Report - Bitcoin Track - Team CryptoWall

Team members: Tae Kun Kim, Johnny Liu, William Luu, Ester Tsai
Completed on: April 11, 2021

Table of Contents

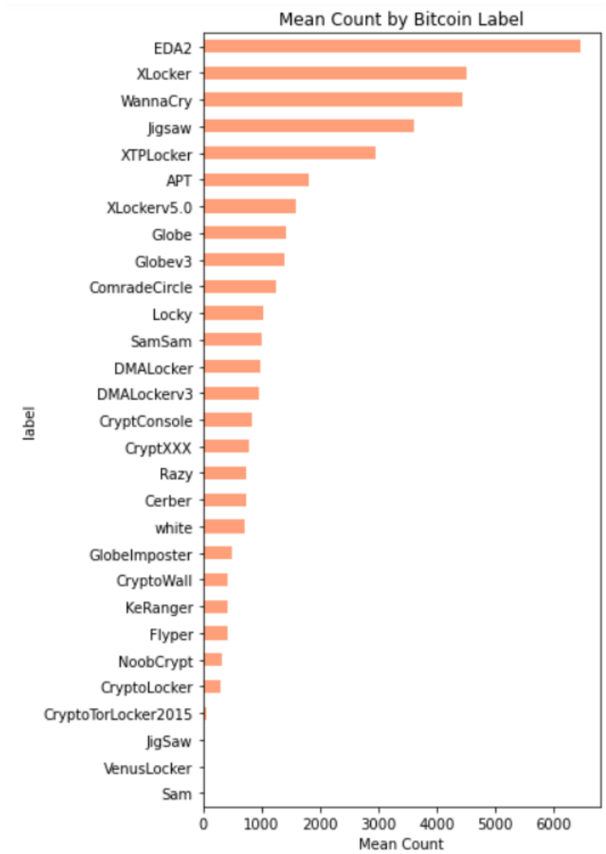
Intro

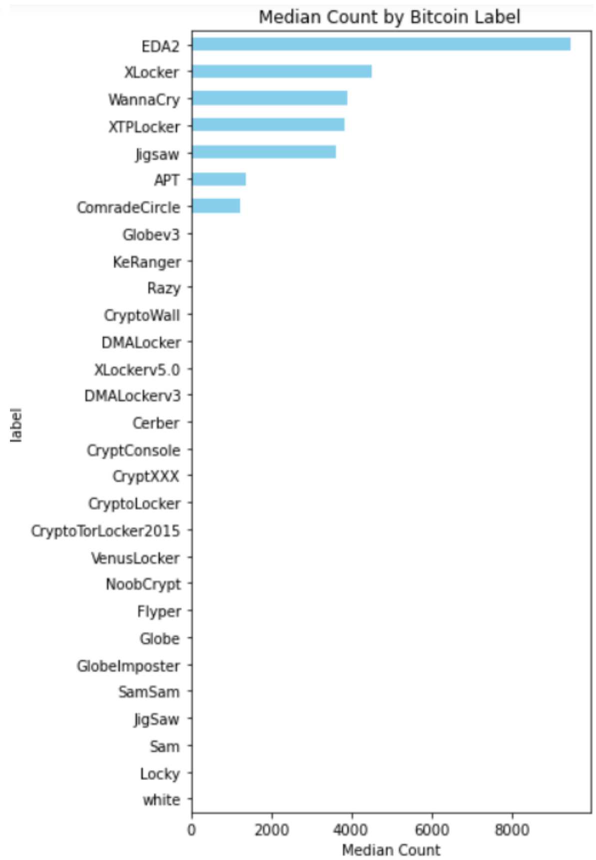
This report details how Team CryptoWall pre-processed, visualized, analyzed, and concluded the dataset for ransomware payments in the Bitcoin network between the years 2009 and 2018. The goals are to: 1) "Determine the top three ransom labels that have the most ransom transactions"; 2) "Define a machine learning model most appropriate for classifying heist incidents into ransomware families"; and 3) "Define a model to predict: a. If a future transaction is ransom or not, and if it is, b. The ransomware family it belongs to".

