of the users were classified as bots.

Table 4.1: Number of users per user type

	Number of Samples
Bot	13,069
Human	103,821
TOTAL	116,890

With the inferred user type in hand, various aspects of the tweets posted by each user group may be compared. The first plot in Figure 4.6 reveals

that bots have the same labeling behavior as humans, i.e., there are more twits marked as bullish (or even without a tag) than bearish. The second plot depicts the growth in the number of twits published by bots and humans in the fourth quarter of 2020, which remained until May 2022. Furthermore, it is worth noting that outlier months in terms of twit volume are outliers for both humans and bots. Finally, the figure's final plot reveals that the most touted crypto assets among humans are the same as those among bots. Overall, the proportion of twits made by each user category is similar across all bar and plots.

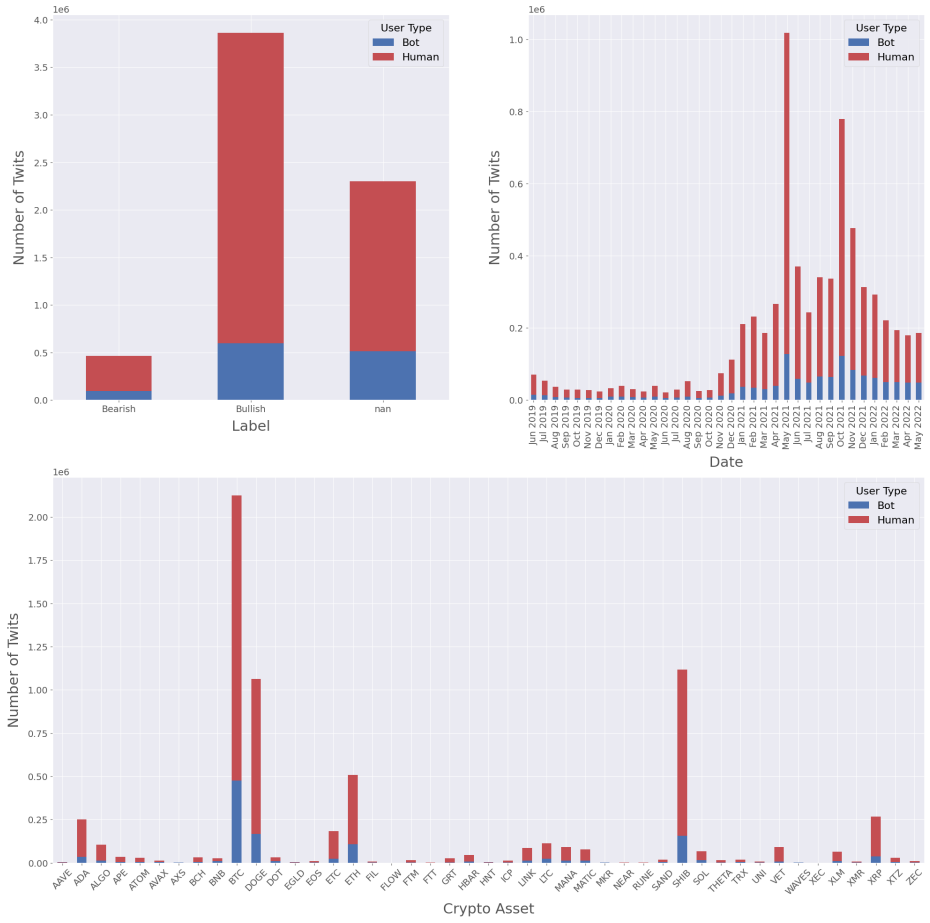

Figure 4.6: Bot and human twit dataset comparison

Furthermore, it is possible to compare the feature's distribution of each user type, as shown in Figure 4.7. Overall it can be said that bots tend to have a higher representation in the tail of the distribution than humans. As projected, there are more bots publishing content with more URLs, emojis, stopwords and tagged assets than humans. The same behaviour is seen in the usage of punctuation in the twits. Its also possible to notice that human twits tend to be shorter in words but also have a higher variety of content (lower average cosine similarity). When it comes to user interaction, it is shown that there are more bots with a higher number of followers, following accounts, twit posting frequency and number of likes than humans.

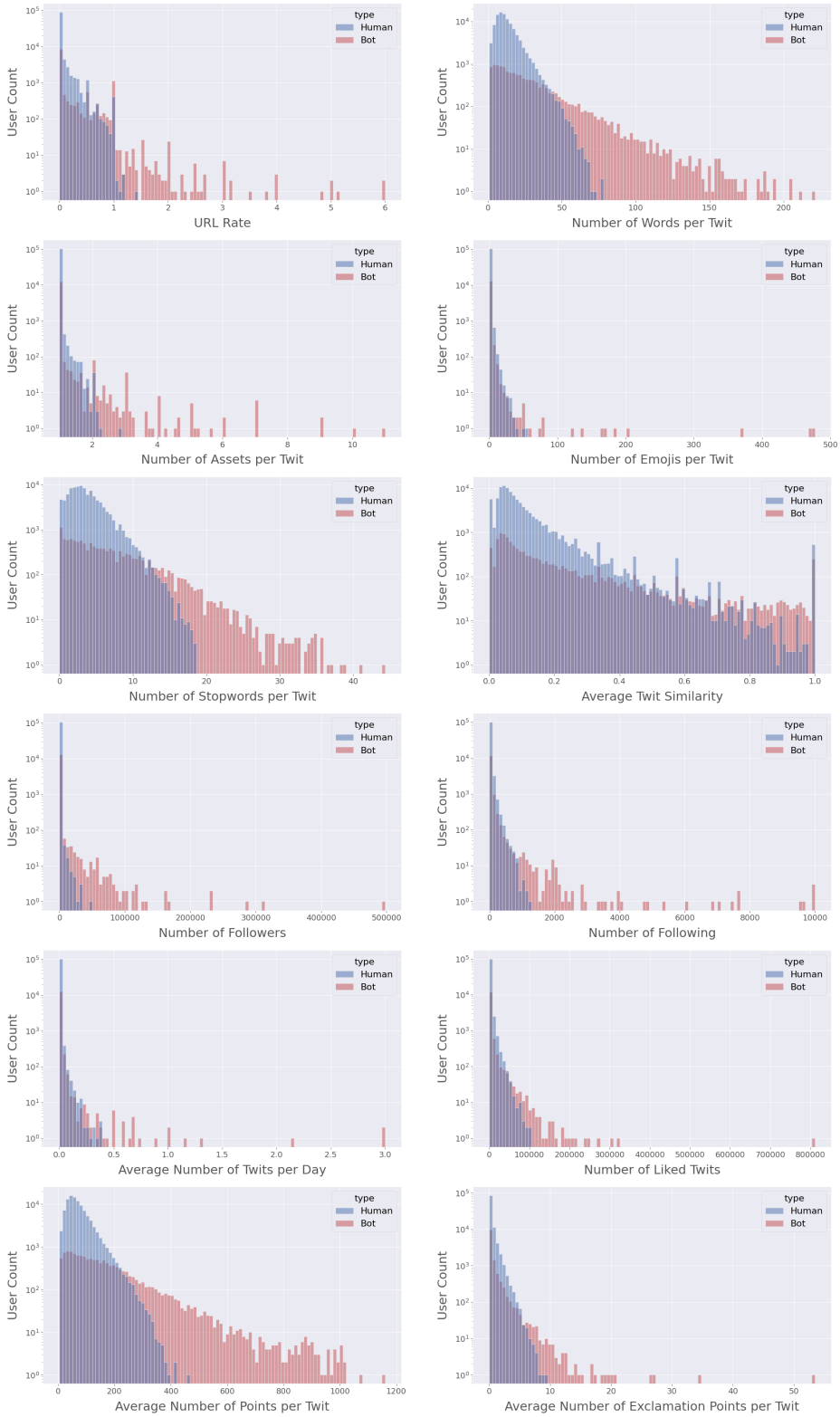

Figure 4.7: Bot and human feature's distribution

5. Sentiment Classification

Because unlabeled twits account for 35% of the twits dataset, identifying the sentiment behind them is a critical step in this study that may be achieved using Natural Language Processing (NLP) techniques. These algorithms may take a twit's corpus as input and return the sentiment expressed by it - between the Bullish and Bearish categories - as output.

Being this a supervised learning problem (most of the rows contain the sentiment label of the corresponding twits), there is a range of algorithms that can be used for this classification task. These algorithms can vary from more simple ones such as Support Vector Machines (SVM) to more robust ones, like deep neural networks. Because this research involves the creation of a reliable sentiment signal, it is important to select an algorithm with high precision. Mariel et al. 2018 demonstrates that, in terms of precision score, deep neural networks outperform simpler techniques for sentiment analysis tasks. Taking this into consideration, it was decided adopt a deep neural network structure to move on with the classification task.

