NOVA IMS Information Management School

MGI

Mestrado em Gestão de Informação
Master Program in Information Management

The Predictive Power of Social Media in Cryptocurrencies Value Fluctuation

A sentiment analysis approach on “memecoins”

João Filipe Véstia Guerra

Dissertation report presented as partial requirement for obtaining the Master’s degree in Information Management

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

THE PREDICTIVE POWER OF SOCIAL MEDIA IN CRYPTOCURRENCIES VALUE FLUCTUATION

by

João Filipe Véstia Guerra

Dissertation presented as partial requirement for obtaining the Master's degree in Information Management, with a specialization in Knowledge Management and Business Intelligence.

Advisor: Fernando Bação, Ph.D.

November 2022

ABSTRACT

Since the emergence of Bitcoin in 2009, cryptocurrencies have been an increasingly spoken asset, becoming a global phenomenon. Since Bitcoin, thousands of other coins have been introduced to the public with their value depending on the perception of the market. The high volatility as well as the lack of regulation makes these assets a risky investment, however, the amount of people who believe that cryptocurrencies represent the future of money has never been higher. The reason behind the high volatility can be understood because cryptocurrencies prices depend on factors like technological progress, regulations and legal matters, security issues, political factors, among others that obviously include supply and demand. This high volatility associated with the rising interest on the assets is driving the need of understanding and predicting the impacts that certain factors may have on the value of cryptocurrencies. This research addresses the ability of using data from social media platforms, specifically Twitter to predict changes in memecoins like Dogecoin, by using learning algorithms.

KEYWORDS

Cryptocurrency; Twitter; Google Trends; Sentiment analysis; Machine learning.

INDEX

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. CONTEXT

Cryptocurrencies exist for several years now, however, since the emergence of Bitcoin in 2009, the interest in these assets has skyrocketed and is currently at an all time high, with more than 4000 cryptocurrencies available as of January 2021, and with the cryptocurrencies market being worth approximately 2 trillion dollars in April 2021.

The recent surge of the market is leading many to believe that cryptocurrency is the future of money, and even companies are adapting their strategies, and many are starting to accept cryptocurrencies as a way of payment.

However, many users do not see their crypto assets as "money" but as an investment and the interest in understanding the factors that influence price variations for these assets has never been higher as investors are trying to strategize their way to profitable investments.

Cryptocurrencies themselves are a very recent market and not associated with any regulator, therefore there is still a lot to understand regarding the factors that impact their value and that make these assets have a high risk and high volatility.

This study focusses on social media data as a factor for value fluctuation and will extend the knowledge surrounding existing literature that suggests a link between the sentiment on social media information and price trends.

Previous research has been focused on the most valuable cryptocurrencies such as Bitcoin and Ethereum, however, this study will add to that approach and will focus on altcoins[1] that have recently gained the attention of the public eye and that have been nicknamed "meme coins".

"Meme coins" are cryptocurrencies that have gained popularity in a short amount of time, usually because of influencers or communities promoting them online. For example, Dogecoin was created by Jackson Palmer and Billy Markus in late 2013 and it is considered the original "meme coin". It rose to fame after recently gaining some traction on social media and cryptocurrency communities and, after Elon Musk began tweeting about this cryptocurrency it started receiving a lot of investment and being an ever-increasing trend on social media, having reached the market cap value of 81 billion dollars.

This study will therefore focus on those coins that seem to have relied a lot on social media to achieve their success and one of the goals of this thesis is to compare how social media affects traditional, high value, well established crypto assets like Bitcoin and Ethereum versus how it has impacted the famous "meme coin". The concept of comparing the impact on different types of cryptocurrencies, namely on "meme coins" is something that, to the best of my knowledge, has not been done yet and will therefore create more knowledge on the topic.

[1] Altcoins are alternatives to Bitcoin that build on the success of Bitcoin by slightly changing the rules to appeal to different users.

The literature review also raised the issue of having several data sources, meaning, several social media platforms where the information is extracted from. This study will focus on Twitter, the platform that is normally used on all the literature on the topic, as well as on sentiment analysis score and volume of tweets. Linking this data, I am expecting to achieve enhanced performance on the prediction models when comparing with the current literature.

This study will therefore be of interest to any potential investor on cryptocurrencies as it will analyze a factor that may have an impact on value fluctuations, as well as to anyone who wants to better understand the factors that make these assets highly volatile. It will extend the knowledge on the current literature on this topic, and it will have a practical approach, adding meaning and interpretations to the obtained results from the data collected.

1.2. Research Gap

Research on the topic showed a wide variety of studies with the same objective on established cryptocurrencies like Bitcoin or Ethereum, however there seems to be a gap when it comes to alternative coins, in particular “meme coins” like Dogecoin. This type of asset more recently caught the attention of the public and has been an increasingly spoken asset.

1.3. Research Questions and Objectives

Can social data be used to predict cryptocurrencies price fluctuations?

How do the predicted values for “meme coins”, like Dogecoin, compare against predicted values for established coins, like Bitcoin?

This proposal aims to test the ability to use Tweets volume and sentiment to predict shifts on the value of Dogecoin.

This will then allow to compare against other studies on the literature review and test if less established cryptocurrencies, are more influenced by social media data when compared to the biggest cryptocurrencies like Bitcoin and Ethereum.

1.4. Methods

The objectives will be achieved by using social data from the Twitter API, in the form of raw tweets, to then perform data preprocessing and sentiment analysis, in order to use the daily mean sentiment and daily volume of tweets as variables to predict price.

1.5. Main Results and Conclusions

This study concludes that social data can be used to some degree for the prediction of Dogecoin, and when comparing against different studies on more established coins, presents some better results in terms of model accuracy.

1.6. Structure of the Document

This document is composed of five chapters, with each one describing the necessary steps to answer the research questions and achieve the objectives:

- Chapter 1 – Introduction
- Chapter 2 – Literature Review: Outlines the state of the art in relation to the subject of interest.
- Chapter 3 – Methodology: Details the work done in terms of data collection, processing and results achieve.
- Chapter 4 – Results and Discussion: Shows the results obtained and critically analyzes them by comparison with other work on the topic.
- Chapter 5 – Conclusions and Future Work – summarizes the main contributions of this dissertation and presents possible research paths for further development in the future.

2. LITERATURE REVIEW

2.1. BACKGROUND AND PROBLEM IDENTIFICATION

Cryptocurrencies are volatile as unexpected changes in market sentiment can lead to highly accentuated moves in price. Also, since these assets are not regulated by any financial institution, some countries may block their use due to anti money laundering concerns. Furthermore, as transactions are conducted online, cryptocurrencies have an inherent cybersecurity risk, with investors having to rely upon computer security systems, as well as security systems provided by third parties.

This high volatility associated with the rising interest on the assets is driving the need of understanding and predicting the impacts that certain factors may have on the value of cryptocurrencies.