Data Cleaning/Pre-processing

Before we began cleaning the data, we had to understand the features variables and what each column represents. The main variable of interest is the label column that represents the label of the transaction. We can group them by "white" label where the transaction is not likely to be ransomware and "non-white" labels that have the known to be a specific ransomware family.

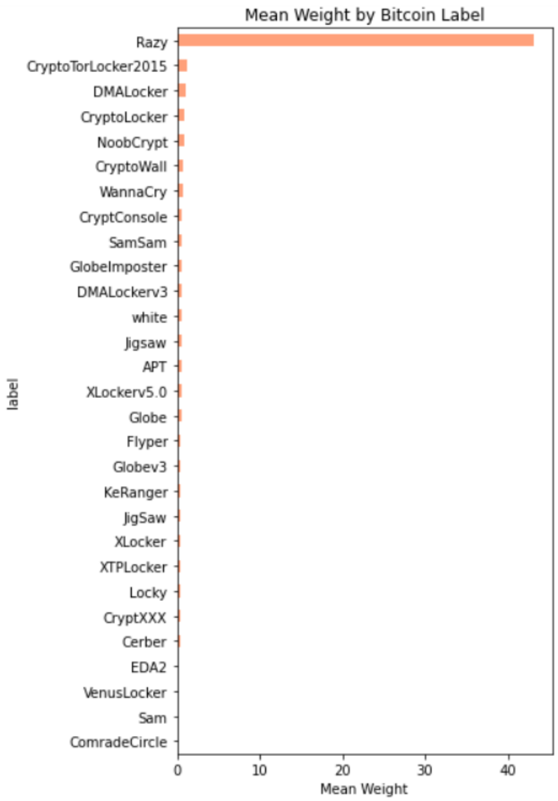
Taking a general glance at the dataframe, we noticed that counts are either one or a significantly larger integer. We took the average of the counts for each ransomware family and noticed that white labels on average would have significantly less counts than non white labels. We interpreted this trend in the sense that it was more likely a ransomware transaction would contain more information than a non-ransomware transaction. It is most likely that ransomware transactions would need more information because of the multiple inputs it must take from the weights feature.

