Using Twitter Data in Non-Fungible Token Investing

Oswaldo Olivo
UC Berkeley School of Information
oswaldo@ischool.berkeley.edu

Abstract

Non-fungible tokens (NFTs) are digital collectibles stored in major blockchains such as Bitcoin, Ethereum, and Solana. Some NFTs are considered works of digital art, provide yield to holders, access to games or social communities, or serve as a simply as a speculative investment. They have become widely popular since February 2021, and are a novel asset class similar to cryptocurrencies, becoming heavily traded in exchanges such as OpenSea and FTX. Determining whether future sales of NFTs will be profitable is both beneficial for artists trying to make sure their project is on the right track as well as collectors trying to choose uncover valuable collectibles. In this project we explore the use of Twitter data to predict whether the next sale of an NFT will sell at a higher, lower, or similar price to the previous sale. We found that using Twitter sentiment provides a 10% increase in terms of classifying sale price movement with respect to the baseline model, suggesting the fact that including features from Twitter data is a promising component of an NFT trading algorithm.

1 Introduction

Non-fungible tokens (NFTs) are digital collectibles stored in major blockchains such as Bitcoin, Ethereum, and Solana. While initially NFTs were regarded as digital art, projects have evolved to offer other benefits to their holders, such as a dividend, access to a game or social community, fractional rights of revenue distribution from apps, among other utilities.

Since February 2021, where CryptoPunks, which were initially airdropped for free, started selling in the order of millions of dollars, NFTs have become a more popular alternative asset class, similar to cryptocurrencies. Having a model for predicting whether NFT sale prices for a project will go up, down, or stay flat will be both useful for artists trying to determine if their creation will be successful in the near term, and for collectors to discover valuable projects before they become prohibitively expensive.

In this project we extract different features from Twitter data and use them to predict whether the next sale price of an NFT will be higher, lower, or relatively the same as the previous sale price. That is, for a given sale event, we formulate a classification problem with the labels UP, DOWN, FLAT, corresponding to the next sale price being higher, lower, and within 5% of the previous sale price.

We collect tweets for different time windows within the sale events, and compute DistilBERT embeddings of the tweets, and compute cosine similarity with tweets for the different classification labels, intuitively determining whether a group of tweets between sale events is closer to tweets corresponding to the group of increasing, decreasing or flat sales for other events.

We train an LDA model to determine the probability of the presence of a top-10 topics of the tweets between sale events, and use these presence probabilities as features.

We compute sentiment polarity of the tweets between sale events using Spacy, which yields -1 for extremely negative and 1 for extremely positive. We use the average sentiment and number of positive tweets between sale events as a feature.

We also add the number of tweets between the sale events.

We collected over 500K tweets for the Cool Cats NFT collection, and all the sale events in OpenSea (around 1700), from the inception of the project (July 2021) to October 2021.

We compare the results of the classification task with decision trees, random forests, Naive Bayes, and LSTM and RNNs. We obtained a 10accuracy and AUC with respect to a baseline model of random guessing in a balanced test set. This shows that features derived from Twitter data provide insight into NFT sale price movements, and are an

important component of a forecasting tool.

2 Project Overview

2.1 Datasets Used

- **NFT sales data:** We collected all sale events for 1000 NFTs from the Cool Cats collection, from its creation (July 2021) to October 2021, using the OpenSea event API (https://docs.opensea.io/reference/retrieving-asset-events). There were 2420 sales in total.

- **Twitter data:** We collected all tweets referencing the Cool Cats NFT project between July 2021 and October 2021, using the Twitter search API (https://developer.twitter.com/en/docs/twitter-api/tweets/search/api-reference/get-tweets-search-all), for which we were granted academic access to query beyond the previous seven days of historical data. There were 54,453 tweets in this time interval.

- **Embeddings data:** We used word embeddings from the output layer of DistilBERT(https://huggingface.co/docs/transformers/model_doc/distilbert), lightweight version of BERT pre-trained on a collection of unpublished books (BookCorpus) and English Wikipedia articles.

