Crypto Anomaly Detection

Changhyun Lee
Center for Data Science
New York University
New York City, NY, USA
cl4017@nyu.edu

Howell Lu
Center for Data Science
New York University
New York City, NY, USA
hl4631@nyu.edu

Victoria Xie
Center for Data Science
New York University
New York City, NY, USA
xx2179@nyu.edu

Oussama Fatri
Center for Data Science
New York University
New York City, NY, USA
of282@nyu.edu

Abstract

In cryptocurrency markets, the price of crypto assets can diverge across markets due to numerous reasons (e.g. exchange downtime and trade volumes). Therefore, outlier detection is extremely important for ensuring that erroneous market data does not distort price feeds. This project aims to detect anomalies (outliers) in the price of cryptocurrency transactions across exchanges and assets using various models including Z-score thresholds, Logistic Regression, Random Forest, Ensemble Voting, LSTM, and XGBoost. As a result, the XGBoost model achieves the highest AUC of 0.99, with taker fees being its most important feature. The final model has the potential of being deployed by Ripple as an inference layer on top of various financial models to ensure data quality.

1 Introduction

There has been increased development of financial instruments that are dependent on the spot prices of crypto assets, such as call-and-put options and secured loans. However, the question of obtaining pertinent, accurate, and timely information is still one that remains to be answered, as accuracy often conflicts with timeliness, and oftentimes data is only partially correct or wholly incorrect. Incorrect data can cause the execution of many financial options and orders to be premature. In most regards, crypto financial assets work in the same manner. Traditional financial assets use clearinghouses and are protected by government regulations, insurance policies, and financial institutions. However, cryptocurrencies are fundamentally immutable, causing each and every transaction to be permanent. Cryptocurrencies also do not have the same level of protection that traditional financial assets do. Therefore, it is imperative that anomalies are labeled and processed so that they do not cause catastrophic, irreversible damage. After thorough research, we propose to solve this problem by implementing an LSTM algorithm to label anomalous cryptocurrency data and using the XGBoost algorithm to classify anomalies. This method succeeded and resulted in a 99% AUC when it comes to predicting short-term volumetric data. What differentiates this work from other related types is the magnitude of the exchanges used, and the granularity of minute-by-minute data coupled with uncommonly used features such as commission size and arbitrage possibilities.

Preprint. Under review.

2 Related Work

There is an abundance of past literature on the subject of financial outlier detection, albeit most of it lies in the realm of fraud detection. Data validation is a widely accepted standard in financial institutions. Commonly implemented applications include credit card fraud detection, which is a highly mature field that is implemented almost universally with an extremely high degree of accuracy and success.

Techniques from credit card fraud are commonly used to analyze wallet-to-wallet transactions from different cryptocurrencies. The k-nearest neighbors (KNN) algorithm is commonly used to detect anomalies in both wallet and credit card transactions [**Bin Sulaiman**, **Rejwan**, **et al.**]. Furthermore, there is recent research on the application of LSTM models to identify anomalous data, as with [**Azevedo**, **V.**, **Hoegner**, **C.**] who applied LSTM to traditional stocks and [**Livieris**, **Ioannis E.**, **et al.**] whom applied LSTM on cryptocurrencies. However, there is limited related literature on anomalous cryptocurrency exchange ticker information.

Currently, Ripple employs rules-based, hard-coded critical values to check for anomalies. The limitation of this approach is that it has to involve people in the loop, which is time-consuming, and imposes a a number of assumptions about the market. As an alternative, Ripple is looking for a dynamic, precise, and data-driven solution to replace this inference layer that filters incoming data for its financial models.

3 Problem Definition and Algorithm

3.1 Task

There has been an ever-growing market for financial instruments such as options, and crypto-loans which are derived from cryptocurrencies. Furthermore, many trading models and portfolios are being built around the spot prices of many cryptocurrencies. In addition, since the very nature of cryptocurrency is founded on the principle of immutability, there are very few options available to reverse any potential damages that are done from incorrectly quoted data, market manipulation, and other errors. However, on the other side of the coin, crypto exchanges and financial instruments are severely hamstrung if options and orders are not placed and executed in a timely manner. In traditional financial markets, rule-based circuit-breakers are in place when exchange prices move too much. Consequently, the main objective of this study is to create a more nuanced approach with the help of machine learning.

