Datahacks 2021

Cryptocurrency Track

Team Wild Rifters

Justin Liang

Jonathan Lo

Eric Wang

Important Background Information/Research

Bitcoin Variables:

1. Address (String)
    - A location where bitcoin transactions occur
    - Similar to a location on a map, but on the internet
    - Experts recommend using multiple addresses
2. Year (Int)
    - Year that the transaction occurred
3. Day (Int)
    - Day of the year (ranging from 1 - 365)
4. Length (Int)
    - Need to Know:
      Bitcoin Mixing is a process in which users "mix" their coins with other users and in order to preserve privacy. This is because while Bitcoin inherently doesn't contain much information about their owner, the movement of them can reveal a lot of personal information. By mixing, this allows users to preserve such info.
    - Length is the quantification of mixing on Bitcoin
5. Weight (Float)
    - Best explanation is given
    - "Quantifies the merge behavior (i.e., the transaction has more input addresses than output addresses), where coins in multiple addresses are each passed through a succession of merging transactions and accumulated in a final address. "
6. Count
    - This may be the number of bitcoins occurring in a transaction. The movement of the count may reveal information about the users.
7. Looped
    - Bitcoin splitting (or halving?) is when the number of bitcoins that can be mined from a block. It means that it occurs when the pace of bitcoin creation is cut in half. Happens roughly every 4 years
    - A bitcoin network is a peer to peer payment network. "Users send and receive bitcoins, the units of currency, by broadcasting digitally signed messages to the network using bitcoin cryptocurrency wallet software. Transactions are recorded into a distributed, replicated public database known as the blockchain, with consensus achieved by a proof-of-work system called mining (https://en.wikipedia.org/wiki/Bitcoin_network).
    - For merging bitcoin to one address: https://fixedfloat.com/blog/guides/what-is-consolidation
8. Neighbors
    - Maybe say that neighbors are the other computers in the network?

    - Bitcoin isn't centralized so it runs off of a network of other computers which transactions occur
9. Income
    - 1 bitcoin = 100 million satoshis
    - It is the smallest unit of bitcoin
    - Feasible for smaller transactions
10. Label
    - Ransomware family is the type of ransomware used to encrypt certain types of data (i.e. business) and to decrypt the data, payment through Bitcoin would be preferred due to its anonymity

<u>Additional Notes:</u>

- Identifying ransomware may enable us to identify heists
- Identifying common ransomware labels may enable us to prevent future heists
- Predicting ransomware family may enable us to track behavior
- Visualization of data is key to unlocking trends

Process

Section 0: Introduction

In recent years ransomware attacks have made national headlines over and over again for holding information hostages. In order to prevent this, we will attempt to identify ransomware attacks and the behaviors of those attacks. By understanding the nature of this, it may enable the prevention of future heists.