The specific factor that will be the target of this research is the ability to use, sentiment analysis on available social media to predict shifts on a cryptocurrency value. The increasing use of social media to broadly communicate, spread news, share knowledge, opinions, and ideas regarding every aspect of our society, has been a driver to make modifications in people's behaviors and habits. Furthermore, in cryptocurrency predictions, online platforms are being used as a common place where communities share their opinion on market events.

In view of the current study, there have been previous research and attempts to utilize sentiment from social media platforms to predict fluctuations for some cryptocurrencies. This thesis will build on the ideas of a wide range of research and topics.

Behavioral economists like Daniel Kahneman and Amos Tversky established that decisions, even ones involving financial consequences, are impacted by emotion and not just perceived value (Kahneman & Tversky, 1979). This is one of the indicators for the possibilities to find correlation between sentiment analysis and price variations.

Overall, the literature review allowed to understand that the authors agree on the predictive power of social media information, even though there is a lot of space for improvement in terms of the predictive models used, the cryptocurrencies analyzed or the data sources.

This thesis will therefore focus on sentiment analysis from information extracted from Twitter, and will test their impact on cryptocurrencies like Dogecoin, alternative coins that are also being commonly referred to as "meme coins" and that seem to have had social media as their main impulse towards success and from the very genesis have been linked with social media trends.

2.2. CRYPTOCURRENCIES CHARACTERISTICS/VOLATILITY

According to the Merriam-Webster Dictionary, a cryptocurrency is "any form of currency that only exists digitally, that usually has no central issuing or regulating authority but instead uses a decentralized system to record transactions and manage the issuance of new units, and that relies on cryptography to prevent counterfeiting and fraudulent transactions." (*Definition of CRYPTOCURRENCY*, n.d.).

Cryptocurrency expert Jan Lanksy thinks that cryptocurrencies will replace traditional money in the future. He lists six specifications to define this form of digital money: (Lansky, 2018):

1. "The system does not require a central authority, distributed achieve consensus on its state."
2. "The system keeps an overview of cryptocurrency units and their ownership."
3. "The system defines whether new cryptocurrency units can be created. If new cryptocurrency units can be created, the system defines the circumstances of their origin and how to determine the ownership of these new units."
4. "Ownership of cryptocurrency units can be proved exclusively cryptographically."
5. "The system allows transactions to be performed in which ownership of the cryptographic units is changed. A transaction statement can only be issued by an entity proving the current ownership of these units."
6. "If two different instructions for changing the ownership of the same cryptographic units are simultaneously entered, the system performs at most one of them."

In 2008, Satoshi Nakamoto introduced a purely peer-to-peer version of electronic cash that would allow online payments to be sent directly from one party to another without going through a financial institution (Nakamoto, 2008).

Bitcoin was therefore the first example of a cryptocurrency, since then these assets have been an increasingly spoken topic, becoming a global phenomenon associated with blockchain technology and new economic implications, that would radically threaten the way traditional financial systems work.

Over the past years, hundreds of new cryptocurrencies have been introduced, and with the rising interest, it is important to learn and understand their key characteristics.

Cryptocurrencies mainly differ from traditional currencies as governmental oversight and regulations are not present, therefore there are no taxes or limits that may be harmful to users. Authorities or financial organizations do not restrict the flow of transactions. Unfavorable fees and limitations are reduced as a result.

Adding to this, cryptocurrency exchange rates and movement are not regulated by any institutions. Trading in virtual currencies takes place all around the world. This avoids trade halts following hacking attempts.

All transactions, whether involving personal or business information, are linked to a random string of characters rather than the identity of the owner. The level of demand and supply may be inferred from the popularity of particular virtual currencies. Contracts cannot really be connected to specific persons or businesses.

A private key can be used to protect specific virtual wallets where cryptocurrency can be kept. This indicates that the collected active is exclusively accessible to the owner.

The way that cryptocurrencies are sent is very different from the way that conventional currencies are transmitted. Transfers of virtual money happen relatively instantly and are not location-dependent.

Finally, holders can utilize their cryptocurrencies through the tools and services that are quickly emerging. It is now possible to exchange cryptocurrencies traditional currencies. Through conversion and exchange solutions, these currencies may be financed straight from the users digital wallet.

With the emerging interest, some risks have been associated with the assets, making it high-risk and speculative and enhancing the importance of understanding the factors deriving value shifts and volatility.

How much an asset's price has fluctuated over time, or how volatile it has been, is measured by volatility. Generally speaking, the riskier an asset is to invest in, the more potential it has to deliver bigger returns or higher losses over shorter time periods than assets that are generally less volatile.

What makes cryptocurrencies so unstable? The trajectory of this market is influenced by a few different variables.

Cryptocurrencies are volatile as unexpected changes in market sentiment can lead to highly accentuated moves in price. Also, since these assets are not regulated by any financial institution, some countries may block their use due to anti money laundering concerns. Furthermore, as transactions are conducted online, cryptocurrencies have an inherent cybersecurity risk, with investors having to rely upon computer security systems, as well as security systems provided by third parties.

These are just some of the factors that have a direct impact on the market price, however there are more. In 2017, Obryan Poyser proposed that the Bitcoin price drivers could be split in internal factors (supply and demand), as well as external factors (attractiveness and adoption; Macro-financial). (Poyser, 2017)

As a still-emerging sector, cryptocurrency is growing quickly in both popularity and investor disillusionment. Despite all the media coverage, this market is still insignificant when measured against conventional money or even gold. The trade can therefore be influenced by even smaller influences, such as a group of individuals that own a significant quantity of cryptocurrency.

Adding to this, the speculative industry dominates the market. In order to profit, investors wagered on whether the price would rise or fall. High volatility results from these speculative bets' abrupt inflow or outflow of capital.

The majority of cryptocurrencies, are solely digital assets with no physical backing. Which implies the rules of supply and demand are solely responsible for determining their price. In the absence of any additional stabilizing element, such as government support, a variety of factors may cause a change in supply or demand.

Furthermore, currencies' underlying blockchain or other alternative technologies are still developing. When a smart contract is not validated within the anticipated timescale, there is a scalability issue that leads to abrupt downward pressure.

Finally, this market is not viewed as requiring knowledge, in contrast to real estate or the stock market. Thus, it is mostly being invested in by non-professionals who tend to become impatient and leave the endeavor when they don't succeed in their expectations of making rapid progress. Volatility is also a result of this constant participation and dismissal of users.

Despite the risk, the interest in these assets has never been higher, with the number of crypto asset users indicating a total of up to 101 million unique users across 191 million accounts opened at service providers as of Q3 2020, according to the 3rd Global Cryptoasset Benchmarking Study from the University of Cambridge (Blandin et al., 2020).

2.3. Existing Approaches for Predicting Price Fluctuation

Connor Lamon, Eric Nielsen, Eric Redondo analyzed the ability of news and Twitter data to predict price fluctuations for three cryptocurrencies: Bitcoin, Litecoin and Ethereum. On their study daily news and social media data was labeled based on actual price changes one day in the future for each coin, rather than on positive or negative sentiment, this way the model could predict price fluctuations without first predicting the sentiment. Their model was generally able to predict the largest (magnitude) price increases and decreases correctly, and the authors suggested strategies for future work like integrating more types of media sources and experiments with different models like neural networks (Lamon et al., 2017).