The task can be broken down into several parts. We first intend to input real-time data from various exchanges with regard to price and volume and to be able to label them as either anomalous or not in a quick time frame. We will then work on introducing a method of labeling anomalies quickly and accurately through an LSTM model, which takes in minute-by-minute data and appends binary indicators for anomalies to the dataset. From there, it is our plan to conduct data cleaning and feature engineering, experiment with various models, cross-validate model performance, tune hyperparameters, and arrive at a final model that best serves the purpose of anomaly detection.

3.2 Algorithm

In developing our algorithms to detect anomalies in crypto transactions, we created two different approaches to label anomalies in the dataset. The first approach uses the z-score model, which identifies pricing changes above the threshold of 3 z-scores and marks the data points as a boolean value of True in order to represent an outlier.

In the second approach, we incorporated a long short-term memory (LSTM) neural network using volumetric data. LSTM is a type of recurrent neural network (RNN) developed to solve long-term dependency problems. Instead of treating each data sequence independently, it retains patterns in the previous sequence. Cell state, hidden state, and input data determine the long-term memory of the network and the output. We also incorporated two dropout layers to avoid overfitting. RepeatVector layer repeats the incoming inputs, and the TimeDistributed layer applies Dense layers to every feature-numbered slice of the data. Each data point's loss between the actual and predicted values from the LSTM model is labeled as an anomaly if the loss exceeds the 80% threshold. The derived

```
model = Sequential ([
        LSTM(128, input_shape=(timesteps, num_features)),
        Dropout(0.2),
        RepeatVector(timesteps),
        LSTM(128, return_sequences=True),
        Dropout(0.2),
        TimeDistributed(Dense(num_features)) ])
```

Figure 1: LSTM Model Parameters

labeled dataset only consists of prediction data points, and train data are removed to avoid data leakage. The structure of the LSTM model is presented as follows 1:

With the labeled dataset, we moved on to the main classification algorithms for experiments. We conducted different sets of experiments for the two labeling approaches. In regards to the z-score labeled dataset, we performed logistic regression, decision trees, random forest, and ensemble voting, which combines all the above algorithms.

Logistic regression utilizes the sigmoid function illustrated below to create decision boundaries for the binary classification. In the formula below, s(z) represents output between 0 and 1 (probability estimate), and z denotes the input to the function (prediction i.e.. mx + b).

$$S(z) = \frac{1}{1 + e^{-z}}$$

Decision trees are built by recursively splitting our training samples using the features from the data. This is done by evaluating metrics, such as the Gini index or Entropy, for classification decisions. Random Forest is an ensemble version of decision trees with the addition of bagging and feature randomness. Each individual tree with selective features outputs a class prediction, and the model's final prediction is the class with the most votes.

The ensemble voting classifier combines logistic regression, decision trees, and random forest using soft voting. Fine-tuned estimators are fed into the Scikit-Learn VotingClassifier class, where each model individually calculates the probabilities of binary classes. Then, the weighted average of those probabilities is utilized to produce the final vote for the class.

In regards to the LSTM labeled dataset, we utilized an XGBoost model, as our previous experiments hinted at the need for a more complex model to reduce bias and alleviate underfitting. XGBoost works by training several decision trees. Each tree is trained on a subset of the dataset, and the predictions from each tree are combined to form the final prediction. A key characteristic of XGBoost is that base estimators are built sequentially in an attempt to reduce bias for the combined estimator.

4 Experimental Evaluation

4.1 Data

The dataset contains long, formatted, minute-by-minute data from 42 exchanges on five different cryptocurrencies: Bitcoin, Ethereum, Ripple, Cardano, and Solana.

