Non-Fungible Token (NFT) Authenticity using Sentiment Analysis

Oleksandr Taradachuk
Macaulay Honors — Hunter College
oleksandr.taradachuk@macaulay.cuny.edu

Anthony Sokolov
Daedalus Honors — Hunter College
anthony.sokolov13@myhunter.cuny.edu

Abstract

The Non-Fungible Token (NFT) market is ripe with scam projects that utilize Twitter as a marketing tool to entrap consumers and walk away with the money they invest. We utilize sentiment analysis of Twitter discussions about projects in conjunction with transaction data from OpenSea, the main secondary market for Ethereum-based NFTs, as inputs to various classification models to tackle this problem. While our models achieve great accuracy and provide promising preliminary results, the size of our dataset makes it difficult to conclude anything definitive and requires further research with more data.

1 Introduction

Non-Fungible Tokens (NFTs) are a recent blockchain development that allow for the verification of digital media. The market for these tradable digital collectibles has exploded in 2021, surpassing over $12 billion within months [5]. However, the decentralized nature of blockchain allows anybody to create an NFT project, market it to get investors, and pull away with all the profits with little-to-no accountability. These scams, often referred to as *"rug pulls"*, are a major problem in the space and result in new investors losing money as well as money flowing towards scammers when it could've went to contribute to the space. Twitter is the main public forum for NFTs and scam projects rely on it as a marketing to tool to trap consumers by creating false engagement and inflating their follower count.

The space continues to grow at a rapid pace which consequently causes a rise in scam projects. This isn't specific to just the NFT space, but the broader decentralized finance sector as well where consumers lost approximately $2.8 billion to *"rug pull"* scams in 2021 [4]. It is clear from these numbers that research is necessary into ways to help mitigate the public from falling for these cash grabs. Having an authenticity metric that can be used to better gauge whether a project is genuine or not would help investors make smarter, better informed decisions about where they invest.

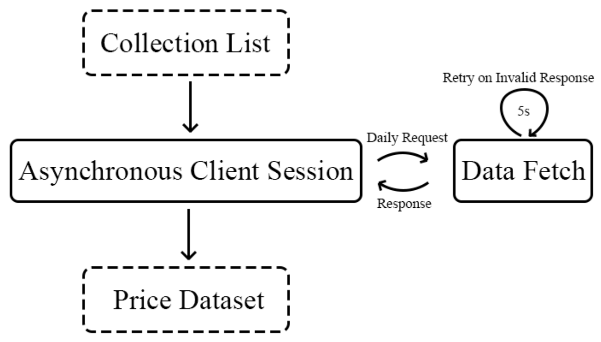
2 Previous Research

There has not been any direct research into classifying non-genuine NFT projects given the novelty of the space, however, it is a similar problem to predicting stock price movement using Twitter. Most of the literature on the topic cites *Bollen, Mao, Zeng* [2] as being the first foundational publication to demonstrate a strong correlation between the collective mood state of users on Twitter and the Dow Jones Industrial Index (DIJA). Following this, many papers expanded the use of sentiment analysis on stock price prediction to the cryptocurrency market. The paper *Price Movement Prediction of Cryptocurrencies Using Sentiment Analysis and Machine Learning* [6] outlines the results of using Multilayer Perceptrons (MLPs), Support Vector Machines (SVMs), and sentiment analysis in predicting the price of digital assets. The paper focuses on the top cryptocurrencies by market capitalization and makes use of market data in addition to social data. The initial dataset of tweets was processed using a *Valence Aware Dictionary and Sentiment Reasoner* (VADER)[3] in order to generate a feature vector to be used as input for MLP, SVM, and random forest models. The paper *Cryptocurrency Price Prediction Using Tweet Volumes and Sentiment Analysis*[1] also attempts to make use of the VADER model to process raw Twitter data but finds them ineffective for making predictions. Our project will further explore if sentiment analysis can be useful for making predictions and detecting the genuineness of an NFT project. We are not simply trying to predict price but also trying to identify founding teams with dishonest intentions; for this reason, sentiment analysis can still be useful.