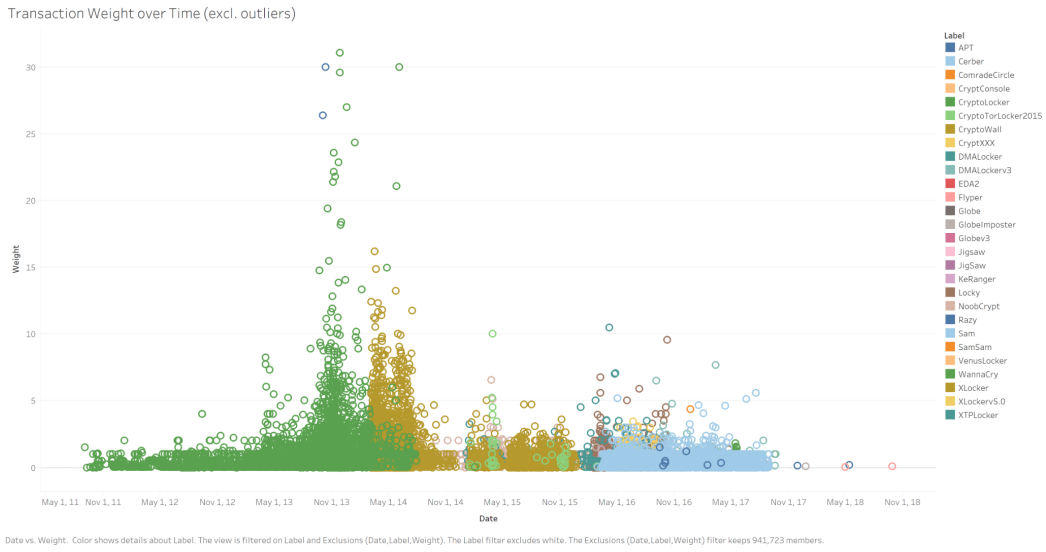
Section 1: Data Cleaning/Preprocessing

First, we used pandas to read the training data CSV into a dataframe. In order to get a quick glance at the structure, we looked at the first five rows of the training data. Moreover, we used .info() to get an insight into the data types that the dataframe contains. Proceeding, we dropped unnecessary columns and combined year and day into one column to flush out the dataset. After, we check all unique values to see if there were any null values, errors, or misspellings.

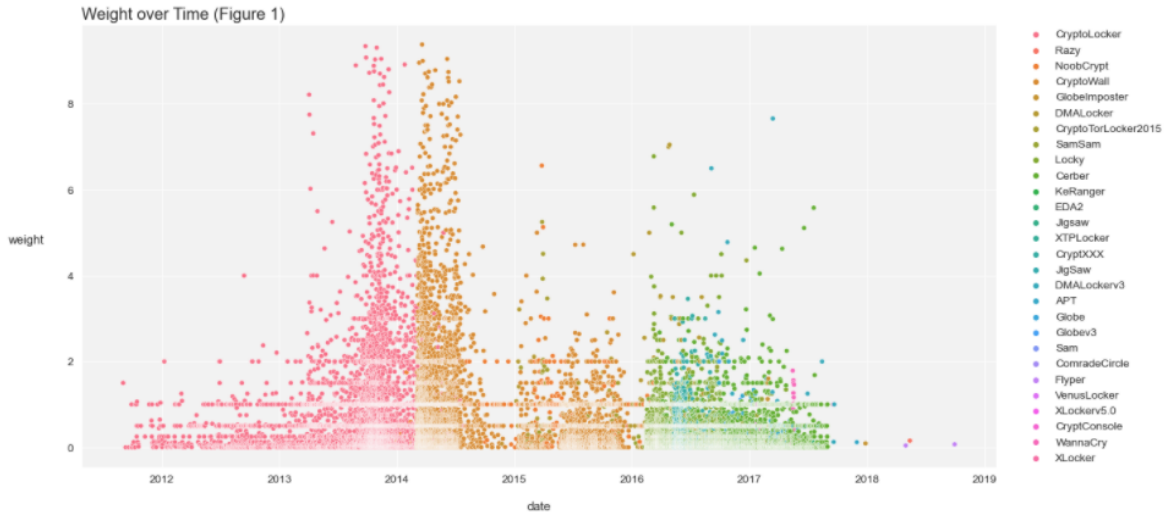
This leads us to our final steps. In order to find the top three ransomware transactions, we found the count of each label. Here we were able to find out that CryptoWall (9872), CryptoLocker (7422), and Cerber (7381) had the most transactions. Knowing this, we can proceed to the next section.