Deep neural networks, however, are unable to operate when they are fed with text data as they require numerical inputs. As a consequence, each twit's text corpus must be represented in a quantitative (i.e. numerical) format. This is usually achieved through a process called word embedding, a technique where individual words are represented as numerical vectors in a lower dimensional space.

There are, nonetheless, several ways to implement word embedding. Commonly used methods are: binary encoding, TF encoding, Word2Vec and BERT. Each one of them has its advantages and disadvantages, but the later one, developed by Google, tends to provide better results as, according to Devlin et al. 2018, it leverages bidirectional transformer layers, meaning it provides a context-aware representation of each word. To exemplify this, consider the sentences below:

$$\text{I will } \textbf{watch} \underbrace{\text{TV tonight}}_{\text{context}}$$

$$\underbrace{\text{It's 5 o'clock according to my}}_{\text{context}} \textbf{ watch}$$

Given the contexts, it is evident that the word "watch" in the first sentence refers to the action of observing something, whereas it refers to the watch object in the second one. The idea is that, while methods such as

Word2Vec would represent the word "watch" in both phrases in the same way, BERT would yield a different vector representation for the word in each sentence, since it has different meanings in each one.

Due to BERT's sophisticated structure, a lot of training time and costs are usually required. Google, however, has made its source code available to the public, allowing developers with more financial resources to pre-train their BERT implementation (using their preferred dataset) and share their model, along with its weights, to the public. With a pre-trained BERT in hand, developers have the option to add it to their model as is, or fine-tune the embedding layer according to their primary task. Since pre-trained BERT models are typically trained with generic texts (e.g. Foundation n.d., Zhu et al. 2015), it is usually a good idea to fine-tune the model using matter-specific data, to ensure that the embedding adapts its weights to the jargon of the texts.

Since twits posted on Stocktwits have a very specific jargon, specially twits concerning crypto-related assets, the embedding weights must be adjusted to the dataset. Accordingly, this study implements its sentiment classification model by fine-tuning a BERT embedding layer and feeding its output to a deep neural network.

5.1 BERT Based Neural Network

As mentioned, this study implements a model that leverages BERT embedding techniques and a deep neural network structure to predict the sentiment expressed by a twit. Li et al. 2021 compares the performance many neural network structures in a problem very similar to this one and show that:

1. Models that leverage Bidirectional Long Short-Term Memory (BiLSTM) layers on top of the embedding one tend to provide better results. This is due to the fact that BiLSTM considers context information from the text in two directions and solves the corresponding dependencies of large text;
2. The usage of an attention mechanism can also increase the accuracy as it gives different degrees of attention to different words;

Taking these into consideration, Li et al. 2021 proposes a structure (illustrated in Figure 5.1) that outperforms most of the models evaluated in their study. The neural network consists of five hidden layers: embedding, BiLSTM, attention, convolutional and fully connected.

In this architecture, a text sentence from the input layer is sent to the BERT embedding layer, which delivers word vector representations to a bidirectional LSTM layer. The BiLSTM layer then extracts information from both the forward and backward directions of the text. Because distinct words have different impacts on assessing the overall emotion polarity of the text, the model includes an attention mechanism that assigns a sentiment weight

to each word. The stronger the emotion in a word, the more attention (i.e. weight) it receives. The data is then sent to a convolutional layer, which isolates the most influential local feature information in the input text information via a filter, allowing the data dimension to be reduced and the model to become position invariant. Once the convolution is complete, a pooling operation is conducted to further compress the convolutional feature vector, reducing the vector dimension and computational cost. The result is then subjected to a batch normalization technique in order to accelerate training and decrease overfitting. The normalized data is subsequently delivered to a fully connected layer to be converted into the final emotional representation. Finally, the final representation is passed to the output layer - which contains one neuron with Sigmoid activation function - for label classification.

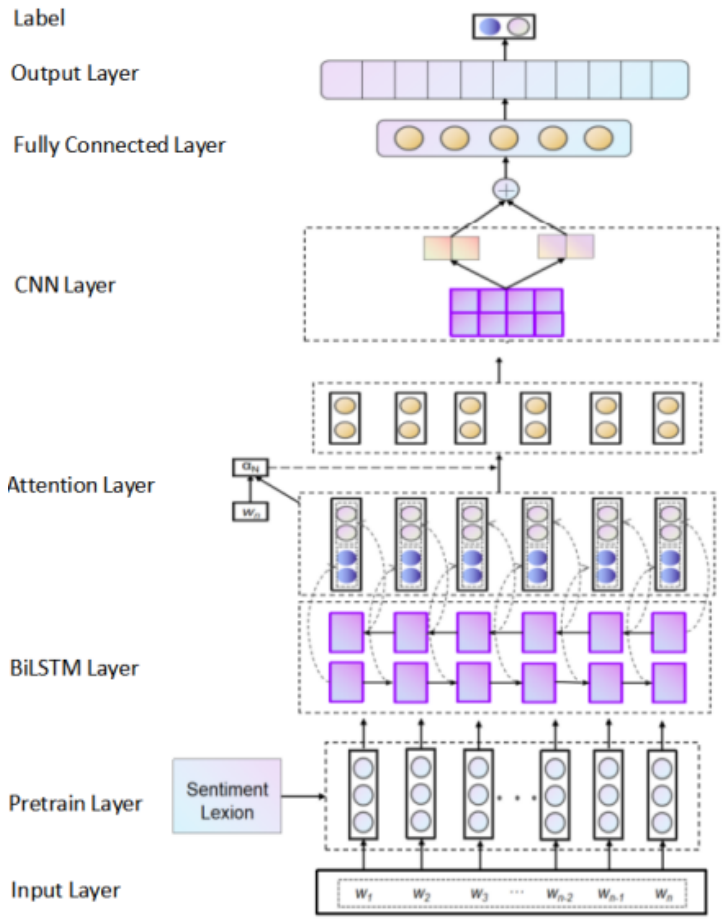

Figure 5.1: Neural network structure proposed by Li et al. 2021 for text classification (Li et al. 2021)

Given the results presented by Li et al. 2021, it was decided to adapt their proposed model to this study's characteristics and constraints by: (1) adding dropout layers, (2) implementing only max pooling operations and (3) using a smaller implementation of the BERT embedding layer.

1. As suggested by the authors, to avoid overfitting, it was added to the network two dropout layers: one after the embedding

layer and one after the pooling layer, with 60% and 40% dropout rates, respectively.

2. Instead of performing both average pooling and max pooling to the output of the convolutional layer and concatenating the results, for simplicity, only max pooling was performed.
3. Due to hardware constraints for this study, instead of using the original BERT Base embedding structure (12 transformer blocks, 768 neurons in each hidden layer and 12 attention heads), it was used the Smaller BERT model (4 transformer blocks, 128 neurons in each hidden layer and 2 attention heads) provided by the TensorFlow Hub library.

It is important to acknowledge that, aside from the dropout regularization, which tends to improve the quality of the results by avoiding overfitting, the adjustments applied were due to project restrictions and are likely to slightly degrade the classification's accuracy.

5.1.1 Text Preprocessing

Before feeding the BERT model with the twit corpus, it is good practice to reduce the text noise from each twit, i.e., delete/transform words and characters in the text that degrade the NLP task model. Noise reduction (or text preprocessing) is a very broad concept that can be applied through a combination of techniques.

Alzahrani et al. 2021 shows that BERT tends to perform better with little to no text processing because, according to the authors *"[...] pre-trained models perform better on larger texts and need every token that they might learn from."*. Taking this into account, and considering that data visualization is an important step in modeling the data, it was decided to apply two versions of text preprocessing: one with light text cleaning to feed its results to the classification model and one with heavy text cleaning to improve data visualization.

Table 5.1: Example of light and heavy text cleaning to a twit corpus

	Twit Corpus
Original Text	$BTC.X Sorry bears we bought the dip. We will not be fooled by this weak rug pull. Better cover, your short is burning cash!! 🔥 🔥 🔥 lmao
Light Text Cleaning	btc sorry bears we bought the dip . we will not be fooled by this weak rug pull . better cover , your short is burning cash ! ! :fire: :fire: :fire: lmao
Heavy Text Cleaning	btc sorry bears bought dip will fooled weak rug pull better cover short burning cash fire lmao

Light Text Cleaning

This text cleaning variation combines a set of noise reduction techniques suggested by Nguyen et al. 2020 with techniques developed specifically to handle content from Stocktwits. It consists in the following text transformations: (1) lowercase the text, (2) remove Stocktwits' specific stock tags (e.g. "$btc.x" to "btc"), (3) replace URLs with the "httpurl" special token, (4) replace user mentions with the "@user" special token, (5) apply the TweetTokenizer from the nltk python library to the text and (6) translate emojis to text strings. Table 5.1 illustrates the results of applying these techniques to a twit.