3 Method

3.1 Data Collection

We collected data related to 40 NFT projects on the Ethereum blockchain. For each project, we collected both transaction data as well as tweet data related to the discussion of that project. Each project has an associated boolean label of genuine or non-genuine depending on whether the project was known to be a scam. Given the scarcity of data collected on NFT projects due to their novelty, a majority of our work revolved around gathering appropriate data that we could analyze. OpenSea, the main secondary market for Ethereum-based NFTs, provided us with access to their API so that we could scrape transaction data pertaining to specific projects. However, due to the heavy throttling of the API, we had to set up asynchronous tasks that would only complete if a valid response was returned. We decided that a two-week period after a project went public was sufficient enough of a time-frame to provide valuable information about its financial performance. To achieve this, we scraped daily transactions within a two-week period for a particular project due to the response limits of the API. Overall, the transaction scraping structure is depicted below:



For the tweets, we used the Twitter API to collect data on the set of NFT projects we've identified as part of our data. We used the Tweepy Python library to act as a wrapper around the API requests. We used the All Time Search endpoint from Twitter, which returns tweets from any time-frame related to a query. We were using a free license, which meant that we were limited to 100 Tweets returned per request and only 50 requests total. Across 40 projects we collected upwards of 2000 Tweets. While we were able to collect up to 100 Tweets per project, some queries returned less and some even returned no tweets. The only projects returning no tweet data were projects that we identified as scams. We generally found less data on the scam related projects; we credit this to the fact that many scams are less known than the other established projects included in our data set. Imperfect queries are another reason for lack of data, "FPS Loot NFT" may return different results than "FPS Loot"; due to limitations on the number of requests available we were unable to query for the different variations of a project name.

Another issue with our data collection is the fact that it in many cases the Twitter API returned many duplicate and spam tweets. Two options to mitigate the effect of this on the learning of our models would be to collect more data and to filter out retweets when querying the Twitter API. Both of these options were only available under the premium version of the Twitter API.

One option we explored to better collect data was to use the Streaming API offered by Twitter. This allows for use of the API to monitor Twitter for Tweets related to a query in real time. We did not have success in using this method because it requires that an instance of the Streaming API be running for the entire collection period. This approach also does not let us collect historical data.

In our collection, we also logged the timestamps of tweets be able to track the Tweet volumes of different projects.

3.2 Feature Extraction

From the data collected, we extracted the following features to be used for analysis:

3.2.1 Transaction Count

We scraped 55,831 transactions across the 40 projects that we compiled. Using the transaction data, we tracked the total number of transactions that were received for a particular project. This was often inconsistent as many projects were not entirely bought up weeks after they were first made public. For this reason, we chose our window of two-weeks as we found that a one-week window often resulted in minimal transactions being recorded.

3.2.2 Total Volume

With every transaction we parsed, we tracked how much total Ethereum (ETH) was being traded for a particular project to create a total volume feature to complement our transaction count.

3.2.3 Average Price

Our final feature for price analysis was an average trading price for a project over a two-week period. This was done by taking our total volume feature and dividing it by our transaction count.

3.2.4 Sentiment Analysis

We used a Valence Aware Dictionary and Sentiment Reasoner (VADER) model to perform sentiment analysis on our Tweet dataset. We used the implementation found in the vaderSentiment Python library for our analysis. The VADER model computes a negative, positive, and neutral sentiment score for a series of text. In our project, we used the VADER model to compute sentiment for every tweet in our dataset related to a particular project. The average of all the sentiment vectors was then taken in order to produce a single 1x3 vector of sentiment scores for a project. This sentiment vector is what would act as the input to the machine learning classification models. Each of the three sentiment scores were broken up to be their own feature.

3.2.5 Tweet Volume

We tracked an approximation of Tweet volume as an additional metric to provide to our machine learning classification models. We computed this metric by taking the standard deviation from the average timestamp for all the Tweets related to a project; we converted all timestamps to the number of seconds elapsed since the start of Unix time, then took the average and standard deviation. This is a very imperfect metric because it is entirely reliant upon the data provided by the Twitter API; unfortunately Twitter does not provide much data about how it returns the 100 Tweets that it does.

3.3 Classification Models

We used three different models to train on the data that we gathered: a Gaussian Naive Bayes, a Support Vector Machine (SVM), and a Multilayer Perceptron (MLP). The data was split 80/20 for training and testing and each split set was standardized to have 0 mean and unit variance.