There were a total of 13 features, as shown below:

- *time_open*: the timestamp that indicates the time frame in which the data point occurred
- *exchange*: the exchange where the data point originated from
- *base*: refers to the first currency in a transaction pair
- *counter*: refers to the counter or the second currency in a transaction pair
- *price_open*: the price of the base currency at the time that it was first traded within the timestamp
- *price_close*: the price of the base currency at the time that it was last traded within the timestamp
- *price_high*: the highest price that the base currency was traded at during the time frame

- *volume_base*: the total amount of base currency traded during the time frame
- *high-low*: the discrepancy between the highest and lowest price of each crypto asset in a given day
- *taker_cost*: calculated by finding the amount of cryptocurrency remaining after exchange commissions are paid. Commissions were calculated by how much the exchange would charge to a client which spent 50,000 USD on that exchange within the last 30 days
- *size (largest_ex_loc)*: the size of the exchange was determined by the average amount of bitcoin transacted within a 24-hour period during the month of September 2022. Exchanges were determined to be: *tiny* (< 10m), *small* (>10m & <100m), *normal* (>100m & <1b), *large* (>1b &<5b), *very large* (>5b & < 20b), and *huge* (>20b)
- *location*: the location where the cryptocurrency exchange was registered. The term "txhv" refers to a commonly known tax haven such as the British Virgin Islands, the Seychelles, the Cayman isles, and such. Some exchanges were labeled with "txhv" appended to another location to state that the exchange was originally in one location and then moved to a tax haven. All exchanges that were originally incorporated in China are labeled as "chinatxhv" as China banned all cryptocurrency exchanges in 2017, and every exchange was moved out of the country.
- *arbitrage*: defined as whether the data point could be used to create an arbitrage profit. It was calculated by calculating the price of buying and selling all currencies post-commission for one unit, using the open price, and labeling the ones which could be one-half of an arbitrage pair that generates an arbitrage profit by having either an exceptionally low buy price or an exceptionally high selling price. It makes the assumption that transferring cryptocurrencies between exchanges are nearly instantaneous, and transaction fees are negligible. This assumption of transferring currencies quickly and feeless is generally true with Ripple and Solana taking seconds to process while Bitcoin and Ethereum potentially take hours. Furthermore, it comes with the expectation that USDT is an adequate final currency rather than fiat US dollars.

Not all features were included for modeling (for example, *time_open* was dropped because the dataset was aggregated). Some features (e.g. *exchange* and *size*) were normalized and label-encoded later in the process. Most of the features which were retained for modeling showed very low feature correlation.

The target variable, *anomaly*, is a binary indicator of whether a data point is an anomaly or not. It was label-encoded as well.

Table 1: Example Dataset Part 1

time_open	exchange	base	counter	price_open	price_high	price_low	price_close
2022-09-29 05:26:00 UTC	aax	BTC	USDT	19546.2	19546.8	19537.9	19539.3

Table 2: Example Dataset Part 2

volume_base	high-low	taker_cost	size	location	arbitrage
21.7754	8.9	0.9985	large	uk	false

4.2 Methodology

We have 4 sets of definitions for anomalies that this project aims to detect, which are elaborated as the following:

1. Outliers captured by z-score thresholds in terms of percentage daily returns
2. Outliers in cryptocurrency prices and volumes captured by an LSTM time-series model
3. Significant discrepancies in price among different exchanges