Heavy Text Cleaning

In addition to all of the techniques used in light text cleaning, the heavy variation of text preprocessing also applies: (1) replace letters that consecutively repeat themselves three or more times with the same letter with two consecutive instances, (2) replace consecutively repeated words with the one instance of the same word, (3) removes numbers, (4) replaces word contractions with the original word, (5) removes stopwords, (6) removes punctuation. Table 5.1 compares the results of the preprocessing techniques.

5.1.2 Text Visualization

Before training the model, it is important to visualize the texts that constitute the dataset, since this might give valuable insights about the data. Nonetheless, it is essential to remember that all of the visualizations in this section were generated using heavily cleaned text.

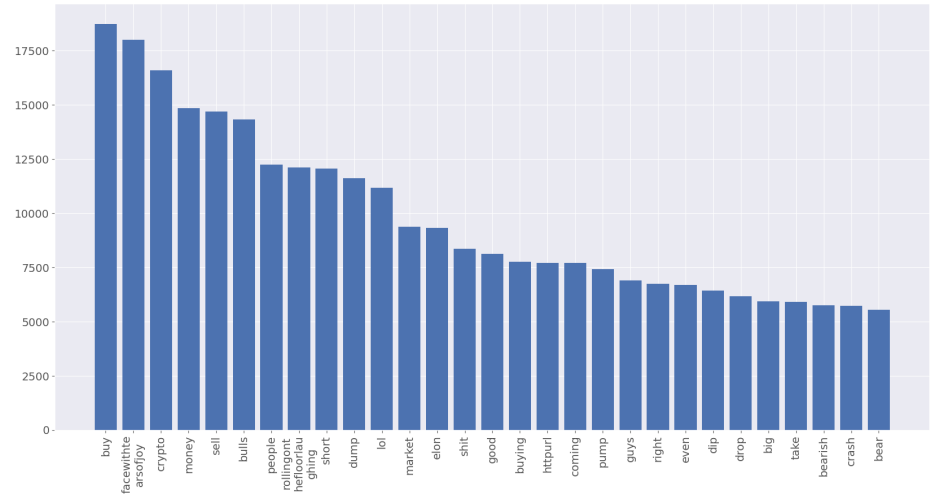

Figure 5.2: Top 30 most frequent words on bearish twits made by humans

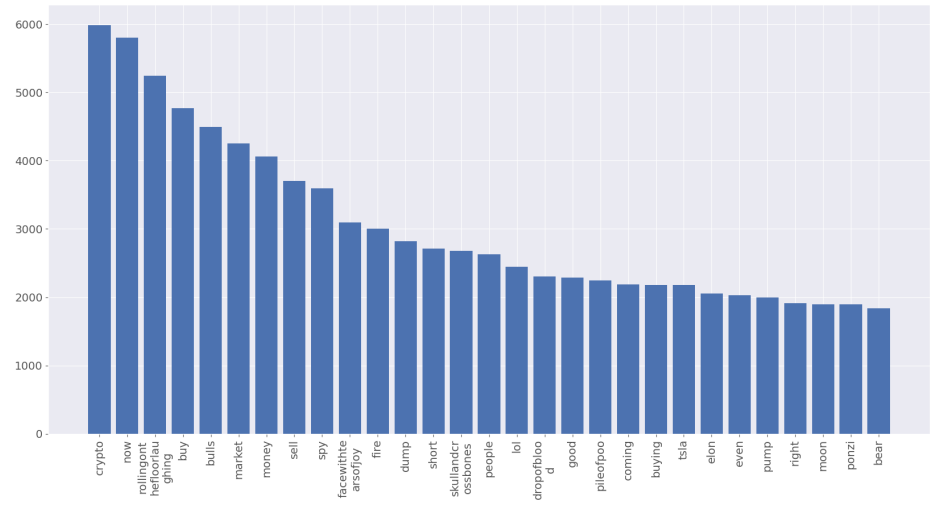

Figure 5.3: Top 30 most frequent words on bearish twits made by bots

Figures 5.2 and 5.3 show the 30 most frequently used words of each user type when publishing bearish content. It is possible to notice that bots tend to use a higher amount of emojis and lesser amount of URLs in their twits when compared to humans, which partially contradicts with Oentaryo et al. 2016's definitions of broadcast and spam bots. Finally, it is worth noting that the words most commonly used by bots are very similar to those used by humans.

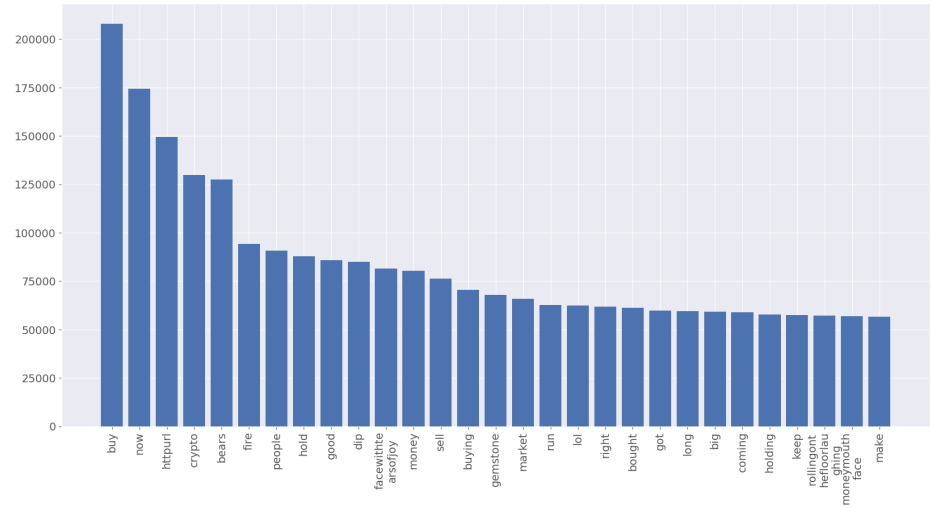

Figure 5.4: Top 30 most frequent words on bullish twits made by humans

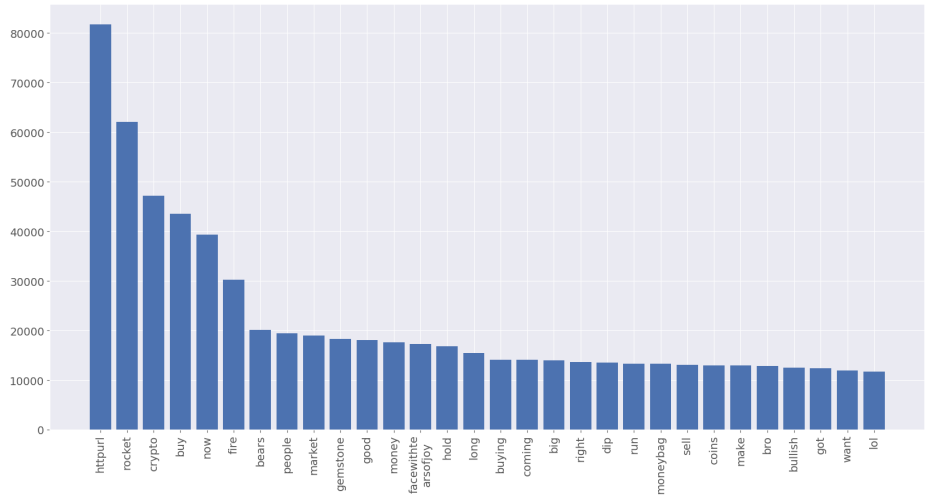

Figure 5.5: Top 30 most frequent words on bullish twits made by bots

Correspondingly, Figures 5.4 and 5.4 show the 30 most frequently used words on bullish twits for each user type. The first thing to notice here is the fact that URLs are by far the most frequently used structure in bullish twits made by bots. This goes according to the definition of spam and broadcasting bots made by Oentaryo et al. 2016, that tend to publish more content with links either to trick the user or to redirect him to a page where the content of the twit is better explained.

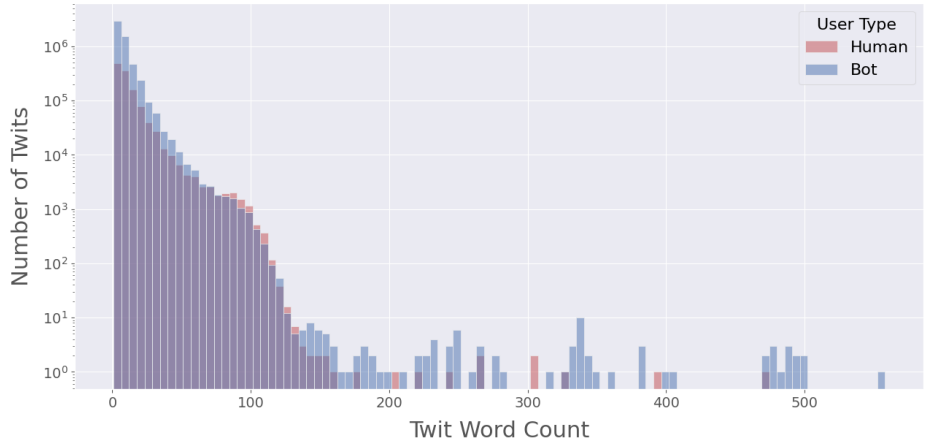

Figure 5.6: Twit word count distribution of each user type

Finally, Figure 5.6 presents the distribution of the number of words present in the twit corpus for each user type. It is notable that the distributions have vary similar shapes, meaning humans and bots have similar behaviour when it comes to number o words written in each post. The plot also shows itself positively skewed, meaning that twits tend to be short in most of the cases. However, it is apparent that bots have a greater proportion of outlier twits with a significantly higher amount of words than humans. Because of this characteristic, the average amount of words in a

bot-generated twit (11.50) is much larger than the average number of words in a human-generated twit (8.19).