3.3.1 Gaussian Naive Bayes

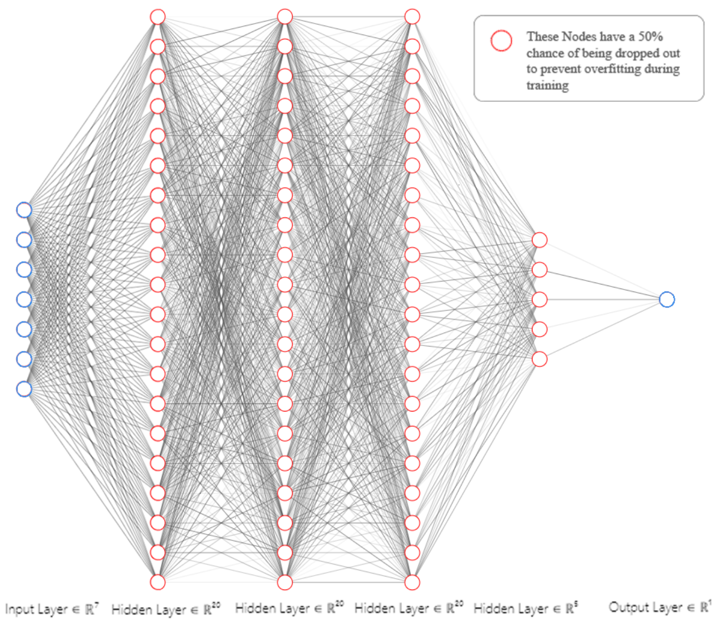
Since we were building a classifier, a Naive Bayes approach was an obvious first choice for us. However, since most of our data is numerically continuous and not categorical, it is not ideally suited for the discrete nature of Naive Bayes. In our code, we utilized the Gaussian Naive Bayes classifier from *scikit-learn* to carry out our analysis.

3.3.2 Support Vector Machine

A Support Vector Machine (SVM) is a machine learning classifier that finds the best fitting hyperplane in an N-dimensional field to maximize the distance between points of two data classes. Given that our problem is binary classification, we chose this classifier as our second approach as it works best with continuous numerical data due to its reliance on Euclidean distance. Similarly to Naive Bayes, we utilized an SVM classifier from *scikit-learn* to carry out our analysis.

3.3.3 Multilayer Perceptron

Our final approach was to employ deep learning using a fully-connected Multilayer Perceptron. The model was constructed using *Tensorflow* to have 4 hidden layers with a 0.5 dropout so that 50% of nodes would be ignored during training to help prevent overtraining and co-dependency. A full representation of our model can be seen below:



4 Results/Evaluation

The accuracy of our models on different data combinations is shown in the following table:

Dataset	SVM	Naive Bayes	MLP
Tweet Only	0.75	1.0	0.75
Price Only	1.0	0.625	1.0
Tweet and Price	0.875	0.75	0.875

Below is a table displaying the precision scores of our models on different data combinations:

Dataset	SVM	Naive Bayes	MLP
Tweet Only	0.75	1.0	1.0
Price Only	1.0	0.0	1.0
Tweet and Price	0.67	0.0	0.5

Below is a table displaying the F1 scores of our models on different data combinations:

Dataset	SVM	Naive Bayes	MLP
Tweet Only	0.75	1.0	0.86
Price Only	1.0	0.0	1.0
Tweet and Price	0.8	0.0	0.67

Our model performed with high accuracy on the test data. However, the small size of our dataset and the low availability of data related to scam projects leads us to believe that our models are over fitting. This is likely due to not enough data. If further research is done using the same data that we compiled, we recommend cross-validation to overcome the issues of train/test variability in small datasets.

The Tweet Only dataset has the highest average recall and F1 Score across all three learning models. This means that the models identified few false positives and few false negatives.

There is reason to be concerned with bias in the transaction data given the performance of the SVM on Price-Only data. A two-week window might've encapsulated projects that revealed themselves as scams within that period which led to the market to react appropriately. However, this is hard to regulate for as projects have been discovered as scams in various timeframes following their public debuts.

The SVM and the MLP achieve the same accuracy on the same data. In this scenario the SVM would be the preferable model as it has less complex of a model than the MLP. The SVM has greater interpretability which is very important with machine learning. The precision and F1 scores between the SVM and MLP models are fairly similar.

The Naive Bayes model has the worst precision score performance. For the Price Only an Tweet/Price datasets the precision score is zero; this means that for all of the projects predicted by the model to be non-genuine, none of them were in fact known to be non-genuine.

Below is a table illustrating the means and standard deviations of the sentiment scores for all of the projects in our training set.

Metric	Mean	Std. Dev
Negative Sentiment	0.0359	0.040
Neutral Sentiment	0.7596	0.210
Positive Sentiment	0.0960	0.059

Our mean values are quite low for both positive and negative sentiment scores. The standard deviations indicate that most of the positive and negative sentiments are not far from the mean. This means that most of the tweets we collected did not have a clear positive or negative sentiment detected, though there was a strong neutral sentiment across projects.

The low polarity among positive and negative sentiment scores is shown by the low max values in the table below. The table shows the maximum sentiment scores found across all projects in the training set.