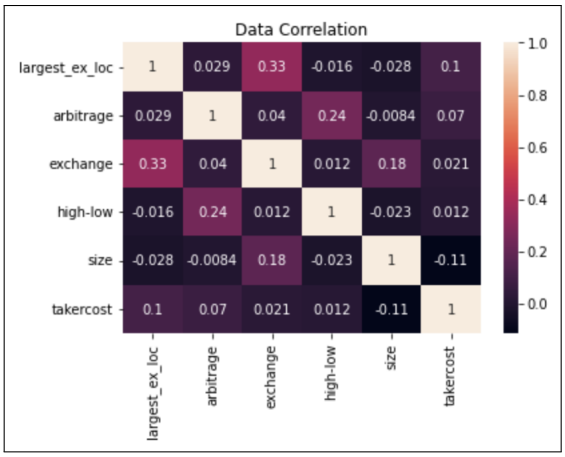

Figure 2: Correlation Matrix

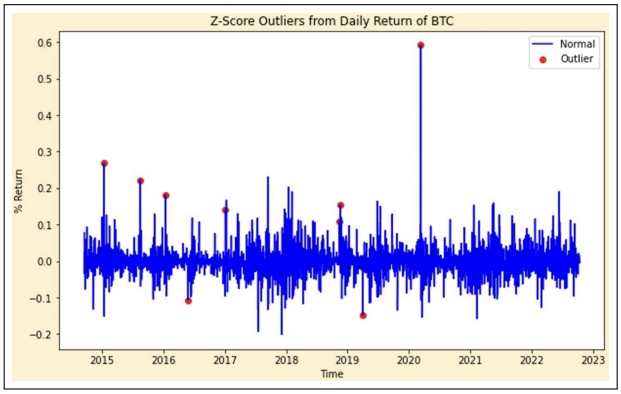

Figure 3: Labeling Approach 1: the 3-Z-score Threshold

4. Arbitrage between different exchanges

In order to capture the 4 types of anomalies mentioned above, we used 2 different approaches in identifying and labeling them. Approach 1 uses z-score thresholds 3, and Approach 2 incorporates an LSTM time-series model 4. The price differences among exchanges and arbitrage opportunities were used as features of the model. The details of the algorithms are illustrated in the Algorithm section.

We hypothesize that such anomalies exist in cryptocurrency transactions, and this study aims to produce a classification model to identify them. A 60-20-20 train-validation-test split was applied to the processed dataset. To evaluate how well the models in the experiments are able to classify

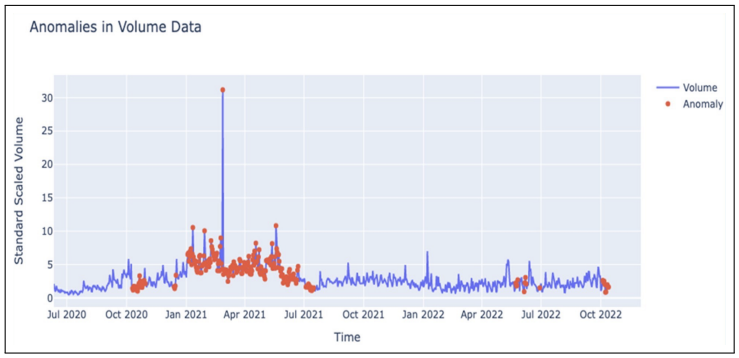

Figure 4: Labeling Approach 2: LSTM Time-Series

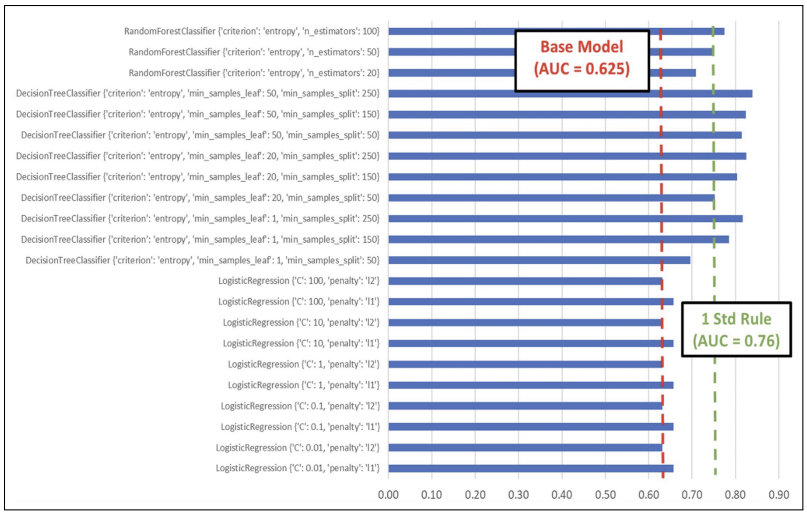

Figure 5: Hyperparameter-Tuning with Grid Search and One-Standard-Error Rule

examples as anomalies and non-anomalies, the metric used is AUC (area under the ROC curve), which provides an aggregate measure of performance across all possible classification thresholds.