5.1.3 Train, Test and Validation Set Split

As previously indicated, the twits dataset's labels are imbalanced. This reality introduces a barrier for predictive modeling as most of the machine learning algorithms used for classification were designed around the assumption of an equal number of examples for each class. Training a model using an imbalanced dataset typically results in poor prediction accuracy, particularly for the minority class, i.e., the model usually turns out to be biased towards the class with the highest representation in the dataset.

To avoid this problem, the train and validation sets were carefully and randomly sampled from the twits dataset in order to keep a 1:1 ratio between bearish and bullish twits within each crypto asset. The test set, on the other hand, is composed by all the labeled twits that are not in the train and validation sets. Table 5.2 displays some proportions of each set.

	Percentage	**Number of Samples**
Train	21.88%	878,748
Test	76.87%	3,086,782
Validation	1.25%	50,196

Table 5.2: Train, test and validation set split

It is important to notice that the sum of the number of samples in each is not equal to the total number of labeled twits collected. This is because some of the twits mention more than one of the cryptocurrencies listed in Table 3.1, resulting in duplicated twits in the dataset. To avoid having a biased model, these duplicates were dropped from the dataset used in the training and evaluation of the sentiment classification model. Additionally, unlabeled twits were also dropped from the dataset as the metrics calculated with the train, test and validation sets rely on the original label of the twits.

5.1.4 Model Training

Having established the architecture behind the classification model, the next step is to build and train the neural network. To do so, it is necessary to define the network training parameters: its optimizer and its loss function.

For the optimizer, when dealing with BERT embeddings it is usually best to work with adaptive optimizes with a low learning rate, thus, to train the network it was used the Adam optimizer with a learning rate of 1E-4. As for the loss function, since it is a binary classification problem, it is standard to use binary crossentropy.

On top of the standard training parameters, the model was also trained with callback functions: Model-Checkpoints (to save the best weights at

every epoch) and Early-Stopping (with patience of 5 based on the loss value, to avoid overfitting).

The model, is then, set to be trained for 10 epochs with a batch size of 512 light cleaned text samples. Figure 5.7 presents the training metrics for both train and validations sets over the epochs.

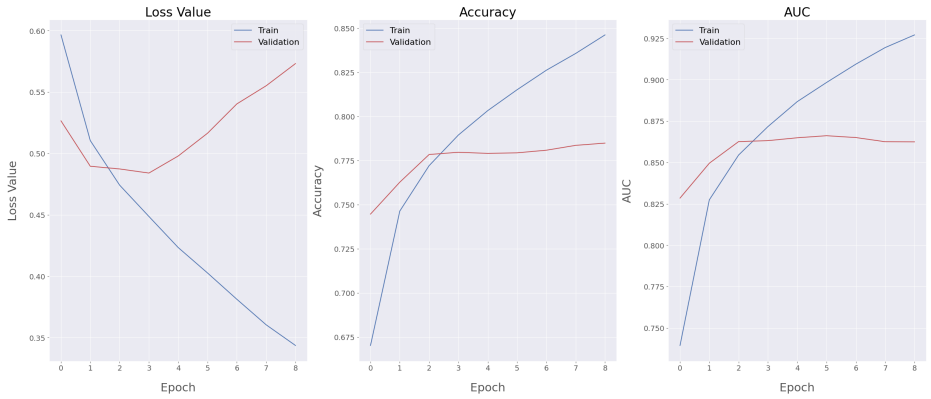

Figure 5.7: BERT based neural network training metrics

When looking at Figure 5.7 it is clear that the Early-Stopping callback was called - since the model didn't train for the 10 epochs it was set to. Until the third epoch, the training and validation losses have a negative trend while both accuracy and AUC metrics have a positive trend. From the fourth epoch on, it becomes very clear that, even though the model keeps improving its training metrics, the validation ones either hit a plateau (accuracy and AUC) or present worse results (loss). This is a very clear signal that the model is overfitting and that we should stop training. Due to the Model-Checkpoint callback, though, the weights obtained on the third epoch (i.e. before the model overfits) were the ones saved.

It can also be observed that the validation accuracy of the saved model is around 77%, which is relatively low when compared to models that use the BERT Base model, but comparable to the results obtained by Devlin et al. 2018 when using similar specs for the embedding layer.

5.1.5 Performance Evaluation

Having trained the classification model, the next step is to verify whether it is able to properly categorize twits as bearish or bullish. The first result to observe is the confusion matrix obtained with the test set, shown in Figure 5.8, as it provides a great visualization of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN).

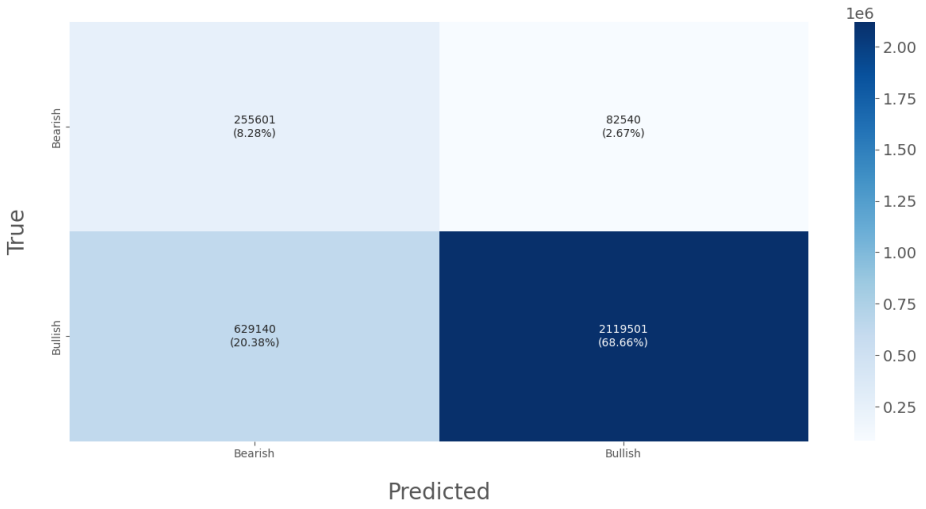

Figure 5.8: BERT based neural network confusion matrix

The confusion matrix shows that the model tends to perform well when predicting bullish twits (as the number of TP is way superior than FP), but the same can't be said when it comes to bearish twits (since the number of TN is inferior than the number of FN). This difference of performance between labels can be seen as a consequence of the imbalance of the test dataset.

To allow more exactitude when it comes to terms used to evaluate the model's performance, the following criteria must be properly defined:

1. **Accuracy:** provides and idea of how well the model is predicting the labels.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

2. **Precision:** a measure of how many of the positive predictions made are correct.

$$\text{Precision} = \frac{TP}{TP + FP}$$

3. **Recall:** is a measure of how many positive cases the classifier predicted correctly out of all the positive cases in the data.

$$\text{Recall} = \frac{TP}{TP + FN}$$

4. **F1-Score:** it is a metric that balances the weighting of the two ratios (precision and recall), needing both to be greater in order for the F1-score value to increase.

$$\text{F1 score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

5. **AUC:** it is given by the area under the receiver operating characteristic (ROC) curve, given by the TP rate over the FP rate.

Having established the metrics used to evaluate the performance of the model, Table 5.3 displays its classification report.

Table 5.3: Classification report of the BERT based neural network

	Precision	**Recall**	**F1-Score**	**Support**
Bearish	0.289	0.756	0.418	338,141
Bullish	0.963	0.771	0.856	2,748,641
Accuracy			0.769	3,086,782
Macro Avg	0.626	0.764	0.637	3,086,782
Weighted Avg	0.889	0.769	0.808	3,086,782

The first thing to notice is the imbalance of the test set, which plays a huge part in some of the metrics. When analysing the precision, it is notable the disparity between the obtained results for each label: while the model tends to have a good precision for bullish twits, the same can't be said for bearish ones. On the other hand, when it comes to recall, both labels have similar results. For the F1-score, given its definition, it makes sense that the bullish label produces better results than the bearish one. Finally, it can be seen that the overall accuracy of the model is around 77%.