Jethin Abraham, Daniel Higdon, John Nelson and Juan Ibarra analyzed tweets to find that tweet volume, rather than tweet sentiment (which is invariably overall positive regardless of price direction), is a predictor of price direction for Bitcoin and Ethereum. By utilizing a linear model that takes as input tweets and Google Trends data, they were able to accurately predict the direction of price changes. They also suggested the use of more complex models instead of just linear ones (Abraham et al., 2018).

Lars Steinert and Christian Herff applied a different approach in comparison to the existing literature by not focusing on Bitcoins but many different altcoins. To do this, they collected a dataset containing prices and the Twitter activity for 181 altcoins. The containing public mood was then estimated using sentiment analysis. To predict altcoin returns, they carried out linear regression analyses and showed that short-term returns can be predicted from activity and sentiments on Twitter (Steinert & Herff, 2018).

Ivo Bernardo, Roberto Henriques and Victor Lobo tested whether sentiment drives price or price drives sentiment. In their study it was used data from Twitter and the activity for some stock markets and tested the reaction time to prove that for some companies the reaction time is very short (intraday predictions are available), and for others it is longer (day-to-day). (Bernardo, I et al., 2017).

2.4. ALGORITHMIC APPROACHES

A range of machine learning algorithms and a set of syntactic and dialectal properties are the foundation of machine learning methodologies. This approach views sentiment analysis as a difficulty in consistently classifying texts (Medhat et al., 2014).

This method predicts the outcome using statistical analysis and the emotion polarity as the main input data. Unsupervised and supervised learning techniques make up the remaining divisions. (Rodrigues et al., 2018)

Machine learning is a predictive technique that predicts the categorization of the present information based on prior discoveries. There are several algorithms in use today, and new ones keep developing as data analytics directly contribute to technological advancements (Gandomi et al., 2015). These methods may also be used to successfully categorize text.

Sentiment Analysis

The act of locating feelings in text and categorizing them as positive, neutral, or neutral according to the emotions conveyed is known as sentiment analysis. Understanding how a given subject or group of subjects feel about any issue may be achieved by utilizing NLP techniques to evaluate subjective and unstructured data via sentiment analysis.

There are several tools available for extracting sophisticated characteristics from text. The majority of these technologies, nevertheless, are difficult to use and take time to fully explore their possibilities. In the current study, tweets will be classified using VADER (Hutto et al., 2014) to identify their polarity.

Machine Learning Approaches

The machine learning techniques are described by the authors in this study as a group of tactics that include training an algorithm with training data before using it on target data, commonly referred to as testing data. (Devika et al., 2016)

The supervised learning technique uses labelled data with known inputs and outputs to train algorithms before classifying unknown data. The NB Classifier and Decision Tree Classifier are two of the most used and well-liked supervised learning approaches. (Kamath et al., 2013)

The authors of this study describe the NB classifiers as a straightforward Bayesian model with a conditional independence assumption - the accuracy of the classifier is not significantly affected by this assumption, but it does speed up the classification process.

On the other hand, the same paper depicts the decision tree classifiers as a tree, in which the internal nodes represent the features, the edges indicate the tests to be run on the feature weights, and the leaf nodes represent the categories that arise from the above. (Joshi et al., 2014)

In numerous voice and language processing applications, decision trees have been employed. According to its definition, Random Forest is a supervised classification algorithm. It is an ensemble learning method based on decision tree algorithms that has gained popularity in recent years due to

its superior performance and robustness over other machine learning algorithms like SVM or even NB (Breiman et al., 1996).

Lexicon-based Approaches

The use of traditional supervised models may be rendered useless in situations where training data is insufficient. Lexicon-based systems need a corpus of pre-compiled sentiment lexicon words, each of which has a sentiment value given (). The creation of these lexicons might be done manually or mechanically.

The authors of this work distinguish between two different lexicon-based methodologies: unsupervised models, which do not need a training corpus of labelled texts, and mixed models, which integrate lexical knowledge with tagged documents. (Bonta et al., 2019)

In this study, the VADER lexicon-based algorithms is the one that is going to be used.

VADER is a rule-based model for general sentiment analysis and compared its effectiveness to 11 typical state-of-the-practice benchmarks, including Affective Norms for English Words (ANEW), Linguistic Inquiry and Word Count (LIWC), the General Inquire, Senti WordNet, and machine learning oriented techniques that rely on NB, Maximum Entropy, and SVM algorithms.

3. METHODOLOGY

3.1. DATA COLLECTION

3.1.1. Historical Currency Data

The market data was collected from Kaggle. Dogecoin historical value was retrieved for the year 2021 at a daily granularity, and include for each day:

- Open Value - Price from the first transaction of a trading day.
- High Value - Maximum price in a trading day.
- Low Value - Minimum price in a trading day.
- Close Value - Price from the last transaction of a trading day.
- Adj Close Value - Closing price adjusted to reflect the value after accounting for any corporate actions.
- Volume - Number of units traded in a day.

3.1.2. Historical Twitter Data

Twitters' Application Programming Interfaces (API) was used to extract social data.

The Twitter (API) provides programmatic access to the Twitter platform, without having to enter the website interface. The tool allows for the conduction of various tasks such as retrieving, creating, updating, sorting, filtering, and removal of data from the platform.

Using Python package Tweepy, an open source Python package that gives a very convenient way to access the Twitter API with Python, data was obtained in the form of raw tweets by applying the following search parameters:

1. Has been created during the year of 2021.
2. Contained the Dogecoin coin tag (i.e., $DOGE), giving a high degree of confidence that the tweet is related to the cryptocurrency and not another topic (for example the dog race).
3. Is written in English, to optimize the functioning of the sentiment analysis tool we will be using.
4. Is not duplicated: while re-tweets were allowed as this may signal a sentimental trend, duplicated tweets were not taken in consideration as they are mostly from bot accounts.

The following relevant objects were retrieved for each tweet:

Object	Title
author_followers	Followers count.
author_tweets	Tweet count.
author_location	The location specified in the user's profile, if the user provided one.
tweet id	The unique identifier of the requested Tweet.
text	The actual UTF-8 text of the Tweet.
created_at	Creation time of the Tweet.
retweets	Retweet count.
replies	Replies count.
likes	Likes count.
quote_count	Quotes count.

Table 1. Tweet field description

3.2. DATA PRE-PROCESSING

Data preprocessing is a data mining approach to create cleaner, more useful information from the raw data acquired from various sources.

The integrity of the dataset is compromised by errors, redundancies, missing values, and inconsistencies, therefore we must address all of these problems for a more accurate result.

As the purpose of this study is to analyze the sentiment and opinion of individuals, it is important to process the data to make sure we get the most value from it.