In terms of experiments, since we have no particular constraints that dictate what types of algorithms should be used – the dataset has a manageable size both in terms of the number of instances and feature dimensions – we ran a horse race among different algorithms based on the two different labeling approaches.

With labels produced by the z-score method, we first explored the following 3 algorithms: Logistic Regression (as a baseline model), Decision Tree, and Random Forest. For each one of these, we conducted 10-fold cross-validation and ran a grid search 5 over an appropriate range of hyperparameters, and picked the specification using the "one-standard error rule". The rule advocates for the set of parameters that produces the result that is one standard error below the best result, for the purpose of preventing overfitting. Then, we went one step further and chose the best model within each family of learning algorithms so far and combined them using the Scikit-Learn ensemble voting classifier, which aggregates the results of the individual learners using a soft vote.

With labels produced by the LSTM model, we first re-ran the previous set of experiments and arrived at generally better results. We then experimented with the XGBoost algorithm, given the decent performance of the assemble methods in the previous part. We checked feature correlation 2, conducted stratified k-fold cross-validation, and ran a random search for hyperparameter-tuning. The model was tested for overfitting and generalization with different timeframes of the data. Additionally, we examined the feature importance produced by the algorithm in order to draw insights on the detection process.

4.3 Results

As a baseline model, the simple logistic regression without any parameter-tuning (using the default parameters: L2 penalty and C = 1.0) shows a mediocre performance with an AUC of 0.625. We ran a random forest and got an AUC of 0.661, a slight improvement over the simple logistic regression. When the 10-fold cross-validation and grid search came into play, we observed a significant improvement in AUC achieved by decision trees. After applying the one-standard-error rule, we arrived at a DecisionTreeClassifier ‘criterion’: ‘entropy’, ‘min_samples_leaf’: 20, ‘min_samples_split’: ‘50’ with a 0.76 AUC.

The ensemble voting classifier took the performance to another level, by combining the best model within each family of learning algorithms so far with a soft vote. The AUC on the validation set ended up being around 0.845 6.

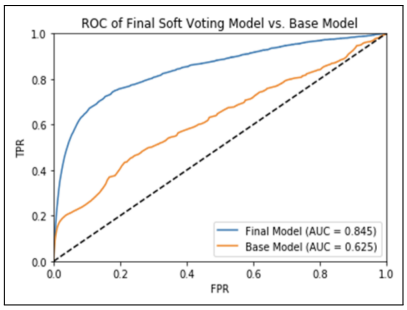

Figure 6: Baseline Logistic Regression vs. Ensemble Voting Classifier

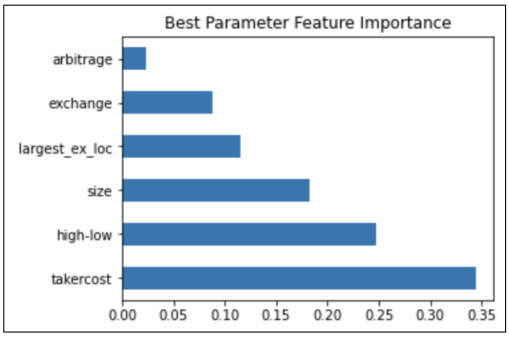

Figure 7: Top 6 Feature Importance

When the second labeling approach became available, we re-ran the above individual learners on the new labels and arrived at generally improved results – 0.803 AUC for logistic regression, 0.907 AUC for the random forest with a depth of 3, and 0.979 AUC for the random forest with a depth of 7.