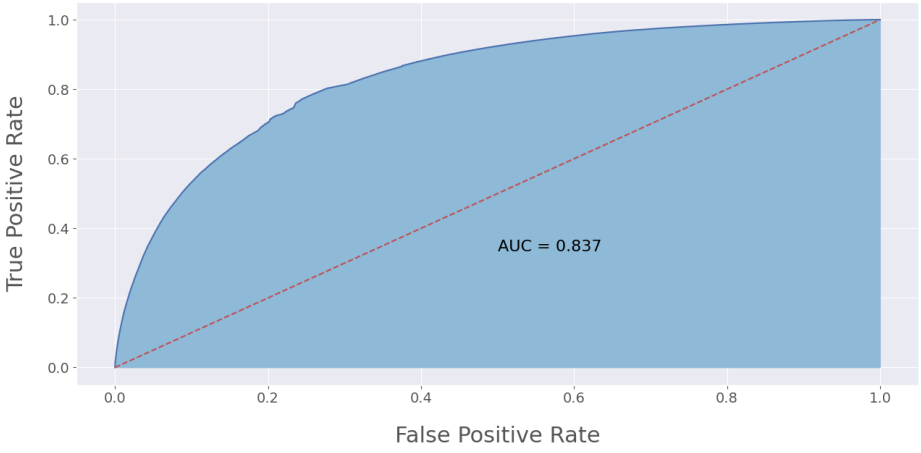

Figure 5.9: BERT based neural network receiver operating characteristic curve

Figure 5.9 illustrates the ROC curve for the test set. The more the blue area resembles a step function (AUC = 1), the better the model's predictions. If the blue area is below the red line (AUC < 0.5) the model is useless when it comes to its predictions. As observed, the classification model obtained an AUC score of 84%, meaning that the model is relatively good at classifying bearish twits as bearish and bullish twits as bullish.

6. Sentiment Signal

The purpose of this stage is to develop an hourly signal for each crypto asset, user type and label based on associated twits. To accomplish so, the NaN labels of unlabeled twits from the processed dataset are replaced with the classification model's predicted labels.

6.1 Engagement Rate

The rise of social media-focused businesses over the last decade has prompted the development of many indicators for analyzing social media postings' performance. These metrics are mainly used by businesses and influencers, but they are applicable to any user in the platform.

The engagement rate is broadly used as the standard measure of a social media post's success. This metric reflects the degree of engagement with followers generated by a user's content. This indicator is defined by:

$$\text{Engagement Rate} = \frac{\text{Total Engagement}}{\text{Total Potential Engagement}}$$

As one can conclude, the definition of the engagement rate is fairly broad, and it is often adapted to different contexts. Based on the data in the twits dataset, the engagement rate is best defined by:

$$\text{Engagement Rate} = \frac{\text{Likes} + \text{Reshares}}{\text{Followers} + 1}$$

The $+1$ in the denominator of the engagement rate formula defined for this research is to avoid addressing $-\infty$ or $+\infty$ to posts made by users with no followers. Table 6.1 displays the average engagement rate for each user type and label.

Table 6.1: Average engagement rate for each user type and label

	Bot	**Human**
Bearish	0.025	0.143
Bullish	0.094	0.311

When it comes to the user type, humans tend to have more engagement in their posts. This is probably due to the fact that human users typically follow their friends on the network to interact with them. In terms of twit label, however, bullish posts tend to have more engagement from other users than bearish posts.

6.2 Aggregated Engagement Rate

This step aims at expressing the average engagement rate for each crypto asset, user type, and label on an hourly basis (i.e. obtaining hourly sentiment signals). Having access to each twit's engagement rate, the signals are obtained by grouping the twits by crypto asset, hour, user type, and label and computing the average engagement rate within each group.

This metric, thus, indicates the average intensity with which users engage with twits that meet the group's criteria. The greater it's value, the more "attention" related posts are receiving in the platform. Figure 6.1 illustrates the behaviour of each one of the calculated signals for the BTC cryptocurrency.

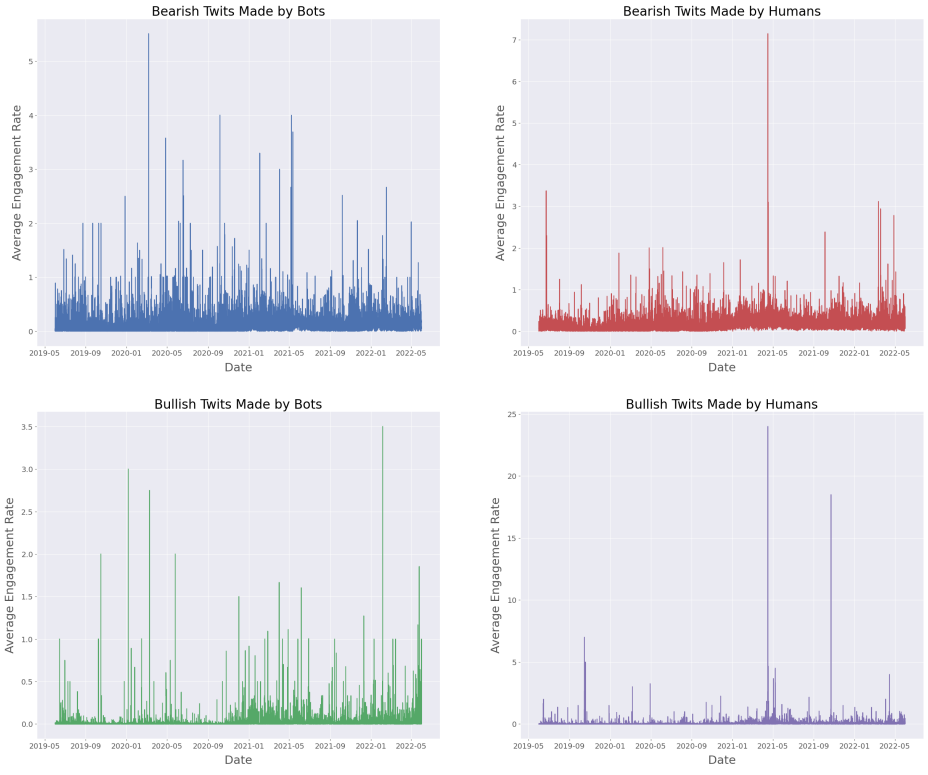

Figure 6.1: Hourly sentiment metrics of the BTC cryptocurrency

The same behaviour observed in Table 6.1 can be seen in Figure 6.1 as well. Additionally, it can be observed that the sentiment signals generated for humans tend to have higher variance than the ones generated by bots. One possible explanation for this phenomenon is that twits published by

bots are frequently liked and shared by the same accounts - which could be other bots attempting to emulate human behavior.

7. Market Effects

The access to the sentiment signals of each cryptocurrency enables the ultimate step of this research: understanding the effects of twits posted by bots on the crypto market. A wide range of metrics are used to quantify/measure the effects of one variable on another. Each one gives distinct insights into how the variables interact with one another and aids in drawing different conclusions.

This analysis is done in three steps: (1) calculating the Pearson Correlation Matrix, (2) obtaining the Granger Causality Matrix and (3) performing a linear regression.

1. **Person Correlation Matrix:** a matrix containing the Person Correlation between the sentiment signals and the corresponding crypto asset returns. This metric allows understanding the strength of the linear relationship between two variables;
2. **Granger Causality Matrix:** a matrix containing the p-values of the Granger Causality Test. This metric allows understanding whether past values of one variable can help predicting another variable. To properly apply the Granger Causality Test, it is necessary to guarantee that all the time-series being handled are stationary, i.e. the statistical properties of the series don't change over time. To verify that, the Augmented Dickey-Fuller (ADF) Test with a 5% significance level is applied to each one of the series. The difference operator is applied to series that fail the stationarity test (i.e. are non-stationary) until they pass the test. To obtain the maximum number of lags used in the Granger Causality Test, fifty different Vector Autoregressive (VAR) models are created - with the number of lags varying from 1 up to 50. The model that produces the best (i.e. lowest) Akaike Information Criterion (AIC) is selected as the best model and its number of lags is used as the maximum lags for the causality test. It is, then, plotted the Granger Causality Matrix containing the minimum p-values for the test between each pair of variables;
3. **Linear Regression:** a linear regression using crypto asset returns as the endogenous variable and the scaled (with MinMaxScaler) sentiment signals as exogenous variables is imple-

mented to verify the magnitude of the coefficients and their statistical significance;

As previously stated, this part of the analysis is conducted only on BTC, ETH and DOGE data. The following considerations influenced the selection of assets: (1) BTC and ETH account for over 60% of the crypto market share, implying that their price movements have a significant impact on the market, (2) DOGE is an asset highly volatile, believed by many to have its price influenced by social media posts (3) all these three cryptocurrencies - as previously shown - have a better quality of data when it comes to gaps in the twits dataset, consequently providing more reliable sentiment signals.