This phase includes steps like:

- Noise cleaning – cleaning the raw tweets by removing URL, usernames and hashtags.
- Tokenizing – breaking a text into distinct words.
- Lower-case conversion – converting all tokens into lowercase.
- Removing stop words – removing meaningless words in comments.
- Stemming – reducing inflectional forms and transforming the derivationally forms of a word to a common base.

3.3. SENTIMENT ANALYSIS

“Sentiment Analysis or Opinion Mining is the computational study of opinions, sentiments and emotions expressed in text” - Bing, Liu (2010).

The text of the tweets on cryptocurrencies was subjected to VADER in order to conclude if an incoming message's underlying sentiment is favorable, negative, or neutral.

VADER stands for (Valence Aware Dictionary and Sentiment Reasoner). It was created by C.J. Hutto & E.E. Gilbert at the Georgia Institute of Technology. It is a lexicon and rule-based sentiment model specially created for texts in social media. It has over 9,000 words, and every word was marked by ten independent people from -4 (extremely negative) to 4 (extremely positive) and after that, the final score is the average of all 10 scores. (Hutto, C. et al., 2014)

VADER was chosen as it is an open-source tool that has been human-validated, is customized particularly for Twitter material, and requires few time consumption.

The sentiment score was calculated for each tweet and a daily sentiment mean was then calculated based on that score.

3.4. PREDICTING PRICE VARIATIONS

A time series of market and Twitter data made up the data set. The data set was split 70/30 for training and testing, with 70% of the data set being set aside for training and 30% being used for testing.

The models used were the Random Forest Regressor Model and the XG Boost Regressor Model, as they are more complex models, that were suggested in the literature review as potential good models for better predictions.

The first step towards preparing the data was to create the input features X and the target vector y. We used the window_data() function to create these vectors.

- X: The input features vectors.
- y: The target vector.

The function has the following parameters:

- df: The original DataFrame with the time series data.
- window: The window size in days of previous closing prices that will be used for the prediction - after testing, the optimal amount of days was 3 for both models.
- feature_col_number: The column number from the original DataFrame where the features are located. The features considered are the daily sentiment score and the daily volume of tweets.
- target_col_number: The column number from the original DataFrame where the target is located. The target feature is the Adj Close price as it was the feature that delivered a better accuracy in both models.

To evaluate the models the two indicators that will be used are the Root Mean Squared Error and the R-squared:

- Root Mean Squared Error - the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit. Root mean square error is commonly used in climatology, forecasting, and regression analysis to verify experimental results.
- R-squared - is a statistical measure in a regression model that determines the proportion of variance in the dependent variable that can be explained by the independent variable. In other words, r-squared shows how well the data fit the regression model (the goodness of fit).

4. RESULTS AND DISCUSSION

The market data collected revealed the following trends in terms of close price and traded volume over time:

Figure 1. Adj close Price over time

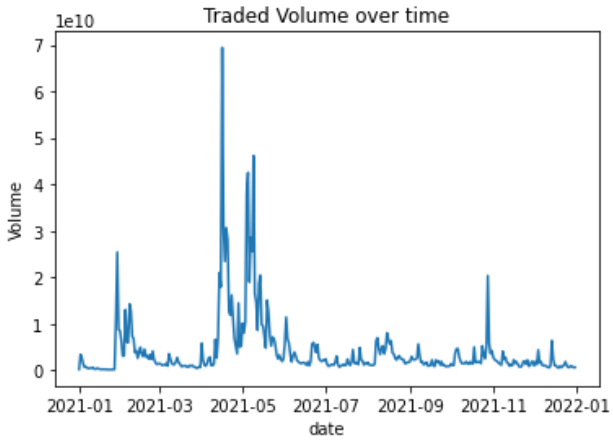

Figure 2 . Traded volume over time

As for the tweets, a total of 2 506 320 tweets were collected for the year 2021.

```
<Tweet id=1466195370778767366 text=@cz_binance @elonmusk @binance #DOGEorTesla #Binance
How to sleep with this indecision?
A Tesla would be fun, but 50,000 in DOGE is a good investment.
I will probably convert everything to BNB and BUSD to trade and increase my portfolio.
To facilitate the negotiation, I will choose $50,000 in $DOGE https://t.co/OqNL7Ozc2o>
```

Figure 3. Example of a retrieved tweet.

The sentiment score was calculated for each tweet and a daily sentiment mean was then calculated based on that score.

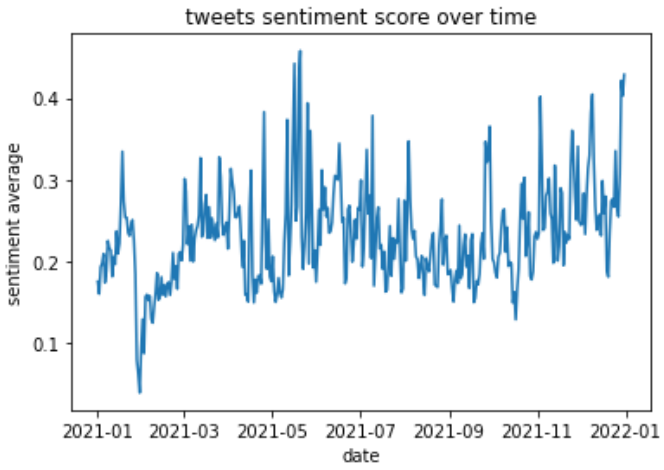

Figure 4. Tweets sentiment score over time

4.1. PREDICTING PRICE VARIATIONS

Random Forest Regressor Model

The Random Forest Classifier is an ensemble approach, which means that it consists of several little decision trees, known as estimators, each of which makes a separate prediction. The estimators' estimates are combined by the random forest model to yield a more precise forecast. This model returned the following results:

```
Root Mean Squared Error: 0.11724172339117649
R-squared : 0.7688046703104161
```