Section 2: Data Visualization

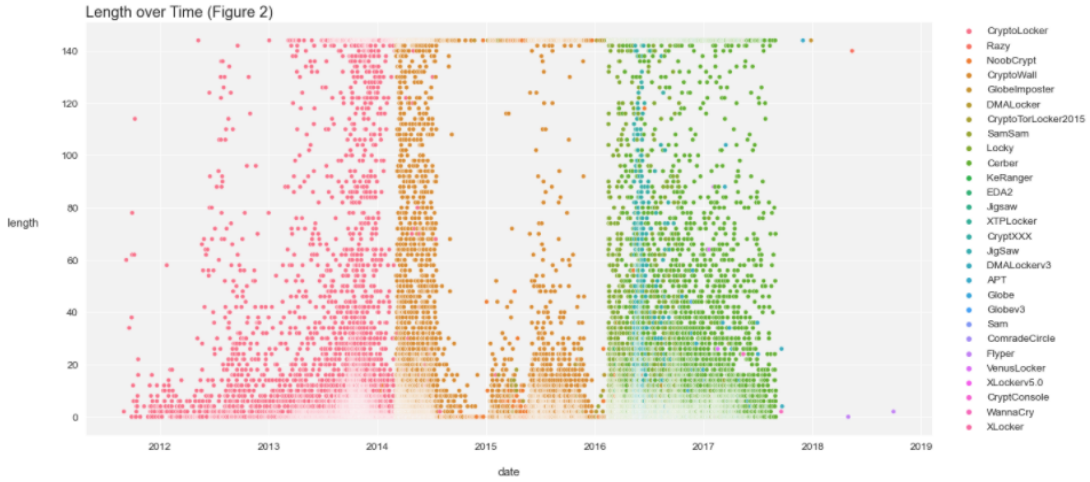
In order to find possible trends or correlation in the data, we used Tableau. To do this, we grouped different variables against each other to explore possible associations within the dataset. One example of the variables we set against each other include weight over time, and we excluded outliers to get a better visualization of the data, as the outliers don't impact the exploratory visualization process. This visualization allowed us to understand that large ransomware labels occurred in high density in large "periods" of time.



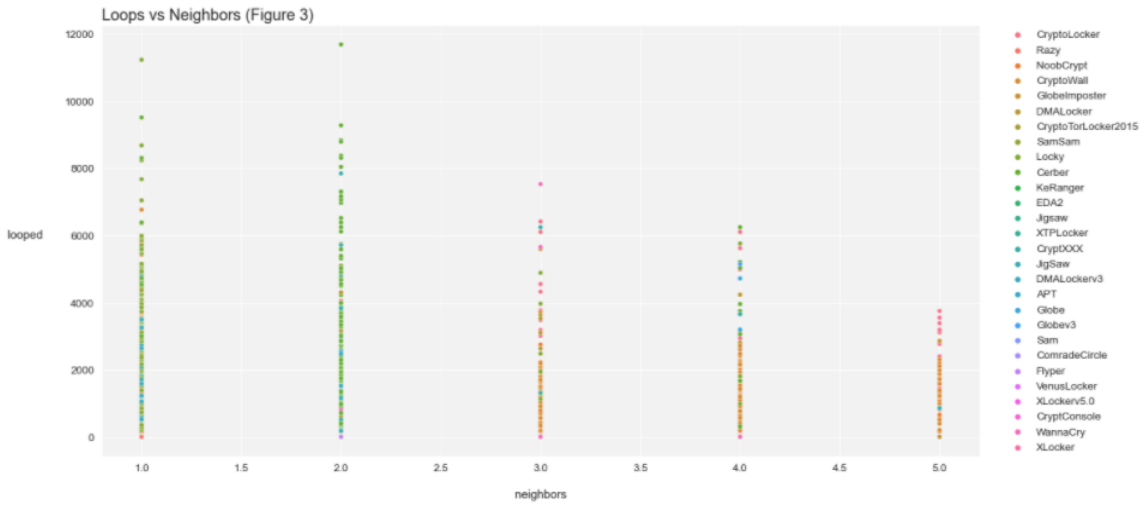
However, we expanded upon these Tableau visualizations in our Jupyter notebook because Matplotlib and Seaborn offer greater versatility and handling of the data. Since our main focus are the ransomware labels, we excluded the white labels from our visualizations. Moreover, by removing outliers and plotting, we found three key details.