7.1 Bitcoin

Bitcoin (BTC) was the first cryptocurrency to hit the market. It has now become the most well-known cryptocurrency in the world, accounting for a market share of over 40% of the crypto market. As a result, it is not surprising that numerous social media accounts, some of which are controlled by very influential figures such as Elon Musk, speculate and express their sentiment about the asset. Figure 7.1 shows the Pearson Correlation between the BTC associated sentiment signals an the price changes (returns) of the crypto asset.

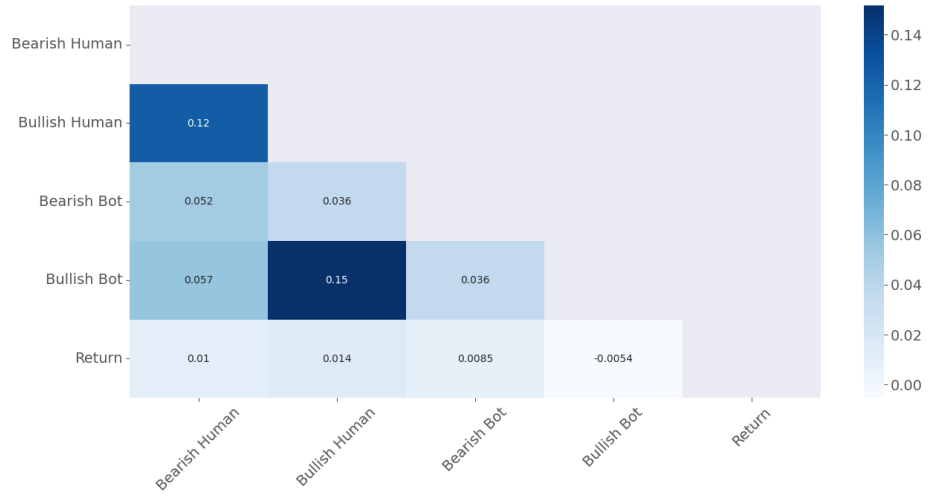

Figure 7.1: Pearson Correlation Matrix of BTC sentiment signals and returns

The correlation matrix indicates a very low Pearson Correlation between sentiment signals, meaning that there is a weak linear relationship between the average engagement rate of bearish and bullish twits posted both by humans and bots. Accordingly, the last line of the matrix shows a very low Pearson Correlation between the sentiment signals and the BTC returns, meaning that BTC's price movements have a weak linear relationship with the sentiment metrics.

Following with BTC's analysis, the Granger Causality Matrix is obtained using all the original time-series (as according to the ADF test they are already stationary), with the number of maximum lags being 25. Figure 7.2 shows the minimum p-values for the causality tests:

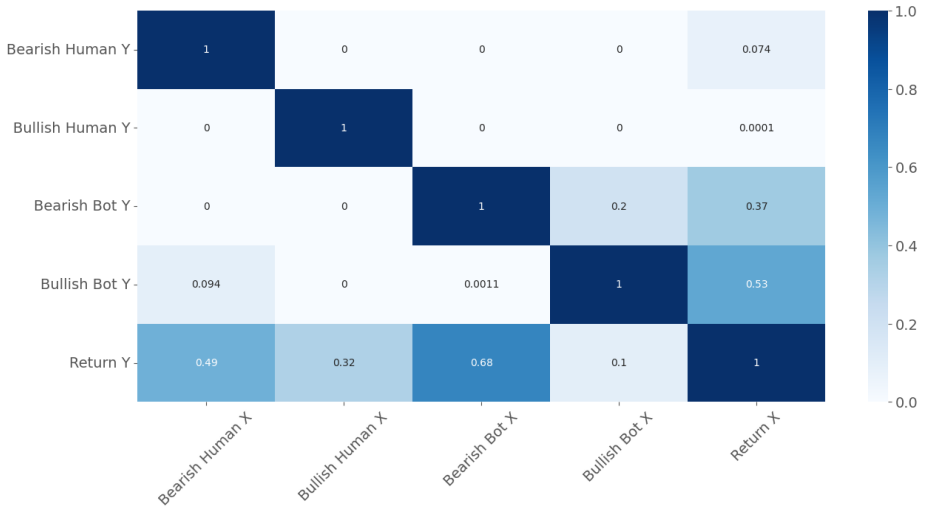

Figure 7.2: Granger Causality Matrix of BTC sentiment signals and returns

The Granger causality Matrix displays the p-values of the tests considering the hypothesis "the variable X causes the variable Y", because of that it may not necessarily result in a symmetric matrix. As one can observe, when it comes to the sentiment signals Granger causing BTC returns (last row of the matrix), the p-values are superior to the significance level of 5% previously established, meaning that past sentiment signals do not help in predicting BTC price movements. Interestingly, however, when observing the last column one can verify that the p-value of returns Granger causing bullish human twits is below the significance level, meaning that past values of BTC returns are a good parameter to predict the average engagement rate of bullish twits posted by humans.

Finally, the following regression is applied to the BTC's scaled sentiment signals and BTC returns:

$$\begin{aligned}\text{Returns} = \text{const} + \alpha_1 \text{Bearish Human} + \alpha_2 \text{Bullish Human} \\ + \alpha_3 \text{Bearish Bot} + \alpha_4 \text{Bullish Bot} + \epsilon\end{aligned}$$

The parameters obtained through this regression can be verified in Table 7.1. Notice, though, that instead of displaying the name of the coefficients as α_1, α_2, α_3 and α_4, the table displays as, respectively, Bearish Human, Bullish Human, Bearish Bot and Bullish Bot.

Table 7.1: Linear regression's results for BTC related time-series

	coef	std err	t	P> |t|	[0.025	0.975]
const	-0.0002	9.88e-05	-1.651	0.099	-0.000	3.06e-05
Bearish Human	0.0029	0.002	1.223	0.221	-0.002	0.008
Bullish Human	0.0068	0.003	2.045	0.041	0.000	0.013
Bearish Bot	0.0031	0.003	1.133	0.257	-0.002	0.009
Bullish Bot	-0.0065	0.006	-1.188	0.235	-0.017	0.004

A quick glance at Table 7.1 shows that only the Bullish Human sentiment signal is statistically significant at the level 5%. Furthermore, the positive value of this coefficient indicates that an increase in the engagement rate of bullish twits posted by humans tend to positively change the price of BTC. The small absolute value of the coefficient, however, indicates that the effect that the engagement rate of bullish twits made by humans have on BTC price changes is very little.

7.2 Ethereum

Ethereum (ETH) is the world's second most well-known cryptocurrency, accounting for more than 19% of the crypto market. This asset's technology serves as the foundation for the development of many other assets. Because of these factors, this cryptocurrency has received a lot of attention on the social media platforms. Figure 7.3 shows the Pearson Correlation between the ETH associated sentiment signals an the price changes (returns) of the crypto asset.

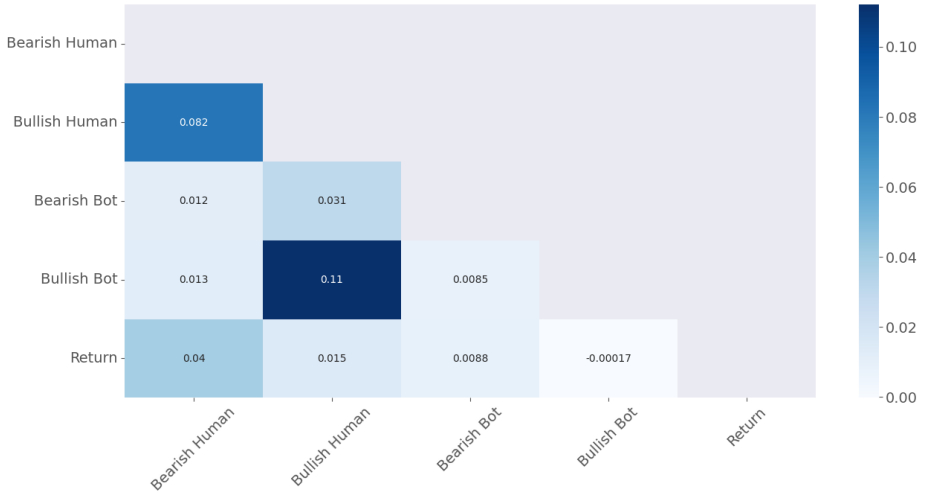

Figure 7.3: Pearson Correlation Matrix of ETH sentiment signals and returns

Once again, the correlation matrix indicates a very low Pearson Correlation between sentiment signals, meaning that there is a weak linear relationship between the average engagement rate of bearish and bullish twits

posted both by humans and bots. Correspondingly, the last line of the matrix shows a very low Pearson Correlation between the sentiment signals and the ETH returns, meaning that ETH's price movements have a weak linear relationship with the sentiment metrics.