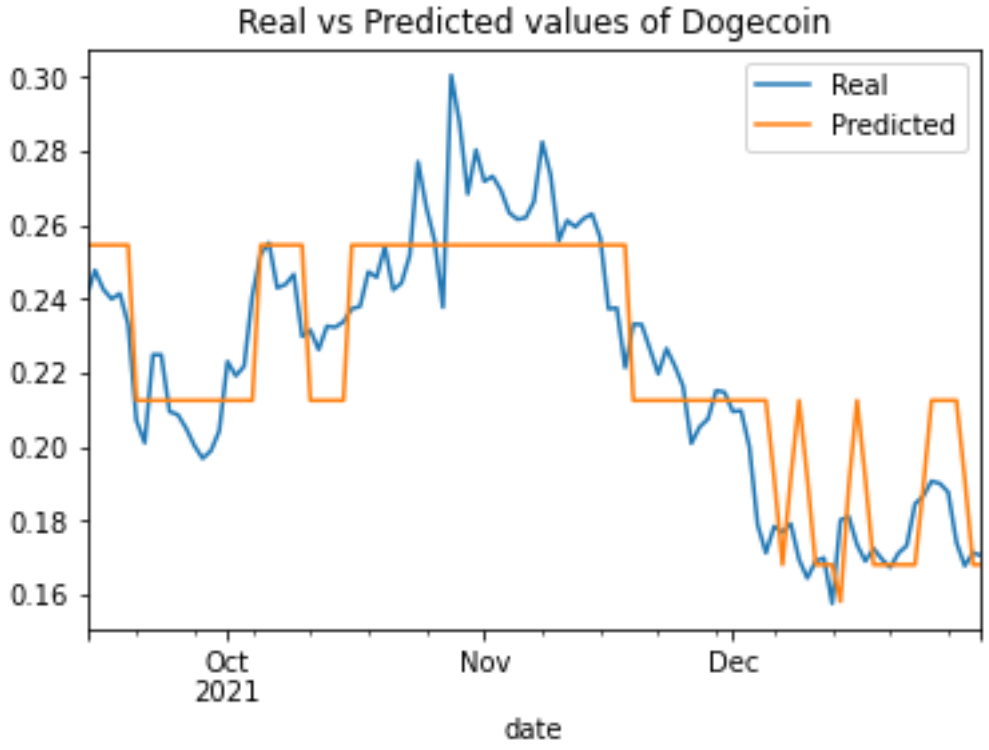

Figure 5. Real vs Predicted values of Dogecoin using Random Forest Regressor Model

XG Boost Regressor Model

Gradient Booster algorithm builds trees one at a time, where each new tree helps to correct errors made by previously trained tree. The model gets more expressive as additional trees are added. Each tree constructed is normally shallow, and there are three criteria to consider: learning rate, number of trees, and depth of trees.

Extreme Gradient Boosting (XG Boost) is an efficient implementation of the gradient boosting machine learning algorithm. This model returned the following results:

```
Root Mean Squared Error: 0.10061733880544414
R-squared : 0.8297213012783521
```

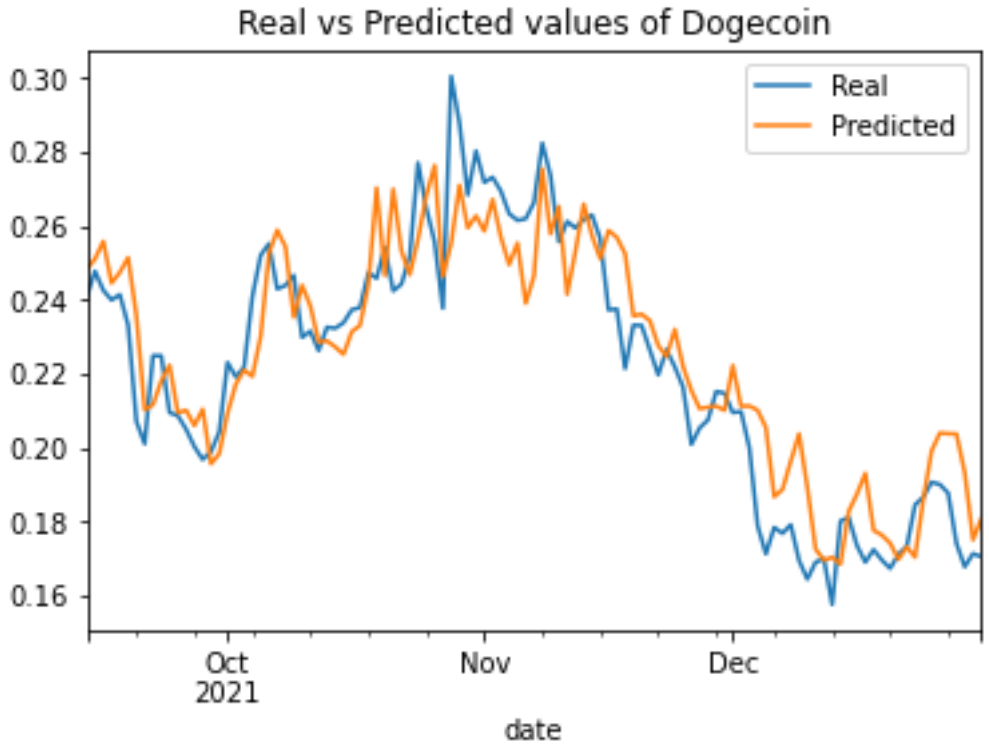

Figure 6. Real vs Predicted values of Dogecoin using XG Boost Regressor Model

As accordingly to the studies on the literature review, the results lead to believe that social data from Twitter, particularly the sentiment analyses on tweets and the volume of tweets can be used to predict cryptocurrencies values.

Additionally, when comparing to models developed in other studies on more established coins, such as Bitcoin or Ethereum, the model developed here seems to have a better accuracy, which leads to believe that indeed less established coins can be more influenced by social data and social trends.

Furthermore, the sentiment analysis results tend to be overly positive every day, however that did not affect largely the models accuracy even in a testing time that coincided with a notable decrease in price.

5. CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORKS

Although the results somewhat match the objectives of the study, and seemingly Dogecoin is more influenced by social data than well-established crypto, it can be dangerous to assume that social data on its own is driving the value of this assets and that the main reason for price surges or decreases is due to social trends. As for many different assets, there are a lot of factors that drive their value, and the search to understanding what this factors are and how they influence cryptocurrencies will surely be an on-going and ever growing topic that will have more interest in the upcoming years.

The work done on this project was subject to a time constraint, that being one of the main limitations. Another limitation is due to the limits of the Twitter API, which only lets extracts an amount of 10 million tweets per month, making it hard to do a study on multiple coins, something that I believe would be useful in future work in order to directly compare if the same exact model delivers different results in differently established coins.

Additionally, for future work, there could be some more efforts put into the sentiment analysis, due to time constraints I used a lexicon based algorithm and the accuracy could improve if a supervised learning algorithm was used here. Also, there could be some more work done in terms of data cleaning as there are a huge amount of bots tweeting on the topics, so for future work the identification of these bots could open an interesting dilemma on how the results of real public opinions on its own compare to the current results.

Finally, it could be interesting to test time-series predictability, by having hourly models and seeing if they deliver better results.

For future work, this study can be applied to different coins and can be implemented in a live platform that collects and predicts the value in real time.

6. BIBLIOGRAPHY

Abraham, J., Higdon, D., Nelson, J., & Ibarra, J. (2018). *Cryptocurrency Price Prediction Using Tweet Volumes and Sentiment Analysis*. *1*(3), 22.

Blandin, A., Pieters, G. C., Wu, Y., Dek, A., Eisermann, T., Njoki, D., & Taylor, S. (2020). 3rd Global Cryptoasset Benchmarking Study. *SSRN Electronic Journal*.

Kahneman, D., & Tversky, A. (1979). Prospect Theory: An Analysis of Decision under Risk. *Econometrica*, *47*(2), 263–291.

Lamon, C., Nielsen, E., & Redondo, E. (2017). *Cryptocurrency Price Prediction Using News and Social Media Sentiment*.

Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. 9.

Poyser, O. (2017). *Exploring the determinants of Bitcoin's price: An application of Bayesian Structural Time Series*. 47.

