# PPOL 564 Final Report

Kelly Looman

Word Count: 2954

## 1 Introduction

In late February of 2021, NFT artist Beeple sold his latest piece titled "Everdays: the First 5000 Days" for 69 million dollars. Many are confounded that a physical piece of art can sell for millions of dollars, and the befuddlement deepens when people learn that the piece was not even physical artwork but the rights to digital art. NFTs or non-fungible tokens are blocks of data denoting ownership over another digital file. In other words, NFTs are digital contracts signifying ownership over digital media. A NFT, however, does not restrict others from sharing or copying that piece of media. Though many find the idea of NFTs incredulous, as of October 2021, NFT sales reached 10 billion USD in total volume.

Given the novelty of NFTs, there is very little research, studying its trends, communities, or creators, although there is plenty of data available for analysis. The proceeding sections will give a brief overview of the NFT space, outline a potential issue in terms of understanding NFT transactions, and provide a roadmap of how the data was acquired, cleaned, transformed, and modeled. Finally, the results and implications are explored with a discussion regarding how the model could be improved and future research avenues.

## 2 Problem Statement and Background

The original motivation of this project was to obtain a better understanding of blockchain technology with a particular focus on the NFT space. Blockchain is a decentralized and secure digital ledger of all transactions across a peer-to-peer network, and blockchain technology presents a wonderful conundrum. All blockchain transactions are recorded on a *public* ledger, so the data is accordingly accessible to everyone; simultaneously, however, the data is not as accessible as it seems. Its anonymous nature and the sheer amount of data one needs to sift through prevents interested parties from easily understanding what is happening underneath the digital "hood."

Specifically, in the NFT space, one of the most pertinent but unanswered questions is what are the determinants of a successful NFT project? Thus, in this project, I seek to develop a model trained and tested on NFT transactions in Etherscan in order to predict the future success of a NFT project.

NFTs are an emerging technology, so there is little established literature on the subject. The SEC, for example, is still in the midst of determining whether NFTs should be classified as securities. In terms of the understanding why certain NFTs may be successful as opposed to other NFT projects, NFT community members claim the following:

- Has the NFT creator cultivated a strong community and following?
- Is there a scarcity of tokens available, creating a sense of rarity and exclusivity?
- Does the NFT grant you access to other benefits or provide utility? This includes the opportunity for NFT "airdrops", tickets to in-person conferences, or access to a community forum (e.g., Discord).
- Do the tokens offer an interactive experience? This could include individualized "missions" based on the token type (e.g., Koala Intelligence Agency) or the tokens can serve as items or characters in

a video game (e.g., ZED Run).

These qualities – community, scarcity, access and utility, and "interactiveness" – have not been shown to empirically lead to successful NFT projects. However, a quick glance at some of the top NFT projects shows that many of these qualities are clearly present. A Bored Apes Yacht Club token, for instance, will grant the owner access to member-only benefits including additional NFTs, a collaborative graffiti board, and exclusive merchandise. Loot (for Adventurers) tokens represent items that can then be used for RPGs, while other NFT project owners are using the money collected to fund their own video game ideas.

# 3 Data

My data is acquired from Etherscan, a block explorer and analytics platform. Etherscan records all transactions on the Ethereum blockchain, so NFT transfers can be viewed in real-time on the site. The data was gathered using Etherscan's API, which offers a limited but free tier subscription. A NFT transfer is recorded in three parts or three API calls: i.) Normal, ii.) Internal, and iii.) NFT Transfer. The diagram below provides a workflow of a NFT transaction.

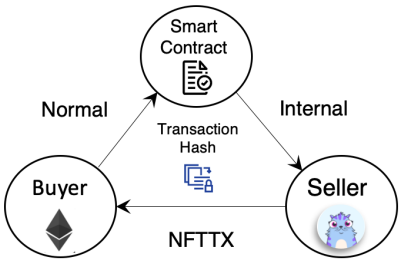



Figure 1: Etherscan API Call

As can be seen, each of these calls offers a snapshot of a transaction. Normal is recording the interaction between a buyer and a smart contract [1]; internal is recording the interaction between a smart contract and a seller; and NFTTX or NFT Transfer is recording the final transfer of the token from the buyer to the seller.