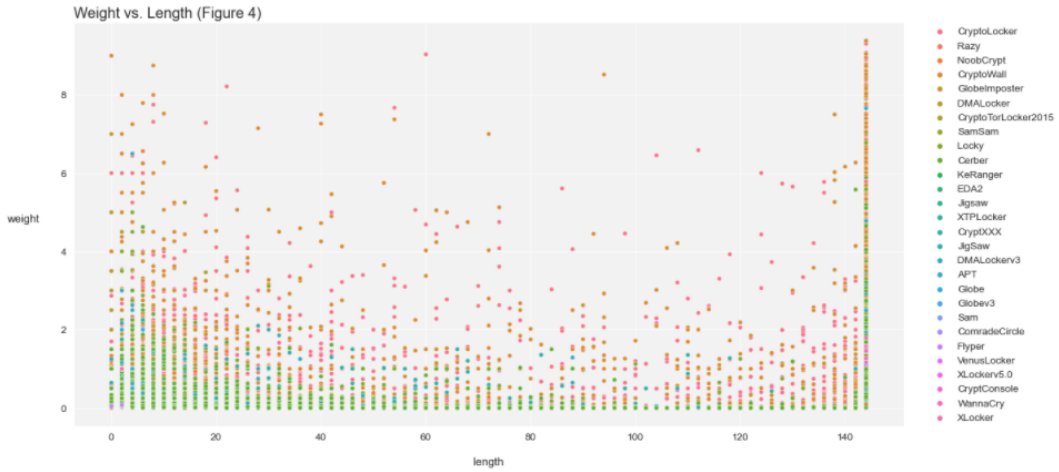
First, between these “periods”, there was some overlap between them. Second, we found that there were only a few unique ransomware labels early, but gradually became less distinct as time progressed. Third, was that we can see horizontal lines at different weight intervals. We came to the conclusion that this was some kind of rounding process by ransomware.



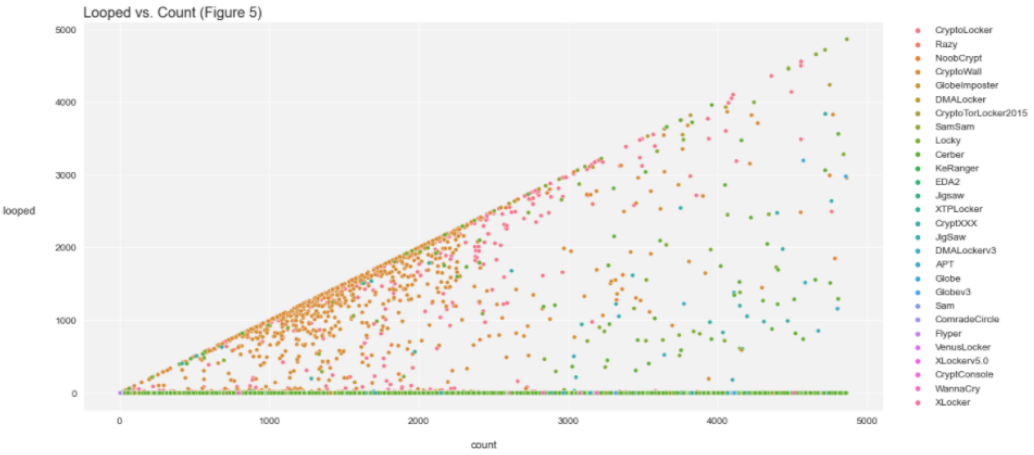
In figure 2, we can see that the length of the ransomware caps at around 145 for all ransomware transactions. Many users may not mix their coin to that degree and we can expect their length to settle near the bottom. However, we may be able to detect ransomware if we look at how well mixed the coins are.



In figure 3, we see that there is a greater density of loops among all numbers of neighbors below 6000 loops. From this, we can see that regardless of the number of neighbors that the number of times that each transaction is looped (split, moving coins within a network, and merging) a similar amount of times. However, it must be noted that the type of the ransomware changes among the number of neighbors, so the identification of ransomware can be noted based upon the number of neighbors counted based upon the number of loops.