Steinert, L., & Herff, C. (2018). Predicting altcoin returns using social media. *PLOS ONE*, *13*(12)

Bernardo, I., Henriques, R., Lobo, V. (2017). Social Market: Stock Market and Twitter Correlation.

Medhat, W., Hassan, A., & Korashy, H. Sentiment analysis algorithms and applications: a survey. Ain Shams Eng. J. 5 (4), 1093–1113 (2014).

Rodrigues, R., Camilo-Junior, C. G., & Rosa, T. (2018). A taxonomy for sentiment analysis field. International Journal of Web Information Systems, 14(2), 193-211.

Gandomi, H. (2015). Gandomi A., Haider M. Beyond the hype: Big data concepts, methods, and analytics, International Journal of Information Management, 35(2), 137-144.

Hutto, C., Gilbert, E.: Vader: A parsimonious rule-based model for sentiment analysis of social media text. In Proceedings of the International AAAI Conference on Web and Social Media (Vol. 8, No. 1) (2014).

Devika, M. D., Sunitha, C., & Ganesh, A. (2016). Sentiment analysis: a comparative study on different approaches. Procedia Computer Science, 87, 44-49.

Kamath, S. S., Bagalkotkar, A., Kandelwal, A., Pandey, S., & Poornima, K. (2013, April). Sentiment analysis based approaches for understanding user context in web content. In 2013 International Conference on Communication Systems and Network Technologies (pp. 607-611). IEEE.

Joshi, N. S., & Itkat, S. A. (2014). A survey on feature level sentiment analysis. International Journal of Computer Science and Information Technologies, 5(4), 5422-5425.

Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.

Bonta, V., & Janardhan, N. K. N. (2019). A comprehensive study on lexicon based approaches for sentiment analysis. Asian Journal of Computer Science and Technology, 8(S2), 1-6.

7. APPENDIX

7.1. TWEEPY API SCRIPT

```
'created_at': tweet.created_at,
'retweets': tweet.public_metrics['retweet_count'],
```

```
import tweepy
import time
import pandas as pd

client =
tweepy.Client('AAAAAAAAAAAAAAAAAAAAAD3sXQEAAAApdlaHRR5QtChRDXVhqQXmNuGdao%
3DS19j6DY5sXcAZb7zgYz8lVolHiFIxJkY6QKBCcV16BcBNyDp04',
wait_on_rate_limit=True)

hoax_tweets = []
for response in tweepy.Paginator(client.search_all_tweets,
query = '$DOGE -is:retweet lang:en',
user_fields = ['public_metrics', 'location'],
tweet_fields = ['created_at', 'public_metrics', 'text', 'id'],
start_time = '2021-01-01T00:00:00Z',
end_time = '2022-12-01T00:00:00Z',
max_results=500):
time.sleep(1)
hoax_tweets.append(response)

hoax_tweets[0].data[0]

hoax_tweets[0].includes['users'][2]

hoax_tweets[0].includes['users'][2].description

result = []
user_dict = {}
# Loop through each response object
for response in hoax_tweets:
# Take all of the users, and put them into a dictionary of dictionaries
with the info we want to keep
for user in response.includes['users']:
user_dict[user.id] = {'username': user.username,
'followers': user.public_metrics['followers_count'],
'tweets': user.public_metrics['tweet_count'],
'description': user.description,
'location': user.location
}
for tweet in response.data:
# For each tweet, find the author's information
author_info = user_dict[tweet.author_id]
# Put all of the information we want to keep in a single dictionary for
each tweet
result.append({'author_id': tweet.author_id,
'username': author_info['username'],
'author_followers': author_info['followers'],
'author_tweets': author_info['tweets'],
'author_description': author_info['description'],
'author_location': author_info['location'],
'tweet id': tweet.id,
'text': tweet.text,
```

```
'replies': tweet.public_metrics['reply_count'],
'likes': tweet.public_metrics['like_count'],
'quote_count': tweet.public_metrics['quote_count']
})

# Change this list of dictionaries into a dataframe
df = pd.DataFrame(result)
df

# Adjust time zone from columns
df['created_at'] = df['created_at'].dt.tz_localize(None)
df

df.to_csv("DogecoinTweets.csv", index = False)
```