- **Sentiment data:** We used a pre-trained sentiment analysis model from SpaCy (https://spacy.io/), which was pre-trained with OntoNotes, an annotated text corpus containing telephone conversations, newsgroups, blogs, and religious texts.

- **Stopwords:** We used the set of English stopwords from the NLTK package as part of preprocessing the Twitter data for tour models.

2.2 Brackground and Problem Approach

There has been significant work in finding relationships between Twitter data, either by volume of tweets, sentiment, or other features, to predict stock prices or stock price directions.

More recently there's been a body of work applying these ideas to cryptocurrencies. In particular, our paper was influenced by the work in using word embeddings and cosine similarity of tweets, along with topic modeling, to attempt to predict whether a cryptocurrency closes below, above or about the same price from the starting price. Similarly, there is a body of work computing average sentiment of tweets for cryptocurrencies and finding relationships between the sentiment and the closing price.

One complication about our work is the smaller trading volume: whereas the stocks and cryptocurrencies studied in literature are traded thousands of time per day, we observed some days without any NFT trades and about 2,000 trades in a period of four months (July 2021 - October 2021). Another unique feature about NFTs is that each traded token is closer to a collectible rather than a stock of a cryptocurrency. While the different shares of a company are worth the same at a given time, some NFTs within the same collection are more rare than others or have more appealing aesthetics. Thus, only comparing the sale price of the same NFT across time is a fair comparison, rather than comparing prices of different NFTs at a specific time interval.

In this project we use a combination of ideas from previous research of predicting cryptocurrency prices with twitter data. We formulate our problem as a classification task: given twitter data between two sale events for the same NFT, determine whether the price from the previous sale to the next sale will go up (UP), down (DOWN), or stay flat(FLAT). We use a 5% threshold to assign the labels: a decrease of more than 5% is a DOWN label, an increase of more than 5% is an UP label, and a change between -5% and 5% is FLAT.

We apply the idea of computing embeddings of the tweets and computing cosine similarity to the different label groups, and use them as features. In particular, we're trying to determine whether a collection of tweets is closer to the group of tweets with the UP, DOWN, or FLAT labels, by using cosine similarity to embeddings. Given the special characteristics of working with NFTs, we create a dataset entry for each pair of sale events for a specific NFT (analogous to creating an entry with Open and Close price for a stock or cryptocurrency, but taking into account the uniqueness of each NFT and low trading volume). Also due to the low trading volume, where an NFT might not be sold for weeks, we collect tweets for different time intervals preceding the next sale of an NFT, of one hour, 12 hours and 24 hours. The idea is to try to find a correlation between a sale and a short-term window of preceeding tweets about the NFT collection.

Apart from computing cosine similarity with

tweet embeddings, we also perform topic modeling on the tweets through the Latent Dirichlet Allocation implementation in sklearn (https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.LatentDirichletAllocation.html). LDA creates a probabilistic distribution of a fixed number of topics n in a text dataset, based on word co-occurrences. Similar to another paper, we compute TFIDF vectors from tweets, and we train and apply an LDA model with $n = 10$, to obtain the probabilities of the presence of a top-10 topic in the TFIDF vectors of a sequence of tweets between sale events, and use these probabilities as features in our model.

We also add the number of tweets as a feature, which can be used as a weighting factor for features related to text analysis.

Finally, similar to another approach, we perform sentiment analysis over the tweets between sale events using the spaCy framework. The model is pre-trained on OntoNotes, and provides a probability score for three sentiments (positive, neutral, negative). We use the average probability of each of the sentiments across all tweets between sale events, as well as the number of tweets per sentiment, where a tweet is considered to have a specific sentiment according to the largest probability provided by the sentiment analyzer.

2.3 Parsing tools used

We queried the Twitter and OpenSea APIs to obtain tweets and NFT sale events , respectively. The endpoints provided json data, so we implemented some functions to flatten the json into CSV rows, extract the subset of fields (tweet id, author id, created at, text for tweets; NFT id, sale date, and sale price for NFT sales), and generate CSV files to be read as a Pandas dataframe.