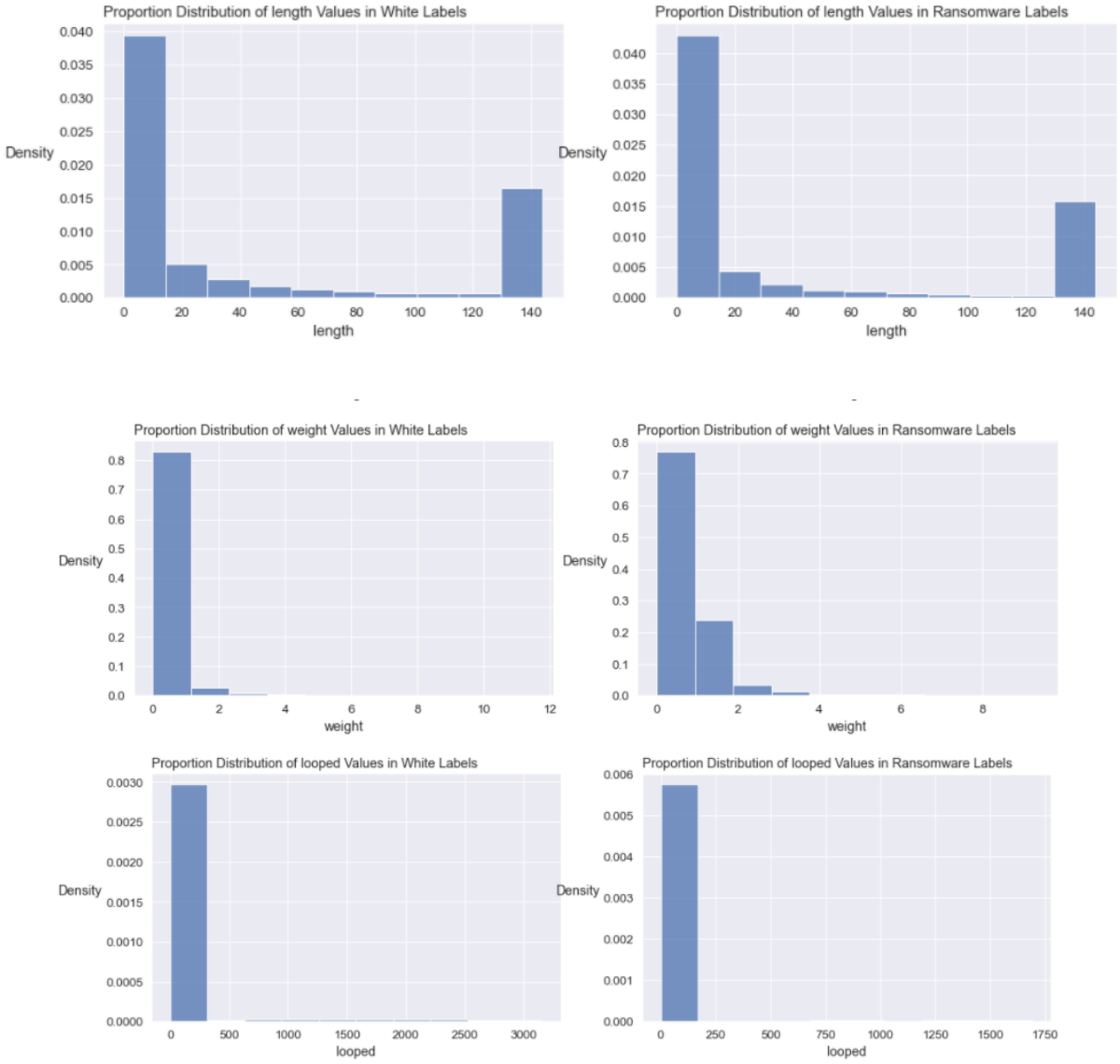


In figure 4, we can reach the same conclusion as we did for figure 2. Much of the ransomware labels are concentrated at 145. Moreover, The weight of these transactions occurring at 145 tend to have a significantly higher weight than at any other lengths. By tracking the weight vs length distribution, it may lead into some insights that relate to identifying ransomware.



In figure 5, we can see that there is a positive correlation when comparing the weight of a transaction and the count of it. The number of bitcoin in a transaction may indicate how well mixed the transaction is. Overall, this may not be able to help in identifying ransomware but may is important enough to note.