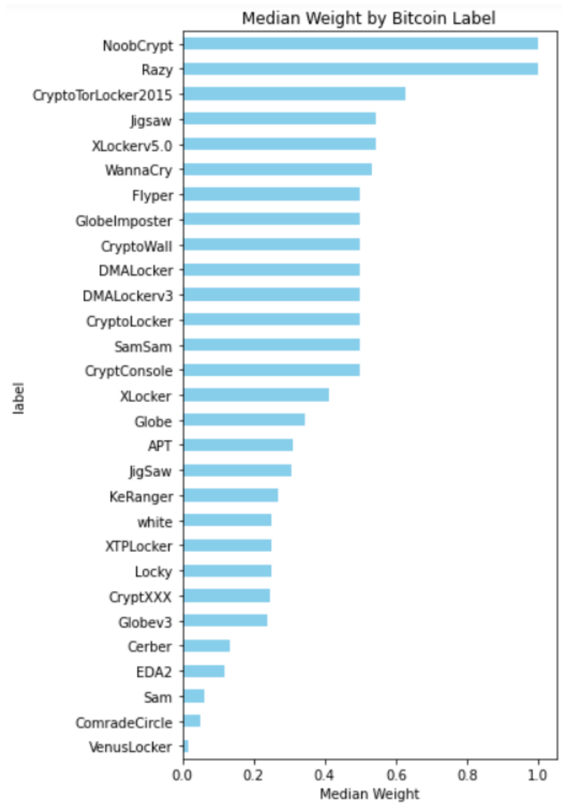




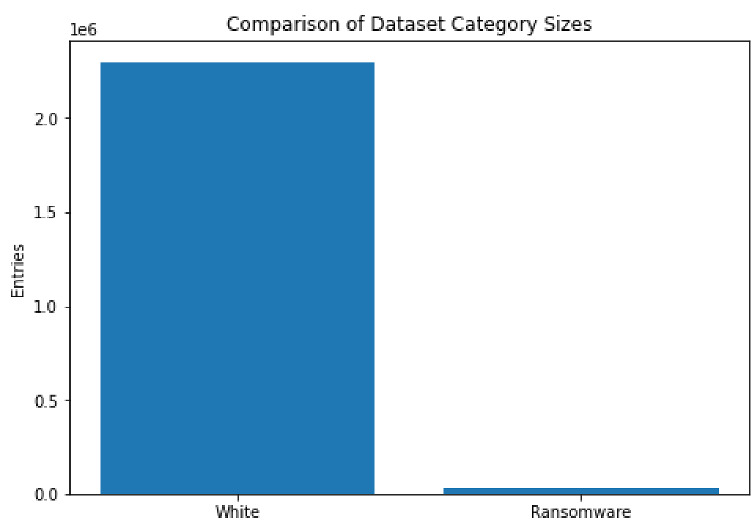
We also noticed that the weights average was slightly higher for non-white labels compared to white labels we found the label “Razy” to be a massive outlier bringing the average weight higher for non-white. However even after finding the average of non-white weights without Razy, the weight average was still significantly higher for ransomware families as seen in the figure below. The median reduces the extremity of the outliers yet we still see that most of the ransomware families have a larger weight.

Since we notice that it is more likely that amount of information on ransomware transaction is larger than non-ransomware transactions and that the ransomware transactions will usually have more merging of multiple addresses than non-ransomware transactions, we hypothesize that those two features would have an impact when trying to identify if a ransomware transaction is ransom or not. This could be due to how tracking down ransomware transactions is harder if it is coming from multiple addresses being merged together hence why there is so much information on it. We would keep these trends in consideration when developing our models.





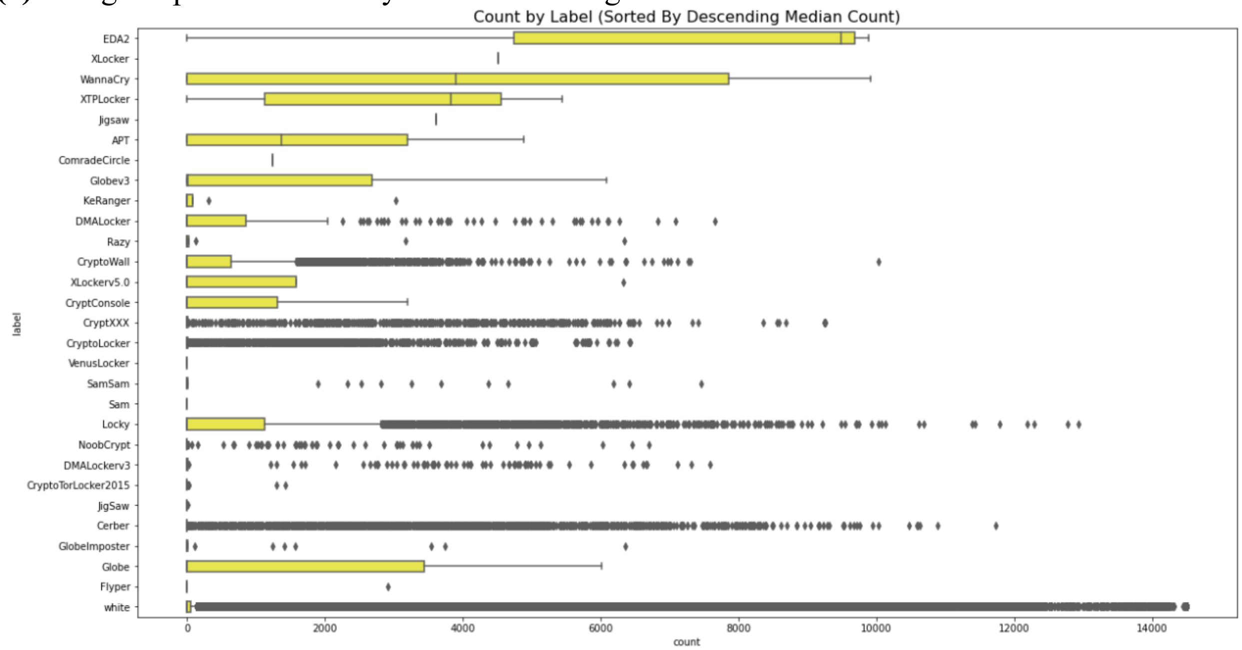
However, the data suffers from a strong imbalance of classes in the dataset, as typical when trying to identify trends among fraudulent transactions. As seen in the figure below, about 98.5% of the dataset describes “white” transactions, while only about 1.5% described features of ransomware transactions.