2.4 Data preprocessing

We cleaned the tweets by applying the following pre-processing steps:

1. Conversion to lowercase.
2. Removal of extra spaces.
3. Removal of URLs.
4. Word tokenization using the NLTK tokenizer (https://www.nltk.org/).
5. Removal of stopwords, using the set of NLTK stopwords.
6. Applied stemming using the NLTK Porter stemmer.

We created the dataset entry for each pair of consecutive NFT sale events, where the starting price is the price of the previous sale and the price of the next sale. Then, we computed the target label for each entry: a decrease of more than 5% corresponds to an *DOWN* label, an increase of more than 5% corresponds to an *UP* label, and a change within 5% of the starting price is the *FLAT* label.

We divided the dataset into 80% training and 20% testing. Similar to previous work, we extracted a sample of the dataset that is balanced in terms of classification labels.

3 Features

In this section we describe all the features computed from the Twitter data that are fed as inputs to our classification model.

3.1 DistilBERT Embeddings and Cosine Similarity to Tweet Groups

Previous research in using Twitter data to determine direction of closing prices of cryptocurrencies used DistilBERT word embeddings as a feature. DistilBERT is a lightweight version of BERT, pre-trained on a collection of unpublished books (BookCorpus) and English Wikipedia articles. It has 40% less parameters and runs 60% faster than BERT, while preserving 95% of the performance on the GLUE natural language understanding benchmark, according to[https://arxiv.org/abs/1910.01108], which makes it a popular alternative for computing word embeddings.

For each dataset entry, we concatenated all the tweets for the last t hours before the end sale (where $t = 1, 12, 24$). Then, we compute the embeddings using the pre-trained DistilBERT output layer, which returns 768 language features for the input text. After computing the embeddings for all entries in the dataset, we concatenate all the tweets with the same label, which results in three groups of concatenated tweets (*UP*, *DOWN*, and *FLAT*). Then, for each entry, we compute the cosine similarity between the embeddings of the tweets for the data point and the embeddings of the three groups of concatenated tweets for the labels. We use the cosine similarity to each group as a feature, as well as an indicator variable of which group has the greatest similarity to the data point.

3.2 Latent Dirichlet Allocation Topic Modeling

Latent Dirichlet Allocation (LDA) topic modeling is also used in predicting the direction of end-of-day closing prices in cryptocurrencies.

We compute TFIDF vectors from the concatenation of the tweets for the last $t = 1, 12, 24$ hours preceding the end sale of an NFT, and apply the LDA topic modeling from sklearn(https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.LatentDirichletAllocation.html) with $n = 10$ topics. We add 10 features to our datapoint, corresponding to the probabilities for each of the top-10 topics in the concatenation of the tweets preceding the NFT sale.

3.3 Sentiment Analysis

Sentiment analysis has been a key componenent in many research papers that use Twitter data in predicting stock and cryptocurrency prices or price directions. We use the spaCy sentiment analyzer (https://spacy.io/universe/project/eng_spacysentiment), which has been used in previous research for predicting price fluctuations in cryptocurrencies.

For each datapoint, we compute the sentiment probability distribution (positive, neutral, and negative) for each tweet in a given time window of $t = 1, 12, 24$ hours before the sale. Then we add the following six features to our data point: the average probability for each sentiment polarity across all tweets in the time window, and the number of tweets per polarity (e.g. a tweet is positive if the probability of positive sentiment is greater than the neutral and negative sentiment, etc.).

4 Model

In this section we discuss the models that we used for our classification task.

4.1 Baseline Model

Given that we have a balanced test dataset, our baseline model corresponds to randomly guessing one of the three candidate labels, which results in a 33% accuracy and 50% Area Under Curve (AUC).