(Figure 6)


Proportion Distribution of length Values in White Labels
Density
length
Proportion Distribution of length Values in Ransomware Labels
Density
length
Proportion Distribution of weight Values in White Labels
Density
weight
Proportion Distribution of weight Values in Ransomware Labels
Density
weight
Proportion Distribution of looped Values in White Labels
Density
looped
Proportion Distribution of looped Values in Ransomware Labels
Density
looped

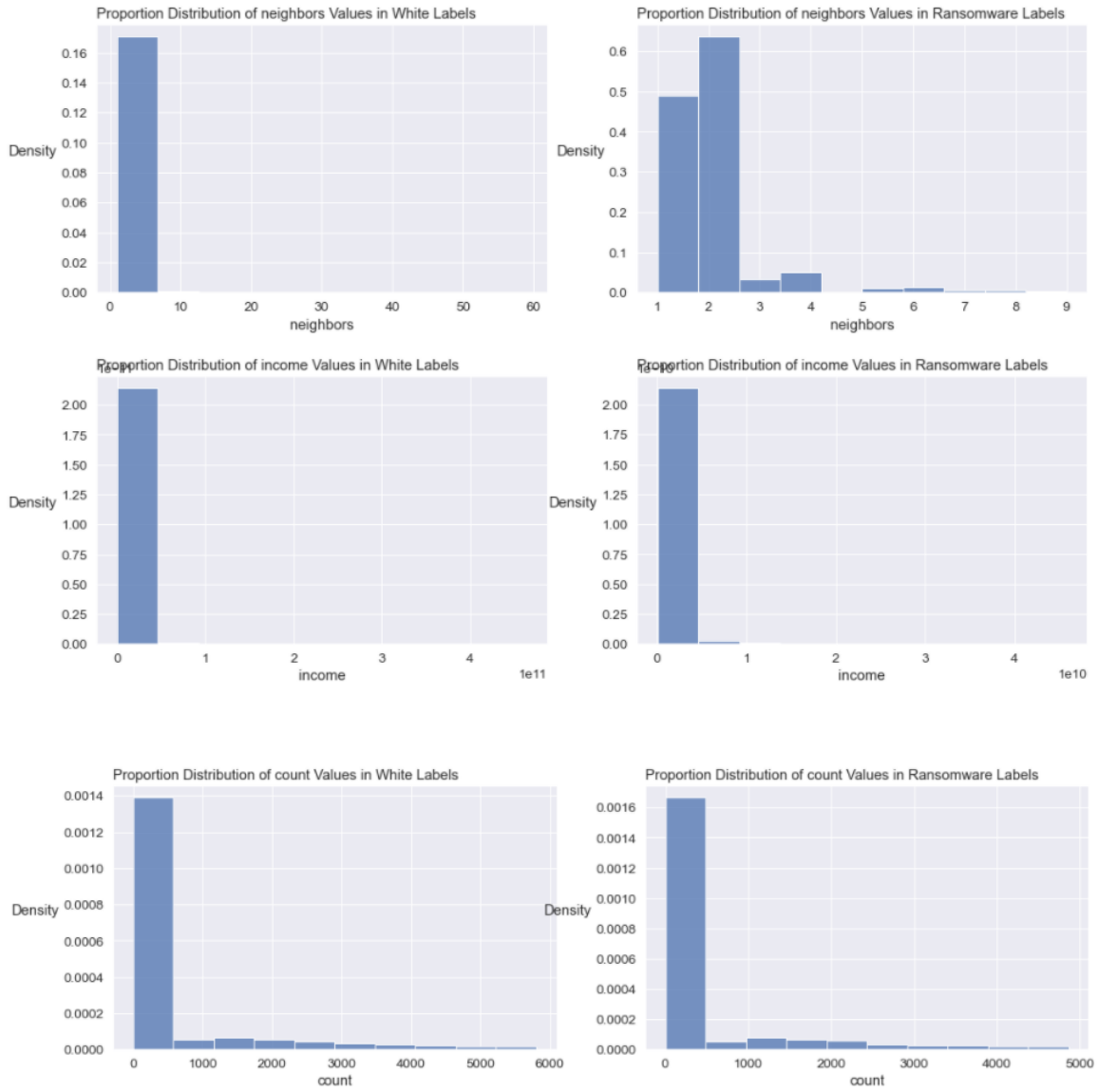


Figure 6 gives some insight on how the histogram distributions compare between white labels and ransomware labels across all features. Overall, the white label and ransomware label distributions appear similar for most features. It can be noted that the white label sample size is much greater than the number of ransomware label samples. This results in the white labels having a greater spread of values, even after outliers have been removed.