An imbalance such as this will result in a machine learning algorithm that tends to solely predict “white” for all instances of transactions and receive a high accuracy purely due to the split of classes in the data rather than analyzing trends. Since “white” transactions were usually in the middle of the pack for the mean distribution of features, as well as the overall large dataset size, we decided to solve this imbalance by under-sampling. Thus, we decided to under-sample the majority class, in this case the “white” transactions, so both classes would have equal dataset sizes.

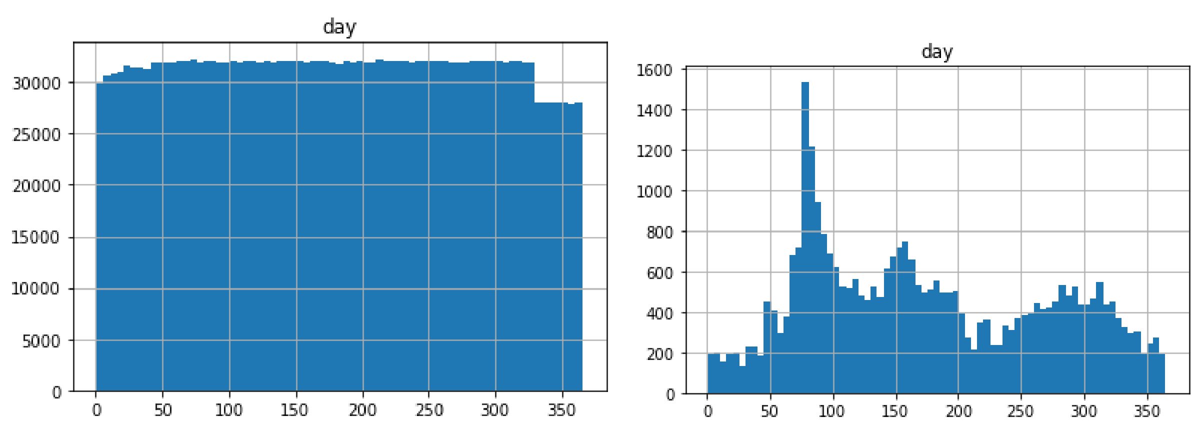
Visualizations for Trends and Patterns

(1) Using boxplot of “Count by Label” to distinguish the labels



We used multiple box-plots for each label in the dataset to find the general trend of the distribution of the median count in descending order. The count is an important variable in determining if a transaction is ransom or not and which ransomware transaction it may fall under. The box-plot is an important EDA tool to see the location of the spread of the first and third quartile where ~50% of the counts would usually lie. When the median is in descending order, it is easier to detect the variation of how the counts are distinguished between each other. We noticed that most of the ransomware labels have lots of outliers that may have caused skews in the mean and median that the previous histograms may not have captured.