7.2. DATA PRE-PROCESSING & SENTIMENT ANALYSIS SCRIPT

```
import pandas as pd
import csv
import numpy as np
import nltk
import re,string
from bs4 import BeautifulSoup
from nltk.tokenize import word_tokenize
from nltk.tokenize import sent_tokenize
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer
import seaborn as sns
from rake_nltk import Rake
from collections import Counter
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
import matplotlib.pyplot as plt
import glob
import os
# Load dataset

path = r'C:\Users\joaof\Dropbox\Tese\2. Data Sources\Dogecoin'
# use your path
all_files = glob.glob(os.path.join(path, "*.csv"))     # advisable to use
os.path.join as this makes concatenation OS independent

df_from_each_file = (pd.read_csv(f) for f in all_files)
datatweet   = pd.concat(df_from_each_file, ignore_index=True)
# doesn't create a list, nor does it append to one
# Add column date to  Dataframe
datatweet['date'] = pd.to_datetime(datatweet['created_at']).dt.date
datatweet
#Total Number of Tweets Retrieved
len(datatweet)
# Text preprocessing
def textPreProcess(rawText, removeHTML=True, charsToRemove =
r'\?|\.|\@|\/|\!|\;|\.|\www|\%|\"|\,|\(|\)|\&|\:|\-', removeNumbers=True,
removeLineBreaks=False, specialCharsToRemove = r'[^\x00-\xfd]',
convertToLower=True, removeConsecutiveSpaces=True):
    cleanedText = []
    for x in (rawText[:]):
```

```
        # Remove HTML
        if removeHTML:
            procText = BeautifulSoup(x,'html.parser').get_text()

         # Remove punctuation and other special characters
        if len(charsToRemove)>0:
            procText = re.sub(charsToRemove,' ',procText)

        # Remove numbers
        if removeNumbers:
            procText = re.sub(r'\d+',' ',procText)

        # Remove line breaks
        if removeLineBreaks:
            procText = procText.replace('\n',' ').replace('\r', '')

        # Remove special characters
        if len(specialCharsToRemove)>0:
            procText = re.sub(specialCharsToRemove,' ',procText)

        # Normalize to lower case
        if convertToLower:
            procText = procText.lower()

        # Replace multiple consecutive spaces with just one space
        if removeConsecutiveSpaces:
            procText = re.sub(' +', ' ', procText)

        # If there is a text, add it to the clean text
        if procText != '':
            cleanedText.append(procText)
    return cleanedText
# Tokenize texts
def tokenize_words(texts):
    words_new = []
    for w in (texts[:]):
        w_token = word_tokenize(w)
        if w_token != '':
            words_new.append(w_token)
    return words_new
# Stemm texts
def stemming(words):
    procText = []
    for w in (words[:]):
        stemmed_word = [snowball.stem(x) for x in (w[:])]
        procText.append(stemmed_word)
    return procText
#generating a new dataframe only with the column PreProcessedText keeping
the default index
processedtweets =  pd.DataFrame(data=textPreProcess(datatweet.text),
index=datatweet.index, columns=['PreProcessedText'])
processedtweets
# Create sentiment analysis object
analyser = SentimentIntensityAnalyzer()
```

```
# To test, let's evaluate first sentence of third tweet (1st and 2nd are
just 1 word-long tweets)
# Scales:
#   compound: -1:most extreme negative, 1:most extreme positive
#     positive: compound >=0.05
#     neutral: -0.05<compound<0.05
#     negative: compound <= -0.05
#   pos, neu, neg: proportion of text that are positive, neutral or
negative
score = analyser.polarity_scores(datatweet['text'][0])
print(datatweet['text'][2],score)
# Process sentiment for all sentences
all_scores = []
for t in (datatweet['text'][:]):
  score = analyser.polarity_scores(t)
  all_scores.append(score)
datatweet['Sentiment'] = [c['compound'] for c in all_scores]
# Analysis examples:
# Mean by Sentiment - Most Optimistic Tweets
ex2 = datatweet.groupby('text')['Sentiment'].mean().to_frame()
ex2.sort_values("Sentiment", ascending=False).head(20)
# Analysis examples:
# Mean by Sentiment - Most Negative Tweets
ex2 = datatweet.groupby('text')['Sentiment'].mean().to_frame()
ex2.sort_values("Sentiment", ascending=True).head(20)
# Compute review's sentiment as the mean sentiment from its sentences
meanByReview = datatweet.groupby('date')['Sentiment'].mean()

# Consider sentences with no result as neutral (0)
meanByReview = meanByReview.fillna(0)

# Add column Sentiment to reviews Dataframe
datatweet['Sentiment'] = meanByReview[datatweet['date']].values
Polarity
# Assign a qualitative evaluation to the review
bins = pd.IntervalIndex.from_tuples([(-1.1, -0.05), (-0.05, 0.05), (0.05,
1)], closed='right')
x = pd.cut(datatweet['Sentiment'].to_list(), bins)
x.categories = ['Negative','Neutral','Positive']
datatweet['Polarity'] = x
# Analysis examples:
# Mean by date
ex1 = datatweet.groupby(['date'])['Sentiment'].describe()[['count',
'mean']]
ex1 = ex1.reset_index()
ex1.head()
ex1
ex1.to_csv("DogecoinDailySentiment.csv", index = False)
df = pd.read_excel(r'C:\Users\joaof\Dropbox\Tese\2. Data
Sources\Coins\DOGE\DOGE-USD_2021.xlsx')
df
# Date - Sentiment graph

# x axis values
x = df['date']
# corresponding y axis values
y = df['ts_polarity']
```

```
# plotting the points
plt.plot(x, y)

# naming the x axis
plt.xlabel('date')
# naming the y axis
plt.ylabel('sentiment average')

# giving a title to my graph
plt.title('tweets sentiment score over time')

# function to show the plot
plt.show()
# Date - Tweet Volume graph

# x axis values
x = df['date']
# corresponding y axis values
y = df['twitter_volume']

# plotting the points
plt.plot(x, y)

# naming the x axis
plt.xlabel('date')
# naming the y axis
plt.ylabel('tweets volume')

# giving a title to my graph
plt.title('tweets volume over time')

# function to show the plot
plt.show()
# Date - Price graph

# x axis values
x = df['date']
# corresponding y axis values
y = df['Adj Close']

# plotting the points
plt.plot(x, y)

# naming the x axis
plt.xlabel('date')
# naming the y axis
plt.ylabel('Adj Close')

# giving a title to my graph
plt.title('Adj Close Price over time')

# function to show the plot
plt.show()
```

```
# Date - Traded Volume graph

# x axis values
x = df['date']
# corresponding y axis values
y = df['Volume']

# plotting the points
plt.plot(x, y)

# naming the x axis
plt.xlabel('date')
# naming the y axis
plt.ylabel('Volume')

# giving a title to my graph
plt.title('Traded Volume over time')

# function to show the plot
plt.show()
```

7.3. PREDICTION MODELS SCRIPT

```
# Random Forest Regressor Model

#Importing Libraries
import numpy as np
import pandas as pd
from sklearn.ensemble import RandomForestRegressor
%matplotlib inline
from sklearn import metrics

# Dataframe with Adj close, ts_polarity, twitter_volume of APPL
df = df[["Adj Close", "ts_polarity", "twitter_volume"]]
df.head()

# pct change based on Adj close value
df["Pct_change"] = df["Adj Close"].pct_change()
df.head()

Creating the Features X and Target y Data
The first step towards preparing the data was to create the input features
X and the target vector y. We used the window_data() function to create
these vectors.

This function chunks the data up with a rolling window of Xt - window to
predict Xt.

The function returns two numpy arrays:

X: The input features vectors.

y: The target vector.
```

The function has the following parameters:

df: The original DataFrame with the time series data.

window: The window size in days of previous closing prices that will be used for the prediction.

feature_col_number: The column number from the original DataFrame where the features are located.