Section 3: Hypothesis/Experimental Testing

Since we only want information in regards to label, our hypothesis will only contain the labels length, weight, count, looped, date, and label. For out first hypothesis test, hypotheses as follows:

$\mathbf{H_0}$: The average weight of white labels equal the average weight of non-white labels

$\mathbf{H_A}$: The average weight of white labels does not equal the average weight of non-white labels

$\boldsymbol{\alpha}$: 0.05

_____Due to the fact we are comparing two groups in a sample, we want to conduct an independent samples t-test. This involves calculating the test statistic (t-value) to evaluate against the set t-values to calculate the probability value. We first calculate the mean, count, and standard deviation of each group, and use these statistics to calculate the pooled variance, which follows the formula:

$$s_p^2 = \frac{(n_1-1)s_1^2+(n_2-1)s_2^2}{n_1+n_2-2}$$

Using this formula, we are able to calculate the pooled standard deviation which is then used to calculate the t-value. We ended up getting a t-value of -4.1733 which resulted in a p-value of 3.002e-05 which allowed us to conclude a statistically significant difference in average weights between white and ransomware labels.

We repeated this process for the number of neighbors and got a p-value of 0.0802, so we fail to reject the null hypothesis, meaning that there is no statistically significant difference between the average number of neighbors for white and ransomware labels.

Section 4: Forecasting/Machine Learning

For the prediction of ransomware labels, we decided to use Scikit-learn's pipeline and Bernoulli classifier. We believed this best fit our purpose because the Bernoulli Naive Bayes classifier is a multivariate classifier that "is suitable for discrete data" according to the Scikit-learn documentation. We began first by selecting optimal features for our baseline pipeline model. The features we selected included address (qualitative), length, weight, count, looped, neighbors, income. In order to fit our qualitative data, we used a onehotencoder to transform the address feature. Then, we used the standard scaler to standardize the rest of the quantitative features.

After creating our pipeline, we split the data into training and testing sets using an 80-20 split in order to test the accuracy of our model. After testing the algorithm on our testing set, we used the accuracy score metric and got an accuracy score of 0.9858.

Section 5: Proposal

In the future, we can utilize machine learning methods, such as the ones used in this project, to retroactively track ransomware bitcoin transactions and flag specific bitcoin addresses that were exposed in these transactions. These bitcoin addresses can then be blacklisted from future transactions in order to prevent further ransomware attacks. However, there are issues still present. In order to prevent backtracking, ransomware may create and destroy addresses to prevent this. Ideally, our machine learning model will be able to produce results fast enough to circumvent this issue but that is not entirely feasible.

The next step in this process is to utilize this foundation that is set up to build a new model that may be able to predict attacks. If there can be a security delay between the reception of transaction information and the permanent transaction going through, then a real-time machine learning model could immediately block the transaction. Although difficult, we believe that it is not entirely impossible depending on the nature of these transactions, as there could be additional security concerns.

Section 6: Conclusion

Although this project was primarily focused on identifying ransomware and the family it belongs to, it may lead to more. In the beginning, we cleaned the data in order to prune out any redundant information and consolidated the data into a more streamlined and interpretable form. Moving on, one of the most significant steps in our process was the data visualization. It was important to see what trends lie behind the data and to capitalize on those trends. Moreover, it provided a clearer image of the direction the project was supposed to be headed. Using the data visualization, we were able to create a hypothesis test based on labels of the ransomware. This allowed us to move onto the machine learning process. Utilizing the Scikit package, we selected certain features that indicated ransomware and fit our training data. After running our algorithm, we were able to find results with less than a 0.05 margin error. Understanding the process of this decision making may lead the future development of new algorithms in the prevention of attacks.