(2) Using “day” to distinguish white from non-white labels

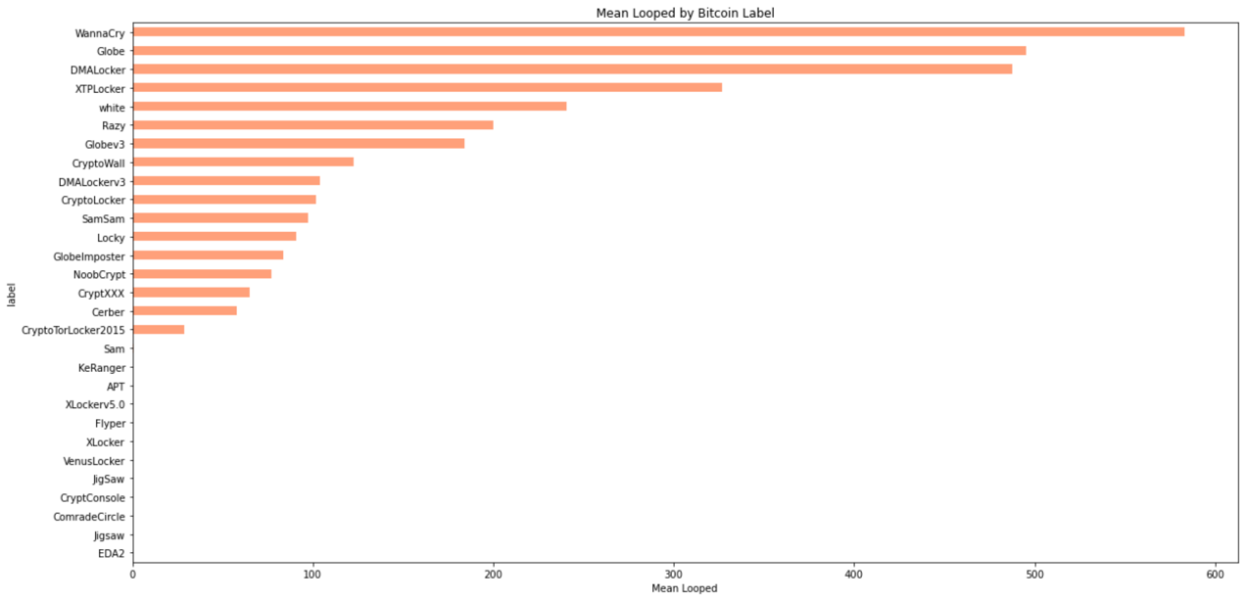


The histogram makes it easier to distinguish when most of the transactions take place to see if there is a significant difference of when a ransom or non-ransom transaction takes place.
The top histogram conveys when all the non-ransomware transactions in the dataset took place, with the number of days on the x-axis for the chronological reading order and the y-axis is the number of transactions that took place that day. We originally believed that the ransomware

transactions would take place towards the far right of the histogram where the holidays usually are and there is more financial traffic. However, the distribution of transactions is steady throughout the year with a short dip at the end of the year.

The bottom histogram is formatted the same as the above histogram however, it is only measuring the ransom transactions. We notice that the ransom transactions distribution is not as consistent as the non-ransom transactions and is more popular during certain periods of the year. This trend helps visualize how the days in which a transaction takes place will help our model predict whether a transaction is ransom or not.

(3) Using “looped” to distinguish the labels



The barchart of “Mean Looped by Bitcoin Label” suggest that labels like “WannaCry”, “Globe”, and “DMALocker” have a significantly higher mean looped than the other labels, while twelve of the labels have a mean looped of zero meaning some ransomware labels have more transactions of splitting coins, moving them and merging to a single address. Such significant differences can play a big role in helping to predict the label.