target_col_number: The column number from the original DataFrame where the target is located.

```
# This function "window_data" accepts the column number for the features
(X) and the target (y)
# It chunks the data up with a rolling window of Xt-n to predict Xt
# It returns a numpy array of X any y
def window_data(df, window, feature_col_number1, feature_col_number2,
feature_col_number3, target_col_number):
    # Create empty lists "X_close", "X_polarity", "X_volume" and y
    X_close = []
    X_polarity = []
    X_volume = []
    y = []
    for i in range(len(df) - window):

        # Get close, ts_polarity, tw_vol, and target in the loop
        close = df.iloc[i:(i + window), feature_col_number1]
        ts_polarity = df.iloc[i:(i + window), feature_col_number2]
        tw_vol = df.iloc[i:(i + window), feature_col_number3]
        target = df.iloc[(i + window), target_col_number]

        # Append values in the lists
        X_close.append(close)
        X_polarity.append(ts_polarity)
        X_volume.append(tw_vol)
        y.append(target)

    return np.hstack((X_close,X_polarity,X_volume)), np.array(y).reshape(-
1, 1)

# Predict Closing Prices using a 3 day window of previous closing prices
window_size = 3

# Column index 0 is the `Adj Close` column
# Column index 1 is the `ts_polarity` column
# Column index 2 is the `twitter_volume` column
feature_col_number1 = 0
feature_col_number2 = 1
feature_col_number3 = 2
target_col_number = 0
X, y = window_data(df, window_size, feature_col_number1,
feature_col_number2, feature_col_number3, target_col_number)
```

```
# Use 70% of the data for training and the remainder for testing
X_split = int(0.7 * len(X))
y_split = int(0.7 * len(y))

X_train = X[: X_split]
X_test = X[X_split:]
y_train = y[: y_split]
y_test = y[y_split:]

Scaling Data with MinMaxScaler

We will use the MinMaxScaler from sklearn to scale all values between 0 and
1. Note that we scale both features and target sets.

from sklearn.preprocessing import MinMaxScaler

# Use the MinMaxScaler to scale data between 0 and 1.
x_train_scaler = MinMaxScaler()
x_test_scaler = MinMaxScaler()
y_train_scaler = MinMaxScaler()
y_test_scaler = MinMaxScaler()

# Fit the scaler for the Training Data
x_train_scaler.fit(X_train)
y_train_scaler.fit(y_train)

# Scale the training data
X_train = x_train_scaler.transform(X_train)
y_train = y_train_scaler.transform(y_train)

# Fit the scaler for the Testing Data
x_test_scaler.fit(X_test)
y_test_scaler.fit(y_test)

# Scale the y_test data
X_test = x_test_scaler.transform(X_test)
y_test = y_test_scaler.transform(y_test)

# Create the Random Forest regressor instance
model = RandomForestRegressor(n_estimators=1000, max_depth=2,
bootstrap=False, min_samples_leaf=1)

# Fit the model
model.fit(X_train, y_train.ravel())

Model Performance
In this section, we will evaluate the model using the test data.

We will:

Evaluate the model using the X_test and y_test data.
Use the X_test data to make predictions
Create a DataFrame of Real (y_test) vs predicted values.
Plot the Real vs predicted values as a line chart
```

```
# Make some predictions
predicted = model.predict(X_test)

# Evaluating the model
print('Root Mean Squared Error:',
np.sqrt(metrics.mean_squared_error(y_test, predicted)))
print('R-squared :', metrics.r2_score(y_test, predicted))

# Recover the original prices instead of the scaled version
predicted_prices = y_test_scaler.inverse_transform(predicted.reshape(-1,
1))
real_prices = y_test_scaler.inverse_transform(y_test.reshape(-1, 1))

# Create a DataFrame of Real and Predicted values
dogecoin = pd.DataFrame({
    "Real": real_prices.ravel(),
    "Predicted": predicted_prices.ravel()
}, index = df.index[-len(real_prices): ])
dogecoin.head()


# Plot the real vs predicted values as a line chart
dogecoin.plot(title = "Real vs Predicted values of Dogecoin")
```

XG Boost Regressor Model

```
#Importing Libraries
import numpy as np
import pandas as pd
from xgboost import XGBRegressor
%matplotlib inline
from sklearn import metrics

# Dataframe with Adj close, ts_polarity, twitter_volume of APPL
df = df[["Adj Close", "ts_polarity", "twitter_volume"]]
df.head()

# pct change based on Adj close value
df["Pct_change"] = df["Adj Close"].pct_change()
df
```

Creating the Features X and Target y Data
The first step towards preparing the data was to create the input features X and the target vector y. We used the window_data() function to create these vectors.

This function chunks the data up with a rolling window of Xt - window to predict Xt.

The function returns two numpy arrays:

X: The input features vectors.

y: The target vector.

The function has the following parameters:

df: The original DataFrame with the time series data.

window: The window size in days of previous closing prices that will be used for the prediction.

feature_col_number: The column number from the original DataFrame where the features are located.

target_col_number: The column number from the original DataFrame where the target is located.

```
# This function "window_data" accepts the column number for the features
(X) and the target (y)
# It chunks the data up with a rolling window of Xt-n to predict Xt
# It returns a numpy array of X any y
def window_data(df, window, feature_col_number1, feature_col_number2,
feature_col_number3, target_col_number):
    # Create empty lists "X_close", "X_polarity", "X_volume" and y
    X_close = []
    X_polarity = []
    X_volume = []
    y = []
    for i in range(len(df) - window):

        # Get close, ts_polarity, tw_vol, and target in the loop
        close = df.iloc[i:(i + window), feature_col_number1]
        ts_polarity = df.iloc[i:(i + window), feature_col_number2]
        tw_vol = df.iloc[i:(i + window), feature_col_number3]
        target = df.iloc[(i + window), target_col_number]

        # Append values in the lists
        X_close.append(close)
        X_polarity.append(ts_polarity)
        X_volume.append(tw_vol)
        y.append(target)

    return np.hstack((X_close,X_polarity,X_volume)), np.array(y).reshape(-
1, 1)

# Predict Closing Prices using a 3 day window of previous closing prices
window_size = 3

# Column index 0 is the `Adj Close` column
# Column index 1 is the `ts_polarity` column
# Column index 2 is the `twitter_volume` column
feature_col_number1 = 0
feature_col_number2 = 1
feature_col_number3 = 2
target_col_number = 0
X, y = window_data(df, window_size, feature_col_number1,
feature_col_number2, feature_col_number3, target_col_number)
```

```
# Use 70% of the data for training and 30% for testing
X_split = int(0.7 * len(X))
y_split = int(0.7 * len(y))

# Set X_train, X_test, y_train, t_test
X_train = X[: X_split]
X_test = X[X_split:]
y_train = y[: y_split]
y_test = y[y_split:]
```

Scaling Data with MinMaxScaler
We will use the MinMaxScaler from sklearn to scale all values between 0 and 1. Note that we scale both features and target sets.

```
from sklearn.preprocessing import MinMaxScaler

# Use the MinMaxScaler to scale data between 0 and 1.
x_train_scaler = MinMaxScaler()
x_test_scaler = MinMaxScaler()
y_train_scaler = MinMaxScaler()
y_test_scaler = MinMaxScaler()

# Fit the scaler for the Training Data
x_train_scaler.fit(X_train)
y_train_scaler.fit(y_train)

# Scale the training data
X_train = x_train_scaler.transform(X_train)
y_train = y_train_scaler.transform(y_train)

# Fit the scaler for the Testing Data
x_test_scaler.fit(X_test)
y_test_scaler.fit(y_test)

# Scale the y_test data
X_test = x_test_scaler.transform(X_test)
y_test = y_test_scaler.transform(y_test)

# Create the XG Boost regressor instance
model = XGBRegressor(objective='reg:squarederror', n_estimators=1000)

# Fit the model
model.fit(X_train, y_train.ravel())
```

Model Performance
In this section, we will evaluate the model using the test data.

We will:

Evaluate the model using the X_test and y_test data.
Use the X_test data to make predictions
Create a DataFrame of Real (y_test) vs predicted values.
Plot the Real vs predicted values as a line chart

```
# Make some predictions
predicted = model.predict(X_test)

# Evaluating the model
print('Root Mean Squared Error:',
np.sqrt(metrics.mean_squared_error(y_test, predicted)))
print('R-squared :', metrics.r2_score(y_test, predicted))

# Recover the original prices instead of the scaled version
predicted_prices = y_test_scaler.inverse_transform(predicted.reshape(-1,
1))
real_prices = y_test_scaler.inverse_transform(y_test.reshape(-1, 1))

# Create a DataFrame of Real and Predicted values
stocks = pd.DataFrame({
    "Real": real_prices.ravel(),
    "Predicted": predicted_prices.ravel()
}, index = df.index[-len(real_prices): ])
stocks.head()

# Plot the real vs predicted values as a line chart
stocks.plot(title = "Real vs Predicted values of Dogecoin")
```