In order to obtain this transaction data, however, I first needed to obtain a list of users who are buying or selling a NFT token. Every NFT project has its own smart contract, which is connected to a unique contract address. Using Etherscan's API, I pulled a list of Normal transactions and subset the data to only include unique user addresses. These addresses were then fed into the three API calls referenced above. It should be noted that Etherscan limits free tier users to five API calls per second. This is actually much faster than scraping the website using Selenium or Beautiful Soup, but it required me to continually pause between each API call in order to prevent errors.

After almost one-hundred hours of run-time and the deaths of many kernels, I finally obtained the raw data set, which was a little under 100 GB of data. For a reference, one GB is equivalent to a five-hour movie, so 100GB is approximately 500 hours of television.

As indicated by the diagram below, at this point, the dataset included all transactions involving our chosen NFT projects, as well as any interactions between owners of the relevant NFT projects and other cryptos or projects not of interest.

[1]Smart contracts are lines of code or functions stored on the blockchain. When the functions are called or initiated, the code will run, automating the execution of an agreement. Smart contracts serve as an intermediary between buyers and sellers to ensure a transaction takes place as it should, but it does not require the fees that an institution or individual might charge.

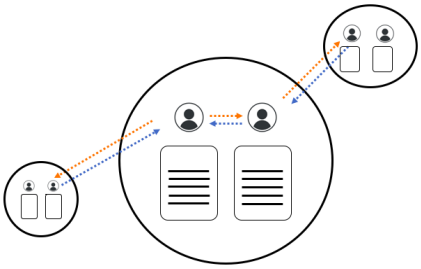


Figure 2:

Using Dask, a Python parallel computing package, the resulting data sets were then merged together based on transaction hash. As the dataset included all transactions involving the original buyer, the data was then subset to only include transactions involving the NFT projects of interest.

In terms of features, the transaction data included many variables such as Buyer, Seller, Contract, Transaction Amount, Timestamp, Gas Price, Gas Limit, and Gas Amount. Despite the large amount of data, there were not enough features or variation within this dataset to run a classification model. Thus, I needed to engineer several relevant features (including the dependent variable).

I originally chose 100 NFT projects – fifty successful projects and fifty unsuccessful projects. Projects were defined as a smart contract with over 100 tokens and with a cohesive purpose. This was to avoid single, high value NFTs (e.g., Beeple's First 5000 Days) or an eclectic variety of NFTs that did not have an underlying theme tying the tokens together. Open Sea – a smart contract platform – has a dashboard listing NFT projects with the highest trade volume, which is how the fifty successful projects were chosen. Success was determined by having a total trade volume of above 80 ether (Ethereum crypto or currency). Many projects would typically have 10,000 tokens, so an 80-ether trade volume would translate to each token on average being bought for 0.008 ether or $32.

I then hand-coded the following variables drawing from Open Sea, the NFT project sites, and Etherscan:

- Success: a dummy variable denoting if a project had a total trade volume of above 80 ether.
- Access and Utility: a dummy variable denoting if the token provided access to membership-only perks.
- Interactive: a dummy variable denoting if the token had interactive features or granted the owner access to an interactive game.
- Art: a dummy variable denoting if the token is purely art with no other utility.
- Community: a dummy variable denoting if the project has an exclusive Discord channel.
- Number of Tokens Minted

These elements were chosen based on the criteria discussed previously.

From the original data set, I calculated the following features for three time periods (i.e., first day of the project release, two-weeks from the start date, and one month from the start date):

i.) Number of tokens bought on the first day of the project release
ii.) Total value of tokens bought on the first day of the project release.
iii.) Average value of tokens bought on the first day of the project release.

The main concern with this dataset is the sample size. I would have liked a significantly larger pool of projects to work with, but it took almost a week of run-time to gather, process, and obtain simple averages and sums of only 100 projects even when using packages such as Dask. In hindsight, one way to

circumvent this issue is to obtain analytics data from OpenSea. OpenSea is a smart contract platform and a marketplace to buy and sell NFTs. They are one of many such marketplaces though they are by far one of the most popular platforms (e.g., Axie, Rarible, Nifty, Larva Labs, etc.). OpenSea draws data from Etherscan (as all Ethereum smart contract platforms do), and they run a series of analytics on each project.