Finally, the XGBoost model proved to be the best-performing model with the following set of parameters: {'subsample': 0.8, 'min_child_weight': 10, 'max_depth': 5, 'gamma': 2, 'colsample_bytree': 1.0}, leading to an almost perfect AUC of 0.99. Furthermore, results show a high level of accuracy, precision and recall in predicting both anomalous and non-anomalous classes, wich a 0.95 accuracy on the test set. Overall, we found the results sufficient enough to support our hypothesis.

Table 3: AUC Summary

Model	AUC
Logistic Regression	0.653
Decision Trees	0.798
Random Forest	0.789
Ensemble Voting	0.845
XGBoost	0.99

4.4 Discussion

Besides the near-perfect evaluation results, we took a deep dive into the feature importance 7 produced by the final XGBoost model, hoping to gain insights on why it performs so well. As the figure suggests, taker fees played the most important role in the XGBoost model. After consulting with the project mentor, we formulated the idea that maybe it was the non-linearity of taker fees that made the feature stand out.

The results are in favor of our hypothesis as such anomalies were found – outliers that are three standard deviations greater than average in terms of percentage daily returns were classified with a high degree of accuracy (0.845 AUC). In addition, the LSTM time-series model was capable of labeling the majority of what the z-score model deems anomalous, plus some more. The last criterion, whether there were substantial differences between exchanges and whether arbitrage opportunities

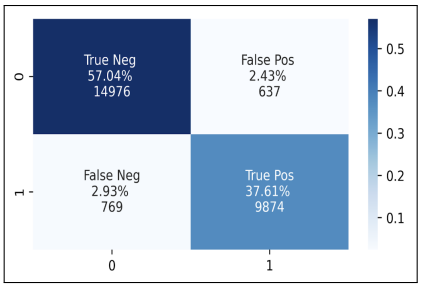

Figure 8: XGBoost Confusion Matrix

existed, was further proven to be true, as we had found many potential possibilities. We calculated that during the month of September 2022, there were 16356 minutes that could have potentiated arbitrage profit with Bitcoin out of a total of 41578 minutes recorded. The largest opportunity was a mispricing of over 40,000 dollars with FTXUS listing Bitcoin at $63,137.4 and OKEX listing Bitcoin at $18664.9.

Another inherent property that this dual LSTM classification contains is that it contains the speed necessary to act as a "circuit-breaker". Training the LSTM model and applying that LSTM model to the test set with 131281 data points took nearly seven hours. Encoding that dataset into numerical values and applying an XGBoost classification model to the same-sized dataset takes no longer than five seconds. Therefore, there is an actual possibility that such a model could be used in production as a "circuit-breaker" or a validation net of sorts.

However, there are still issues with the real-world application of such a model. Due to the computational complexity of creating this model, the LSTM was only trained on 5 days of data and tested on the 2 days that succeed it. This limitation occurs because when the sequence length of the dataset increases, the complexity of the underlying recurrent neural network increases exponentially. Therefore, the results trained on this model are likely to only be as accurate locally as there is an assumption made that this dataset is representative of the spot prices and volume of Bitcoin, which is likely not true. Another limitation of this model is that there was little experimentation with the threshold for an anomaly, and in reality, there could be a wide degree of variation regarding what a user determines an anomaly to be. The test set was fairly balanced with 41% of samples being positive and 59% being negative. It would be interesting to see if the model would produce as high of an F-1 score if there is a more stringent criterion for defining an anomaly and thereby a significantly fewer number of positives 8. Also, it is necessary to inform the readers that the model's performance is only trustworthy with regard to the current dataset. The cryptocurrency market is highly volatile, and future changes in volume and pricing trends may negatively impact the model's performance significantly, as all of our models are based on historical data. Lastly, this model was only tested with Bitcoin, and thus results cannot be deemed representative of all cryptocurrencies.