Following with ETH's analysis, the Granger Causality Matrix is obtained using all the original time-series (as according to the ADF test they are already stationary), with the number of maximum lags being 17. Figure 7.4 shows the minimum p-values for the causality tests:

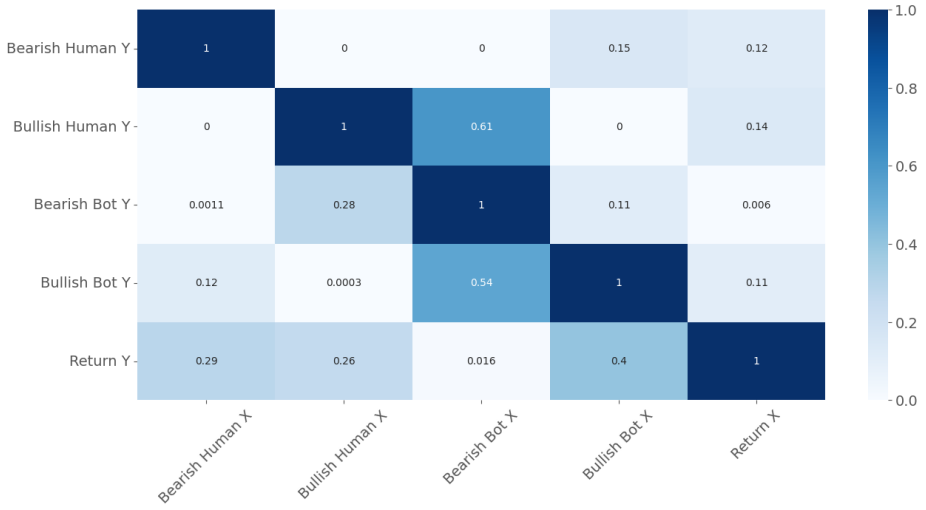

Figure 7.4: Granger Causality Matrix of ETH sentiment signals and returns

As one can observe, when it comes to the sentiment signals Granger causing ETH returns (last row of the matrix), the Bearish Bot signal shows a p-value inferior than the significance level of 5% previously established, meaning its past values can help in predicting ETH price movements. The same can't be said for the other sentiment signals. When observing the last column of the matrix, however, one can verify that the p-values for Bearish Bot is also below the significance level, meaning that past values of ETH returns are a good parameter to predict the average engagement rate of bearish twits posted by bots.

Finally, the following regression is applied to the ETH's scaled sentiment signals and ETH returns:

$$\begin{aligned}\text{Returns} = \text{const} + \alpha_1 \text{Bearish Human} + \alpha_2 \text{Bullish Human} \\ + \alpha_3 \text{Bearish Bot} + \alpha_4 \text{Bullish Bot} + \epsilon\end{aligned}$$

The parameters obtained through this regression can be verified in Table 7.2. Notice, once again, that instead of displaying the name of the coefficients as α_1, α_2, α_3 and α_4, the table displays as, respectively, Bearish Human, Bullish Human, Bearish Bot and Bullish Bot.

Table 7.2: Linear regression's results for ETH related time-series

	coef	std err	t	P> |t|	[0.025	0.975]
const	-0.0008	0.000	-3.033	0.002	-0.001	-0.000
Bearish Human	0.0129	0.004	3.093	0.002	0.005	0.021
Bullish Human	0.0056	0.006	0.960	0.337	-0.006	0.017
Bearish Bot	0.0036	0.006	0.642	0.521	-0.007	0.015
Bullish Bot	-0.0013	0.008	-0.166	0.868	-0.016	0.014

By observing Table 7.2 one can conclude that, between the sentiment metrics, only the average engagement rate of bearish twits posted by humans is statistically significant at the level of 5% when it comes to predicting ETH's price changes. Interestingly, however, its coefficient - even with a low absolute value - is positive, meaning that the higher the engagement rate on bearish twits posted by humans, the more ETH's price tend to raise.

7.3 Doge Coin

Differently from BTC and ETH, Doge Coin (DOGE) is a cryptocurrency that wasn't created to introduce some innovation. In fact it was created as a joke its mostly known for being the most famous "meme coin". The market for this category of crypto assets is marked by its high volatility due to the fact that its value mostly based on speculations. This coin, in special, is also known for being very tied to Elon Musk, Tesla's CEO, as he often publishes tweets (on the Twitter platform) that affects its price. Figure 7.5 shows the Pearson Correlation between the DOGE associated sentiment signals an the price changes (returns) of the crypto asset.

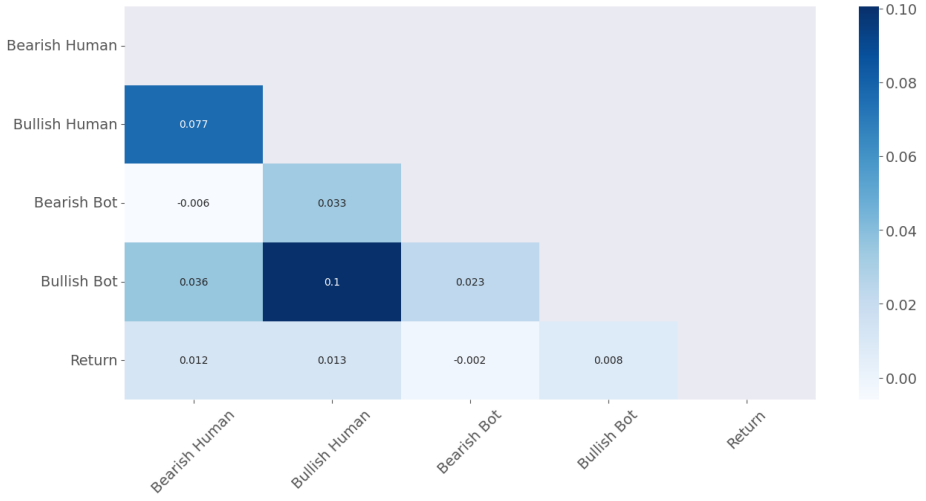

Figure 7.5: Pearson Correlation Matrix of DOGE sentiment signals and returns

Once again, the correlation matrix indicates a very low Pearson Correlation between sentiment signals, meaning that there is a weak linear rela-

tionship between the average engagement rate of bearish and bullish twits posted both by humans and bots. Correspondingly, the last line of the matrix shows a very low Pearson Correlation between the sentiment signals and the DOGE returns, meaning that DOGE's price movements have a weak linear relationship with the sentiment metrics.

Following with DOGE's analysis, the Granger Causality Matrix is obtained using all the original time-series (as according to the ADF test they are already stationary), with the number of maximum lags being 7. Figure 7.6 shows the minimum p-values for the causality tests:

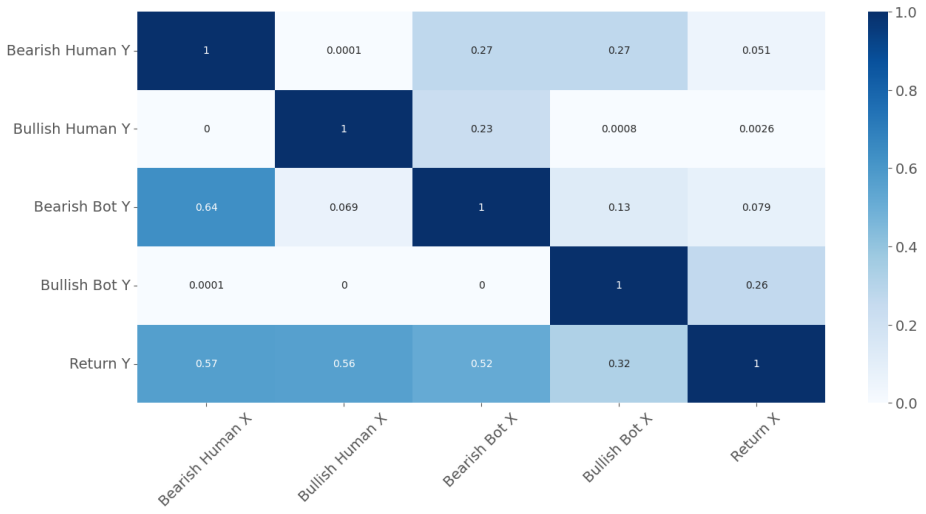

Figure 7.6: Granger Causality Matrix of DOGE sentiment signals and returns

As one can observe, when it comes to the sentiment signals Granger causing DOGE returns (last row of the matrix), the p-values are superior than the significance level of 5% previously established, meaning that past sentiment signals do not help in predicting BTC price movements. When observing the last column of the matrix, however, one can verify that the p-value for the Bullish Human sentiment signal is below the significance level, meaning that past values of DOGE returns are a good parameter to predict the average engagement rate of bullish twits made by humans.