Since the site is protected by Cloudflare, a website security company, normal "bots" are blocked from scraping the site. Rather than using BeautifulSoup, one can use Selenium to simulate a user-like behavior with necessary *__cf_bm* and *__cf_credential* cookies and scrape data from the network log of the site's internal API calls. While data collection would be more time-consuming, it would significantly reduce the amount of data, making it much easier to work with. However, this would only grant us data on transactions that occurred through OpenSea not through any other platforms.

# 4 Analysis

Once my features (e.g., averages, total transaction volumes, etc.) were created, I could finally begin analyzing the data. In Figure 3 below, you can see the distributions and count values of the potential input features.

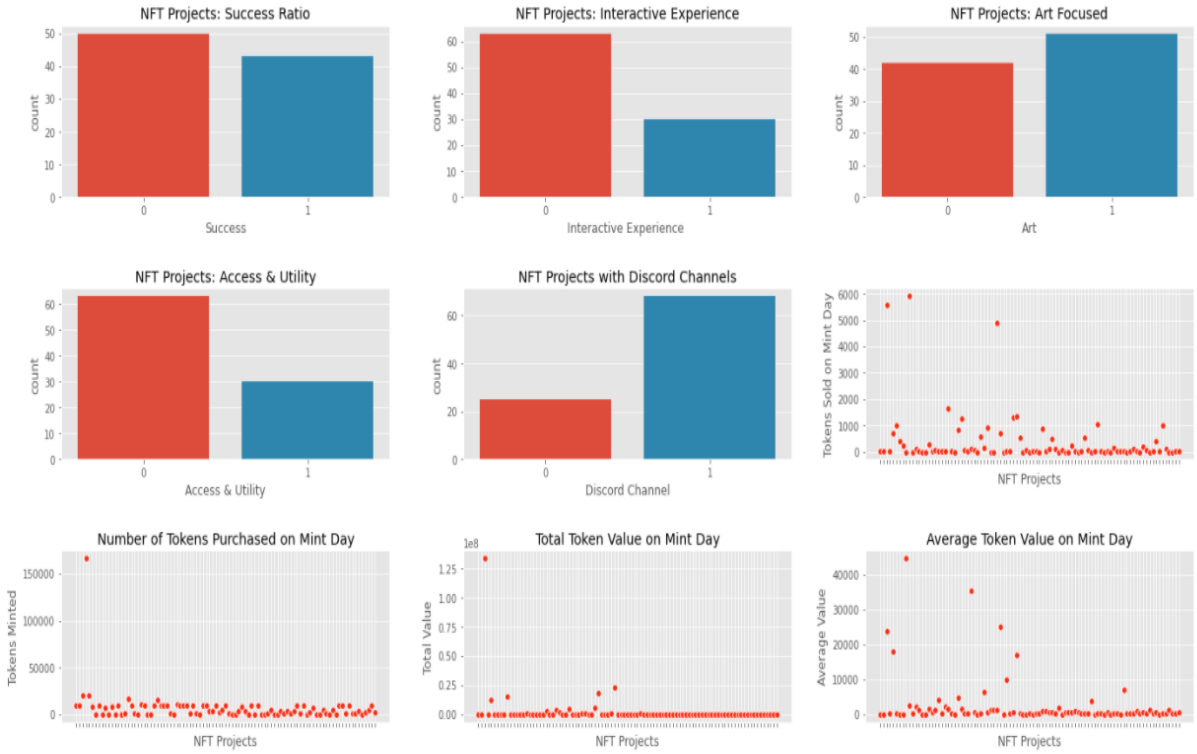



Figure 3: Descriptive Statistics of Independent Variables

After dropping a few NFT projects due to smart contract incompatibility, I was left with 93 projects. My dependent variable (i.e., Success) and potential input features (i.e., Number of Tokens Minted, Quantity Adjusted at Mint Day, Access  Utility, etc.) were assigned to objects. A few of my input features were logged using numpy in order to account for the high variance of values. Some projects sold over 5000 tokens the first day of mint (e.g., CryptoKitties), as they have a relatively small minting price, while other (including successful projects) only sold one or two tokens on their release date. There were no other significant skews, so no other data transformations were conducted on the features. Using the sklearn package the data was then split into a training and testing set (.75: .25 ratio); this allows us to train our model and then assess

its performance on never-before-seen observations.