4.2 Advanced Models

Similar to previous research in cryptocurrency price direction, we ran out-of-the box models, one custom RNN and one custom LSTM. We used the out-of-the-box sklearn implementations of *Random Forests*, *Decision Trees*, *K-Nearest Neighbors* with $k = 3$, *Boosted Decision Trees (AdaBoostClassifier)* and *Gaussian Naive Bayes*. We implemented a custom neural network with a 16-node LSTM layer, three relu dense layers with 16, 8 and 4 nodes, respectively, and an output softmax dense layer. We implemented a custom neural network identical to the previous network, but with an initial simple RNN layer (instead of an LSTM layer). These last two custom neural networks have the same architecture as in previous research for cryptocurrency price movements.

5 Results

In this section we discuss running the models with different features and different time windows of Twitter data.

5.1 Results for Different Feature Sets

The main features are cosine similarity to tweet groups, topic modeling, and sentiment.

- Baseline: 0.33 accuracy, 0.5 AUC.
- Random Forest: 0.346 accuracy, 0.567 AUC.
- Decision Tree: 0.373 accuracy, 0.529 AUC.
- Naive bayes: 0.42 accuracy, 0.595 AUC.
- KNN: 0.366 accuracy, 0.542 AUC.
- Boosted Decision Trees: 0.366 accuracy, 0.573 AUC.
- LSTM: 0.33 accuracy, 0.5 AUC.
- RNN: 0.33 accuracy, 0.5 AUC.

We obtained our best results with Naive Bayes, with an almost 10% improvement in accuracy and AUC with respect to the baseline. Interestingly, the LSTM and RNN models did not improve the baseline. We found that the models overfit the traning data, and tried less layers, early stopping, and dropout layers, without significant differences. We suspect having a larger dataset or a simplification of the architecture might yield improvements, but so far were unable to reproduce the success of the custom neural networks from research in cryptocurrencies.

5.2 Results for Different Time Windows

5.3 Unusual Data Patterns

6 Comparison with Existing Research

In the most closely related work to ours, the researchers were able to obtain an 61-88% accuracy in predicting the closing price direction labels for cryptocurrencies for individual tweets, which is a 28-55% improvement over the baseline. In our case, we were able to obtain an improvement of roughly 10% over the baseline in terms of both accuracy and AUC. However, our problem is more complicated that the cited research, due to low trading volume and lack of a central closing price – we resorted to using pairs of sale events to make a fair price comparison due to NFTs within the same collection being different. We can say that adding text analysis of Twitter data to an NFT trading algorithm clearly outperforms random guessing, and can be used as a signal in a more sophisticated strategy.

7 Conclusion

In this project we have studied for the first time the use of Twitter data in predicting the direction of sale prices of NFTs. This task is more complicated than previously studied problems such as stock and cryptocurrency price predictions due to low trading volume, lack of historical data in an emerging asset class, and the lack of a central price metric for an NFT collection (NFTs are different from each other, thus they sell at different prices within the same time intervals). However, we were able to significantly outperform random guessing in a balanced dataset, by roughly 10% in terms of both accuracy and AUC. Thus, the use of NLP techniques over Twitter data provides a valuable signal as part of a more sophisticated trading strategy.

8 Preamble

The first line of the file must be

```
\documentclass[11pt]{article}
```

To load the style file in the review version:

```
\usepackage[review]{acl}
```

For the final version, omit the `review` option:

```
\usepackage{acl}
```

To use Times Roman, put the following in the preamble:

Command	Output	Command	Output
{\"a}	ä	{\c c}	ç
{\^e}	ê	{\u g}	ğ
{\`i}	ì	{\l}	ł
{\.I}	İ	{\~n}	ñ
{\o}	ø	{\H o}	ő
{\'u}	ú	{\v r}	ř
{\aa}	å	{\ss}	ß

Table 1: Example commands for accented characters, to be used in, *e.g.*, BibTeX entries.

```
\usepackage{times}
```

(Alternatives like txfonts or newtx are also acceptable.)

Please see the LaTeX source of this document for comments on other packages that may be useful.

Set the title and author using `\title` and `\author`. Within the author list, format multiple authors using `\and` and `\And` and `\AND`; please see the LaTeX source for examples.