5 Conclusions

In this study, we hypothesized that anomalies, according to the four sets of definitions we formulated, exist in cryptocurrency transactions, and through extensive experiments, we successfully produced a classification model to detect such anomalies. The XGBoost algorithm was selected to be the final algorithm, with an AUC of 0.99. The results suggested that taker fees are among the most important features in classifying anomalies from cryptocurrency transactions.

The significance of the model lies in its potential of being deployed as a filtering layer on top of Ripple's financial models, in order to capture outliers dynamically, precisely, in a data-driven way without having to keep people in the loop or posing unnecessary assumptions. In real-life scenarios, the market moves extremely quickly, and having a machine learning solution as a tool to ensure data quality is of paramount importance. FTXUS offered many derivative options and the presence of this anomaly detection model likely could have prevented any automated trading programs from buying currencies from FTXUS during their bank run.

Previously, we discussed the limitation of our work that the model is not guaranteed to produce accurate results with more up-to-date data, given the ever-changing nature of cryptocurrencies.

Potential solutions to this limitation include conducting backtests using data from recently insolvent exchanges to see if the model is capable of predicting exchange insolvencies and training with different historical time frames.

For future improvement, it is also recommended that we further expand the feature list to incorporate more diverse metrics relevant to cryptocurrency transactions. Tenure of cryptocurrencies, volatility and moving averages are all good examples of potential features. In addition, feature engineering could be conducted more thoroughly on some of the existing features. For instance, continuous features such as take fees and volumes can be categorized into buckets in order for models to better capture their importance. Last but not least, as this study mainly experimented with supervised learning algorithms, unsupervised models such as k-means clustering might be worth investigating, for they have the potential to capture underlying patterns in the data.

6 Lessons Learned

One major challenge we encountered throughout the course of this project was the lack of ground truth, but that turned out to be a great learning opportunity for all of us. While we are used to dealing with clean, labeled datasets in school projects, it is not always the case outside of the standard classroom environment. Through research into related work and past literature, we decided on two different approaches to label our dataset and proceeded to the experiments with our predicted set of targets. We became more comfortable with the lack of ground truth throughout the course of the project, knowing that our experiments were already an improvement over the current practice.

We also gained familiarity with advanced machine-learning models and hyperparameter-tuning techniques by practicing extensively during the modeling phase. During the poster presentation and meetings with our project mentors, we were able to gather useful feedback and incorporate changes to our work accordingly.

7 References

[1]Bin Sulaiman, Rejwan, et al. "Review of Machine Learning Approach on Credit Card Fraud Detection - Human-Centric Intelligent Systems." *SpringerLink*, Springer Netherlands, 5 May 2022, https://link.springer.com/article/10.1007/s44230-022-00004-0.

[2]Azevedo, Vitor, and Christopher Hoegner. "Enhancing Stock Market Anomalies with Machine Learning - Review of Quantitative Finance and Accounting." *SpringerLink*, Springer US, 30 Aug. 2022, https://link.springer.com/article/10.1007/s11156-022-01099-z.

[3]Livieris, Ioannis E., et al. "An Advanced CNN-LSTM Model for Cryptocurrency Forecasting." *MDPI*, Multidisciplinary Digital Publishing Institute, 26 Jan. 2021, https://www.mdpi.com/2079-9292/10/3/287.

8 Student Contributions

Howell Lu developed the arbitrage model, worked on the features for takercost and size, tested Random Forest on the LSTM model, and formatted the LaTeX document. He attended all meetings and presentations and worked on the report.

Victoria Xie set up meetings with project mentors, contributed to the midterm and poster presentations, translated results into insights, wrote the final report, and formatted the LaTex document.

Changhyun Lee developed the LSTM labeling approach and created the labeled dataset for 41 exchanges. He utilized the dataset for building the XGBoost model. He attended all project meetings and worked on the midterm presentation, final poster, and report.

Oussama Fatri developed the z-score labeling approach, implemented the logistic regression, decision trees, random forest, and ensemble voting models, and created visualizations.

We would like to offer our sincerest gratitude to Jon Wedrogowski and Wenda Zhou for the help and feedback they provided throughout the course of this project.