Analysis

[Q3]

To find the top 3 ransom labels that have the most ransom transactions, we look at all the labels that are not labeled “white” because a “white” label is not known to be ransomware so we can not assume that all white labels are ransomware. Solely just finding the number of times each label is an instance of an overall transaction. However, since each entry of the data of the edge is an edge of a transaction graph so each entry is the result of one or multiple transactions to reach the final receiver. Therefore in descending order, the top 3 ransom labels are Locky, Cerber and CryptoWall as seen in the figure below.

```
In [38]: df_train.loc[(df_train['label'] != 'white')].groupby('label').sum().sort_values('count', ascending = False)
```

Out[38]:

label	Unnamed: 0	year	day	length	weight	count	looped	neighbors	income
Locky	6231904185	10725154	705150	253248	1989.428568	5542529	481585	6822	1.310848e+12
Cerber	8748893390	14882627	1350023	295244	2334.856076	5402451	427317	14840	7.618505e+11
CryptoWall	11619126814	19884772	1401304	472732	7842.422280	4198413	1207754	19718	6.893587e+12
CryptoLocker	8653678681	14940653	1783556	225348	6509.520377	2244009	754929	21180	1.336232e+13
CryptXXX	2294998251	3896928	324597	92270	708.423248	1535356	126131	3865	2.606644e+11
DMALockerv3	343648023	584774	57213	10744	160.709804	277538	30116	332	1.728931e+11
DMALocker	229018152	423300	28653	8416	204.611220	205529	102374	384	1.856083e+11
NoobCrypt	439783796	781788	64377	8186	328.269803	123353	29969	502	8.745836e+10
WannaCry	28227174	48408	3211	2342	15.383665	106545	13999	41	1.473959e+09
SamSam	49301370	90734	9889	1734	26.661929	45609	4377	67	4.674155e+10
Globev3	38153283	56459	5958	1984	12.360396	38764	5147	61	3.387741e+09
Globe	25957933	44356	5692	1068	10.753404	31348	10897	47	1.666855e+09
XTPLocker	9821053	14112	1463	724	2.720264	20620	2286	10	1.909947e+09
EDA2	2321583	6050	486	294	0.368026	19376	0	4	1.115830e+08
GlobeImposter	43700692	72554	8176	1078	19.996960	18007	3018	65	3.703580e+10
APT	11449145	16131	1900	592	4.243335	14439	1	16	2.940237e+09
Razy	13080104	26184	3529	424	561.837218	9683	2597	133	2.966677e+12
XLockerv5.0	5131513	8068	138	148	2.083952	6332	0	5	6.999757e+08
CryptConsole	8325394	14119	282	304	4.153142	5822	0	14	3.182434e+08
XLocker	731000	2017	144	144	0.412207	4511	0	1	1.000000e+08
Jigsaw	913387	2016	120	144	0.542603	3617	0	2	4.200000e+07
KeRanger	5200150	16128	579	358	3.479977	3347	2	8	7.999000e+08
Flyper	5280234	14116	2092	162	3.119608	2916	0	11	3.830065e+08
CryptoTorLocker2015	59947782	94706	7812	536	59.141965	2837	1357	494	3.368139e+10
ComradeCircle	1511448	2016	292	144	0.051214	1241	0	2	2.033200e+08
JigSaw	3272917	8064	832	26	1.662431	10	0	8	2.584881e+08
VenusLocker	5614203	12101	544	198	0.594240	6	0	9	6.000000e+08
Sam	180170	2016	271	6	0.062500	1	1	3	2.900000e+09

[Q4]

Next, to define a model that would be appropriate to classify heist incidents into ransomware families, we decided to use a Decision Tree Classification model.