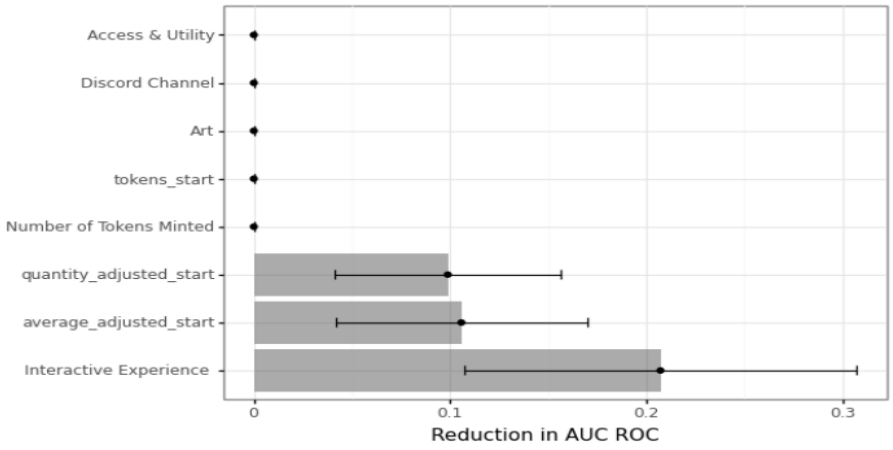
The data was then run through a machine learning pipeline, which helps us automate the machine learning workflow. The pipeline includes K-fold cross validation, pre-processing (i.e., MinMaxScaler), and modeling. Cross validation is a resampling method that splits the observations into a specified number of groups, from which it then delineates between a training and testing set. The training set is "crossed" or analyzed, and predictions are generated and compared to the testing set. Ideally, this will help prevent overfitting. The MinMaxScaler rescales our features to fit into a range of zero or one. This standardizes our data but the cost of bounding the range is smaller standard errors that can diminish the effect of outliers.

I specified three models: Naïve Bayes, K-Nearest Neighbors (KNN), and Decision Tree. To explain these models briefly, a Naïve Bayes will estimate the likelihood of success of a project based on the input values of our independent variables. A KNN classifier will determine how a new observation should be classified based on a predetermined number of "near" or similar observations. A decision tree classifies points by splitting observations based on features (e.g., Number of Tokens Sold ¿ 15), and then our test observations are coded based on which category they fall into.

# 5 Result

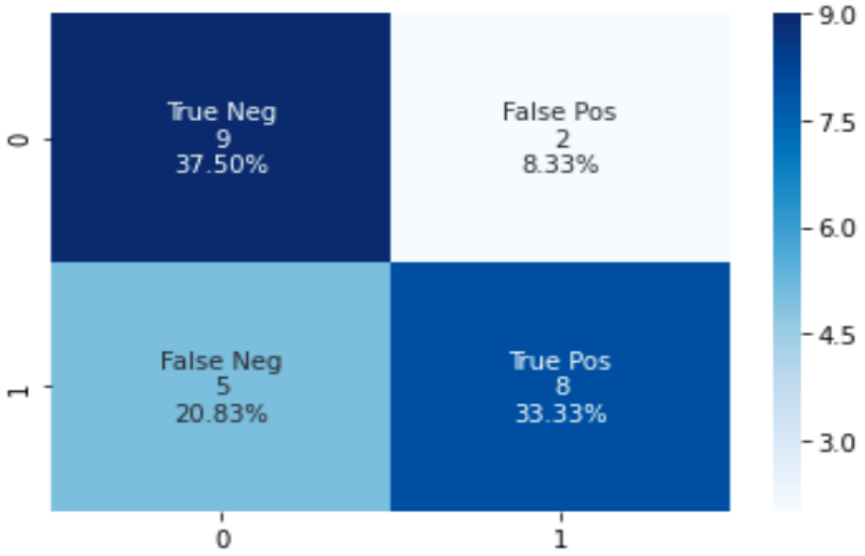
The best performing model was a decision tree classifier with a max depth of two. Our accuracy score was 0.708, meaning our model accurately predicted the success of a project 71 percent of the time. This is not ideal, but it is better than a coin flip. I then assessed whether our model was overfitted by comparing the accuracy scores of our test and training data set. The accuracy score of our training set was 0.826 indicating a high chance of overfitting. Often overfitting is due to too many input features, so I assessed which features were most important for classifying the test observations. As demonstrated in Figure 4 below, *quantity_adjusted_start*, *average_adjusted_start*, and *interactive experience* were the features of greatest significance. Hence, it appears that projects that spent the time to connect their tokens to an experience are most successful. I reran the models with just these three features and surprisingly received very similar results (e.g., a one one-thousandth difference). Thus, the potential cause of overfitting may be due to the small sample size or the features themselves do not best characterize our dependent variable.