By default, the box containing the title and author names is set to the minimum of 5 cm. If you need more space, include the following in the preamble:

```
\setlength\titlebox{<dim>}
```

where `<dim>` is replaced with a length. Do not set this length smaller than 5 cm.

9 Document Body

9.1 Footnotes

Footnotes are inserted with the `\footnote` command.[1]

9.2 Tables and figures

See Table 1 for an example of a table and its caption.
Do not override the default caption sizes.

9.3 Hyperlinks

Users of older versions of LaTeX may encounter the following error during compilation:

```
\pdfendlink ended up in
different nesting level
than \pdfstartlink.
```

This happens when pdfLaTeX is used and a citation splits across a page boundary. The best way to fix this is to upgrade LaTeX to 2018-12-01 or later.

[1] This is a footnote.

9.4 Citations

Table 2 shows the syntax supported by the style files. We encourage you to use the natbib styles. You can use the command `\citet` (cite in text) to get "author (year)" citations, like this citation to a paper by **?**. You can use the command `\citep` (cite in parentheses) to get "(author, year)" citations (**?**). You can use the command `\citealp` (alternative cite without parentheses) to get "author, year" citations, which is useful for using citations within parentheses (e.g. **?**).

9.5 References

The LaTeX and BibTeX style files provided roughly follow the American Psychological Association format. If your own bib file is named `custom.bib`, then placing the following before any appendices in your LaTeX file will generate the references section for you:

```
\bibliography{custom}
```

You can obtain the complete ACL Anthology as a BibTeX file from https://aclweb.org/anthology/anthology.bib.gz. To include both the Anthology and your own .bib file, use the following instead of the above.

```
\bibliography{anthology,custom}
```

Please see Section 8 for information on preparing BibTeX files.

9.6 Appendices

Use `\appendix` before any appendix section to switch the section numbering over to letters. See Appendix A for an example.

10 BibTeX Files

Unicode cannot be used in BibTeX entries, and some ways of typing special characters can disrupt BibTeX's alphabetization. The recommended way of typing special characters is shown in Table 1.

Please ensure that BibTeX records contain DOIs or URLs when possible, and for all the ACL materials that you reference. Use the `doi` field for DOIs and the `url` field for URLs. If a BibTeX entry has a URL or DOI field, the paper title in the references section will appear as a hyperlink to the paper, using the hyperref LaTeX package.

Acknowledgements

This document has been adapted by Steven Bethard, Ryan Cotterell and Rui Yan from the instructions for earlier ACL and NAACL proceedings, including those for ACL 2019 by Douwe Kiela and Ivan Vulić, NAACL 2019 by Stephanie Lukin and Alla Roskovskaya, ACL 2018 by Shay Cohen, Kevin Gimpel, and Wei Lu, NAACL 2018 by Margaret Mitchell and Stephanie Lukin, BibTeX suggestions for (NA)ACL 2017/2018 from Jason Eisner, ACL 2017 by Dan Gildea and Min-Yen Kan, NAACL 2017 by Margaret Mitchell, ACL 2012 by Maggie Li and Michael White, ACL 2010 by Jing-Shin Chang and Philipp Koehn, ACL 2008 by Johanna D. Moore, Simone Teufel, James Allan, and Sadaoki Furui, ACL 2005 by Hwee Tou Ng and Kemal Oflazer, ACL 2002 by Eugene Charniak and Dekang Lin, and earlier ACL and EACL formats written by several people, including John Chen, Henry S. Thompson and Donald Walker. Additional elements were taken from the formatting instructions of the *International Joint Conference on Artificial Intelligence* and the *Conference on Computer Vision and Pattern Recognition*.

A Example Appendix

This is an appendix.

Output	natbib command	Old ACL-style command
(**?**)	`\citep`	`\cite`
?	`\citealp`	no equivalent
?	`\citet`	`\newcite`
(**?**)	`\citeyearpar`	`\shortcite`

Table 2: Citation commands supported by the style file. The style is based on the natbib package and supports all natbib citation commands. It also supports commands defined in previous ACL style files for compatibility.