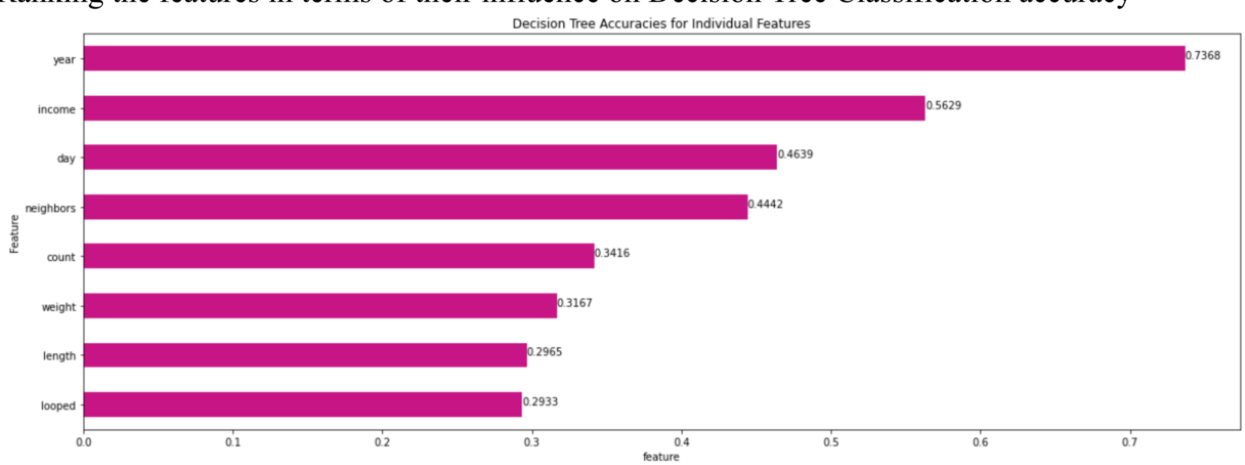
To make this model work, we processed the training data by removing all the rows with label == 'white', then removed the column 'Unnamed: 0', so all the remaining columns (except for 'address') are useful quantitative features. Since the column at index 9 is the 'label', that column reasonably became **y_train**, while the columns at indices 1 to 8 of the training data became **x_train**, and the columns at indices 1 to 8 of the testing data became **x_test**. Column at index 0 ('address') was excluded from the features because 'address' was not a feature we wanted to use.

After processing the data, we put x_train and y_train into a DecisionTreeClassifier and made predictions for x_test. After removing the feature columns, the result looked something like this:

	address	predicted_label
0	16r8CxcVCypUFzvHHZYttyiZtMaGnJn3te	CryptoLocker
1	12EK9jUdG3heM7AF6Abyp38yuNMHN4dcq1	Locky
2	16xUAFderxZwbEp9yuz4FdPnMVxTQntcwN	Cerber
3	1JvUt1UUDey7JY7WYHNTBSUNuhq1Vkbdfd	CryptoLocker
4	138BLKDpeNyKdHnrLT6hZMW119sD4PZJ6D	CryptoWall
...	...	...
583335	123U8HgTRduGkP5A7W99WegVyibLNA7U1D	Locky
583336	1DrPsCAohyjgsgeN377Ntym3Ch1xSRZyYw	Locky
583337	1MKZQMPKfNiC2SyJSqSXZPmaYQVZnWnC7T	CryptoLocker
583338	39fBLaBjEXS66yMfL4sTEjAko8ErpFV7pK	DMALockerv3
583339	19QwMNP5eb2kGH1u2ckHnjcRu5whf2dUxD	CryptoWall

where 'predicted_label' shows which ransomware the DecisionTreeClassifier classified the transaction to be. The issue with the above result is that the model was trained so that it *cannot* classify a transaction as white, which meant all the transactions in the testing data that should be classified as white were classified as ransomware. However, this would not be an issue if the testing data consists of only ransom transactions.

EXTRA NOTES:
Ranking the features in terms of their influence on Decision Tree Classification accuracy



The Bitcoin dataset has eight features that can be used to train a Decision Tree Classification model: year, income, day, neighbors, count, weight, length, and looped.

To find out which feature has the most influence on the ransomware family classification of a transaction, we created a table containing just the feature and the label for each of the eight features, then trained a different Decision Tree Classification model for each table. The resulting list of accuracies is sorted to show a pattern.