Figure 4: Feature Importance




In terms of understanding the implications of our model, a confusion matrix is useful for visualizing the results. Our test dataset only had twenty-four projects, but according to the Figure 5 below, we predicted seventeen projects correctly and seven projects incorrectly.

Figure 5: Confusion Matrix




If this model was used for investment purposes, a main priority would be to reduce the number of false positives (a false positive indicates that we predicted the project would be successful, but it was not). If we were to invest $1000 in ten different projects, two of the projects would have failed, and we would have likely completely lost our $1000 invested in the two projects or $2000 in total. We would accordingly need at least a 25 percent return on investment in order to warrant using this model. This would require that our initial investment in the eight successful projects results in a return of $10,000. For example, an individual would need to buy a token for around 0.26 ETH and then sell for 0.33 ETH (at current ETH exchange rate). The model could certainly use improvement whether through hyperparameter tuning or increasing the number of projects we are observing.

# 6 Discussion

At the beginning of the project, the goal was two-fold. First, I needed to create a reproducible process to collect data from Etherscan, and second, I sought to create a time series model to help investors gauge a good price for a quality asset. In terms of creating a reproducible product, I was originally hoping to create a web-scraper to scrape the data from the Etherscan site. This would have allowed future users to gather already cleaned and processed data, and they would not be reliant on Etherscan to keep and maintain its API. Simultaneously, however, there is something valuable about working with raw data, as you not only familiarize yourself with the projects, but it offers the opportunity to work on different projects in the future. Regarding the time series analysis, I did complete a number of time series model (e.g., ARIMA, exponential smoothing, and LSTM) that focused on a single project (i.e., Vee Friends). These models performed quite poorly however, most likely due to the project's relatively short timeline. Moreover, it did not address one of the main questions in the NFT community: why do certain projects succeed over others? I had considered a classification analysis in the brainstorming stage, but I found the task daunting. However, my experience with the time series gave me confidence that I could obtain and work with the dataset. Thus, I do believe I met my original goal, though the criteria for success in my project was later redefined.

Regarding future analysis, there is a plethora to be done. For example, I only examined 100 projects, but there are tens of thousands of projects that could be examined. As mentioned previously, NFTs and blockchain technology in general is a very new field. One specific area of research that I am interested in is money laundering detection. Given the high prices some NFTs are selling for, many believe NFTs

are being used as way to clean dirty money. Traditional financial institutions have information regarding account owners, but several studies predict that banks only detect about 90 percent of money laundering cases. The anonymous nature of the blockchain makes money laundering detection even more difficult, so detection would have to be determined purely through transaction patterns. One particular idea is to detect routine or repetitive transactions between two user addresses. As I truly believe NFTs and the underlying smart contract technology are here to stay, understanding ways to prevent fraud is important for the future stability of our financial system and integrity of the NFT community.

# 7 Works Cited

Ethereum.(2021)."Non-fungible tokens", https://ethereum.org/en/nft/.

Etherscan - https://etherscan.io/. Accessed November 14, 2021.

Evans, Brian D.(2021)."How To Make An NFT Project As Popular As Nike," *Forbes,* https://www.forbes.com/sites/forbesagencycouncil/2021/10/19/how-to-make-an-nft-project-as-popular-as-nike.