Metric	Max Value in Set
Negative Sentiment	0.153
Neutral Sentiment	0.920
Positive Sentiment	0.238

Upon an inspection of the tweets in our dataset it is apparent that many of the collected tweets do contain a neutral sentiment. However, we also attribute the low positive and negative sentiment scores to the fact that discourse on Twitter includes a lot of novel slang and special characters. This may have confused the VADER model. If we were to continue working on this project, we would also devote more time to preprocessing of the text data.

5 Conclusion

Our models displayed good accuracy on our testing data; however, the poor quality of our data prevents us from drawing any general conclusions about the performance of our model on completely novel data. Despite the low polarity of negative and positive sentiment in the Tweets we've collected, Tweets do show promise at being able to predict whether or not a project is genuine.

Our work does lead us to believe that a Naive Bayes model would not be an effective model to use for this task, as shown by the low precision and recall metrics. On the other hand, SVMs and deep learning models such as MLPs seem to be suited for this task ,given the data is continuous and numeric, and should be explored further using additional data.

6 Future Work

A big limiting factor for us throughout the course of our research was data. Twitter allowed us to scrape only a limited number of projects and finding projects to use as scams was difficult as they get buried in forums by the more successful ones. Additionally, since our models all employed supervised learning, we had to manually label all projects. If the dataset were to be expanded, we could get a more comprehensive view of how viable it is to classify non-genuine projects using machine learning.

For future work, we would recommend compiling a larger dataset of Tweets. This would require the feature of the Twitter Premium API. Our project would be well served with a larger set of Tweets to learn from. As mentioned in the data section, we experienced a lot of duplicate Tweets and spam Tweets. The ability to collect more data would reduce the proportion of these duplicate tweets and present a better sample of the conversation surrounding an NFT project. The ability to filter out Retweets would also contribute to a better data set.

Another way to improve the quality of the data would be to place a greater focus on the timestamps of the collected Tweets. In our data collection method, we simply queried the Twitter API with the name of the project we were interested in. In the future, we would like to keep track of the exact dates when scam projects were determined to be non-genuine. We would also like to focus our collection of data for genuine projects on the earlier Tweets associated with that project, we want to capture data from the period when the future of a project is still uncertain. Our project aims to classify the genuineness of NFT projects, collecting Tweets about a scam project after it has already been revealed to be non-genuine would present our models with information that a potential investor would not be privy to. Accomplishing this would involve manual labelling of key dates for the projects in our dataset as well as expanded access to the Twitter API; due to our time constraints we were not able to accomplish this.

As was mentioned we encountered a number of duplicate and highly similar Tweets during our data collection process. While the ability to filter these results out may improve performance, we would also suggest engineering a similarity metric to represent the amount of duplicate Tweets. While the duplicate Tweets may just be a result of Twitter's API behaving the way it does, a high degree of similar Tweets can potentially be telling about the genuineness of a project. We propose creating a new data feature that tracks the number of duplicate Tweets related to a project, this metric should account for exact duplicates as well as Tweets that differ only be a few words or characters. A measure of entropy among Tweets would also be a good choice.

Another implementation that may improve performance would be to include additional preprocessing of the Tweet data. Tweets contain a lot of slang, new language, and special characters. Had we had more time available to work on this project, we would explore approaches to remove such special characters and handle symbols such as hashtags.

Additionally, due to the growth of NFTs and the expanding volume we've seen in projects in 2021, the window for transaction collection would have to be narrowed to avoid bias in the data as a result of scammers pulling away from projects within the two-week period used in this research.

All in all, we believe this is still a ripe area for research that can be used to protect investors and foster future growth in the community.

References

[1] J. Abraham, D. Higdon, J. Nelson, and J. Ibarra. Cryptocurrency price prediction using tweet volumes and sentiment analysis. 2018.

[2] J. Bollen, H. Mao, and X.-J. Zeng. Twitter mood predicts the stock market. 2010.

[3] C. Hutto and E. Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. 2014.

[4] S. Malwa. Defi 'rug pull' scams pulled in \$2.8b this year: Chainalysis, 2021.

[5] M. Nystrom. A short, non-fungible story, 2021.

[6] F. Valencia, A. Gomez-Espinosa, and B. Valdes-Aguirre. Price movement prediction of cryptocurrencies using sentiment analysis and machine learning. 2019.

7 Contributions

7.1 Alex

- OpenSea scraping and processing
- Compilation of list of projects
- Data Preprocessing
- Classification Models
- Paper: Introduction, Previous Research, Method, Future Work

7.2 Anthony

- Tweet collection and processing
- Feature engineering of Tweet data
- Perfomance metrics collection
- Comparison of performance across different datasets
- Paper: Method, Results, Conclusion, Future Work