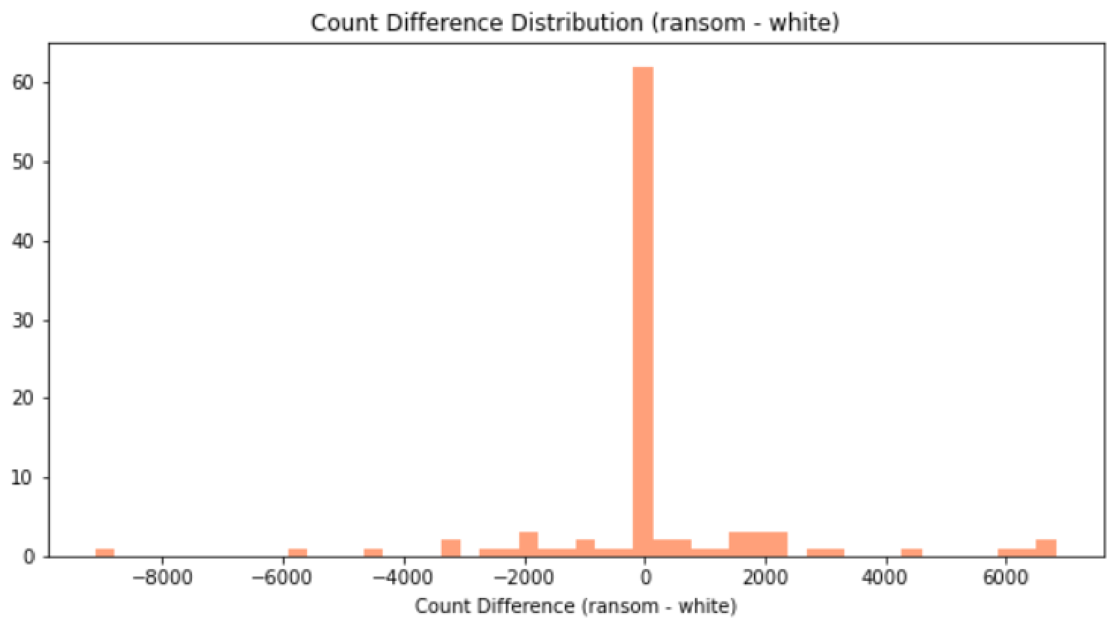
[Q5]

Our model to determine whether a transaction was “white” or ransomware involved renaming the labels to 0 if white, 1 if not. In addition, we also temporarily dropped the address column as we believed this categorical variable would be too complicated for the model to use as a predictive feature. We used the ktrain package, a wrapper for the deep learning library Tensorflow Keras to standard predictive features and predict whether a transaction was white or ransomware.

Proposal

Hypothesis:
A ransom transaction is more likely to have a higher ‘count’ than a white transaction, since a ransom transaction has an incentive to have more starter transactions connected to a ransom address.

Experimental Testing:
Find the percentage of the times a randomly selected ransom transaction has a greater count than a randomly selected white transaction.



Experimental Testing conclusion:

```
percentage_ransom_greater_count_than_white = np.sum(np.array(differences) > 0, axis=0) / len(differences)
percentage_ransom_greater_count_than_white
```

```
0.37
```

About 37% of the differences are positive, meaning that only about 37% of the times a randomly selected ransom transaction has a greater count than a randomly selected white transaction. We reject the null hypothesis because we do not have significant evidence that a ransom transaction is more likely to have a higher ‘count’ than a white transaction.

Conclusion.

After running our model to determine whether transactions are ransom or not, our model predicted that out of the 583,340 transactions in the test data set, 127,699 transactions were considered to be ransom transactions. The three most predicted ransomware families were Cerber, CryptoLocker, and Locky, which isn't too far off the training data frequency. However, some families with low initial frequency in the training data, such as Jigsaw and Sam, were not predicted by our model. This is probably due to low representation in the training data. While having data of equal proportions of class representation would be ideal, real world data is not cleanly distributed. It is important to process data well to result in proper application to analysis methods.