Finally, the following regression is applied to the DOGE's scaled sentiment signals and DOGE returns:

$$\begin{aligned}\text{Returns} = \text{const} + \alpha_1 \text{Bearish Human} + \alpha_2 \text{Bullish Human} \\ + \alpha_3 \text{Bearish Bot} + \alpha_4 \text{Bullish Bot} + \epsilon\end{aligned}$$

The parameters obtained through this regression can be verified in Table 7.3. Notice, once again, that instead of displaying the name of the coefficients as α_1, α_2, α_3 and α_4, the table displays as, respectively, Bearish Human, Bullish Human, Bearish Bot and Bullish Bot.

Table 7.3: Linear regression's results for DOGE related time-series

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
const	-0.0006	0.000	-1.423	0.155	-0.002	0.000
Bearish Human	0.0054	0.006	0.848	0.396	-0.007	0.018
Bullish Human	0.0047	0.005	0.858	0.391	-0.006	0.015
Bearish Bot	-0.0019	0.010	-0.191	0.848	-0.022	0.018
Bullish Bot	0.0028	0.006	0.496	0.620	-0.008	0.014

A quick glance at Table 7.3 shows that none of the sentiment signals are statistically significant when predicting DOGE returns, as all the obtained p-values are superior to the 5% significance level.

8. Conclusion

This study was broken down into five main stages: data collecting and processing, bot identification, sentiment classification, creation of sentiment signals, and market effect analysis. With the exception of the sentiment classification phase, each step is reliant on the preceding one, hence they were carried out in the sequence shown.

In the first step, three main sources of data were considered: CoinMarketCap (cryptocurrency sample space), Binance's API (market data) and Stocktwits' API (social media publications about crypto assets). Each data type was passed through a different data processing algorithm to facilitate the data handling. The twits dataset in special, was divided into two distinct datasets: one holding user information and one containing twits information.

Within the users dataset, inspired by Wang et al. 2021, a variational autoencoder model combined with the k-nearest neighbors algorithm was applied in order to classify users between the "Bot" and "Human" categories. The classification results matches the estimations made by Varol et al. 2017.

The twits dataset was enhanced through the addition of text preprocessing information. The light cleaned text was then sent into the BERT-based neural network, which classified twits with an accuracy of 78%. After that, the deep learning model was used to predict the labels of unlabeled twits in the twits dataset. After labeling all of the twits, a signal was generated using the average engagement rate for each combination of crypto asset, user type, and label.

The generated signals for BTC, ETH, and DOGE were then compared to the returns for the corresponding crypto asset to properly understand the implications of sentiment signals on asset price fluctuations. The Pearson Correlation Matrices, Granger Causality Matrices, and linear regression coefficients were used to make these comparisons. Overall, it is possible to conclude that Stocktwits' average engagement rate has no significant impact on crypto market price movements (both for bot and human users).

9. Future Work

Due to constraints in this study, some of the steps were not conducted in the most optimal way. In order to improve this analysis, the following aspects should be considered during the implementation: (1) use social media data from a more mainstream platform such as Twitter, which contains more relevant data and has more indications of social media posts affecting crypto asset's prices, (2) implement the deep neural network proposed by Basiri et al. 2020 to obtain the sentiment expressed by the collected tweets, (3) use a more complex embedding structure such as BERT Base or BERT Large to better encode the text, (4) adapt the engagement rate metric to the data retrieved by Twitter's API, considering the number of views as the numerator and the number of followers + 1 as the denominator.

References

Alzahrani, Esam and Leon Jololian (2021). "How Different Text-preprocessing Techniques Using The BERT Model Affect The Gender Profiling of Authors". In: *arXiv preprint arXiv:2109.13890*.

Basiri, Ehsan et al. (Oct. 2020). "ABCDM: An Attention-based Bidirectional CNN-RNN Deep Model for sentiment analysis". In: *Future Generation Computer Systems* 115. DOI: 10.1016/j.future.2020.08.005.

Dertat, Arden (2017). *Applied Deep Learning - Part 3: Autoencoders*. URL: https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798 (visited on 10/12/2022).

Devlin, Jacob et al. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. DOI: 10.48550/ARXIV.1810.04805. URL: https://arxiv.org/abs/1810.04805.

EugenioTL (2021). *File:Reparameterized Variational Autoencoder.png*. URL: https://en.m.wikipedia.org/wiki/File:Reparameterized_Variational_Autoencoder.png (visited on 10/12/2022).

Foundation, Wikimedia (n.d.). *Wikimedia Downloads*. URL: https://dumps.wikimedia.org.

Li, Hongchan et al. (2021). "Weibo Text Sentiment Analysis Based on BERT and Deep Learning". In: *Applied Sciences* 11.22. ISSN: 2076-3417. DOI: 10.3390/app112210774. URL: https://www.mdpi.com/2076-3417/11/22/10774.

Mai, Feng et al. (Jan. 2018). "How Does Social Media Impact Bitcoin Value? A Test of the Silent Majority Hypothesis". In: *Journal of Management Information Systems* 35, pp. 19–52. DOI: 10.1080/07421222.2018.1440774.

Mariel, Wahyu Calvin Frans, Siti Mariyah, and Setia Pramana (Mar. 2018). "Sentiment analysis: a comparison of deep learning neural network algorithm with SVM and nave Bayes for Indonesian text". In: 971, p. 012049. DOI: 10.1088/1742-6596/971/1/012049. URL: https://doi.org/10.1088/1742-6596/971/1/012049.

Nguyen, Dat Quoc, Thanh Vu, and Anh Tuan Nguyen (Oct. 2020). "BERTweet: A pre-trained language model for English Tweets". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, pp. 9–14. DOI: 10.18653/v1/2020.emnlp-demos.2. URL: https://aclanthology.org/2020.emnlp-demos.2.

Oentaryo, Richard et al. (Sept. 2016). "On Profiling Bots in Social Media". In: DOI: 10.1007/978-3-319-47880-7.

Ping, Heng and Sujuan Qin (2018). "A Social Bots Detection Model Based on Deep Learning Algorithm". In: *2018 IEEE 18th International Conference on Communication Technology (ICCT)*, pp. 1435–1439. DOI: 10.1109/ICCT.2018.8600029.

Varol, Onur et al. (2017). *Online Human-Bot Interactions: Detection, Estimation, and Characterization*. DOI: 10.48550/ARXIV.1703.03107. URL: https://arxiv.org/abs/1703.03107.

Wang, Xiujuan et al. (June 2021). "Detecting Social Media Bots with Variational AutoEncoder and k-Nearest Neighbor". In: *Applied Sciences* 11, p. 5482. DOI: 10.3390/app11125482.

Zhu, Yukun et al. (Dec. 2015). "Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books". In: *The IEEE International Conference on Computer Vision (ICCV)*.

A. Raw Data Structure

Table A.1: Structure of the collected twits dataset

Name	**DType**
id	int64
body	object
created_at	datetime64[ns, UTC]
user	object
source	object
symbols	object
mentioned_users	object
entities	object
links	object
conversation	object
likes	object
reshares	object
structurable	float64
base_asset	object
owned_symbols	float64
reshare_message	float64

B. Processed Twit Data Structure

Table B.1: Structure of the processed twits dataset

Name	**DType**
id	int64
date	datetime64[ns, UTC]
base_asset	object
user.id	int64
text	object
n_likes	int64
n_reshares	int64
label	float64

C. Processed User Data Structure

Table C.1: Structure of the processed users dataset

Name	**DType**
id	int64
username	object
name	object
avatar_url	object
avatar_url_ssl	object
join_date	datetime64[ns]
official	bool
identity	object
classification	object
home_country	object
search_country	object
followers	int64
following	int64
ideas	int64
watchlist_stocks_count	int64
like_count	int64
plus_tier	object
premium_room	float64
trade_app	bool
trade_status	object
portfolio_waitlist	object
portfolio_status	object
portfolio	object
n_twits	int64

D. Enhanced User Data Structure

Table D.1: Structure of the enhanced users dataset (part I)

Name	**DType**
id	int64
username	object
name	object
avatar_url	int64
join_date	datetime64[ns]
official	int64
followers	int64
following	int64
ideas	int64
watchlist_stocks_count	int64
like_count	int64
plus_tier	int64
premium_room	int64
trade_app	int64
trade_status	int64
portfolio_status	int64
n_twits	int64
suggested	int64
verified	int64
from_us	int64
from_ca	int64
from_in	int64
n_active_days	int64
n_active_days_clipped	int64
twit_freq	float64
idea_freq	float64
url_rate	float64

Table D.2: Structure of the enhanced users dataset (part II)

Name	**DType**
n_words_per_twit	float64
n_assets_per_twit	float64
n_emojis_per_twit	float64
n_stopwords_per_twit	float64
avg_twit_similarity	float64
n_commas_per_twit	float64
n_points_per_twit	float64
n_semicolons_per_twit	float64
n_exclamations_per_twit	float64
n_quotes_per_twit	int64
n_oparentheses_per_twit	float64
n_cparentheses_per_twit	float64