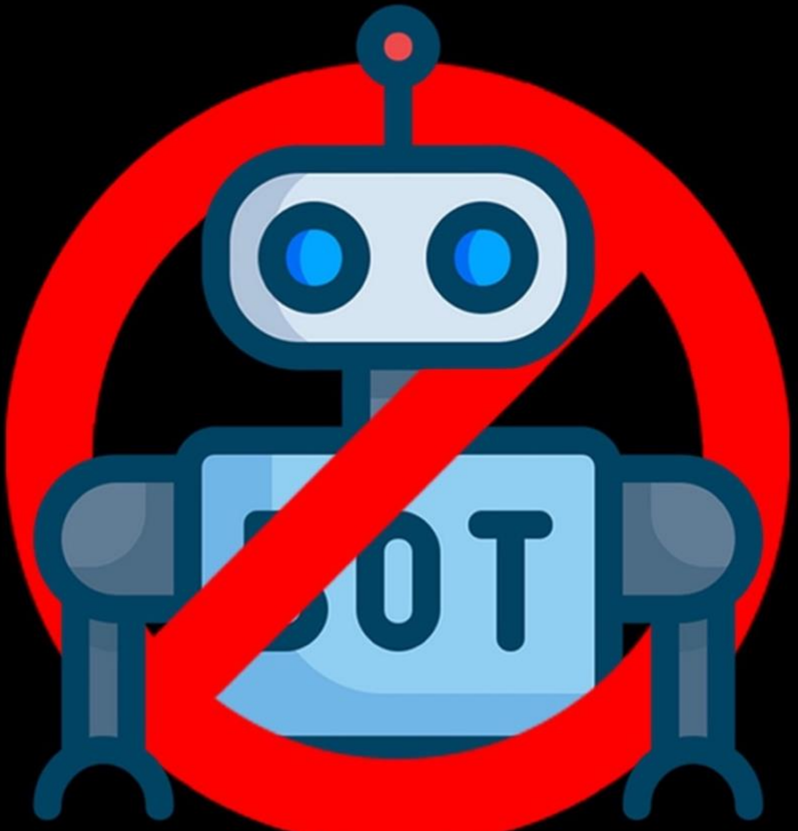
41004 AI/ANALYTICS CAPSTONE PROJECT - AUTUMN 2021

ASSIGNMENT 3: FINAL PROJECT & PRESENTATION



GROUP 35:

ASHLEY NGUYEN (13389465) DENZEL MOK (13229671)

MITCHELL VALENTINUS (13643352) PATRICK SONGCO (10420139)

BOT BUSTERS

PROJECT 24: EXPLAINABLE BOT ACCOUNTS DETECTION IN SOCIAL NETWORKS WITH GRAPH MINING TECHNIQUES

SUPERVISOR: DR. HONGXU CHEN

Table of Contents

Table of Figures

1. Executive Summary

This report was compiled to examine, detect, and visualise the number of bots in social networks with graph mining techniques and to recommend ways on how to accurately detect them.

The research is based on our own scraped data from the Twitter API, supported with Gephi Twitter Streaming Importer Plugin and Botometer API to ease our data mining process. Data pre-processing was also implemented to gain the most influential attributes for our training, validation, and testing models. For our research, we have two main datasets which are imbalanced and balanced datasets, and both were split 60/20/20 into training, validation, and test splits. We approached our datasets with several unsupervised and unsupervised models, including K-Means clustering, Logistic Regression, Gradient Boosting, Decision Tree, Random Forest, Graph Convolutional Network (GCN), and GraphSAGE. The result, explanation, and comparison for each of our approaches to both imbalanced and balanced datasets are stated accordingly.

In our research summary, we found that the Decision Tree model has the highest accuracy with the Test Accuracy score 0.88 and its F1-Score 0.530 for the imbalanced dataset. While on the hand, Gradient Boosting model is better for the balanced dataset as it achieved 0.89 score for its Test Accuracy and 0.880 for its F1-Score. This report also explains our findings through visualizations, such as the combined Bot and Human Network, Cluster Detection, Correlation Matrix, and several attributes' graphs, including Age of Account Distribution, Verification Check, Follower/Following Ratio, clustering and anomaly detection.

Based on our Bot Detection in Social Networks with Graph Mining Techniques report, it is recommended that:

1. Gradient Boosting model has the highest accuracy on bot detection on balanced datasets.
2. K-fold validation method should be implemented when training models on imbalanced or real-world datasets to ensure a higher accuracy in bot detection.
3. Further analysis into bots that are able to mimic human behaviour should be performed as it challenges the process of bot detection.

This report concludes on our encountered difficulties description that challenged our research, such as under sampling due to the imbalanced dataset, explainability, rate limits, account deletions, K-Means Clustering visualization, and decrease in accuracy due to under sampling.

2. Business Problem

The goal of this project is to analyse current progress in social bot detections, visualise it, and study the possible application of graph-based algorithms to the detection procedure.

For our research background, we studied why social media is important for cryptocurrency. We found that Reddit and Twitter are the most popular platforms for cryptocurrency. As of April 2021, the subreddit Bitcoin has 2.7 million members and Cryptocurrency has 2.1 million. On the other hand, Twitter, the platform that we chose for our research project, has 210K tweets daily with the #Bitcoin (BitInfoCharts 2021). Twitter also becomes the industry's newsfeed, where the important debates that define the industry happens, such as Bitcoin vs Gold, where the cryptocurrency reputation is made, and where the industry leaders are, like the top traders, journalists, researchers, and even Elon Musk. Twitter also recognizes that tweets about Bitcoin have been surging throughout the years simultaneously with the rise of Bitcoin price (Lielacher & Pickering 2020), the graph is presented in Figure 1.

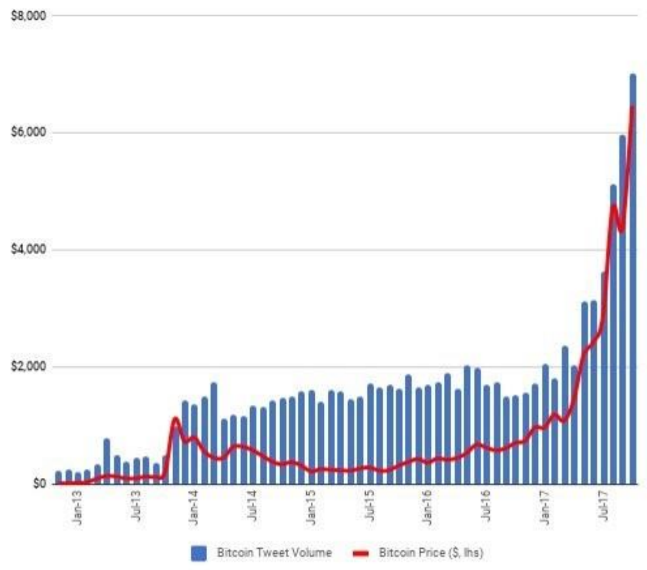


Figure 1: Bitcoin Tweet & Price Graph

As we know how important social medias are to cryptocurrency, bots can be a real threat to the ecosystem. Bots themselves are fake or inauthentic accounts that are made to provide misinformation, increasing fake engagements, and even making scams easier. In our research, we found that there are 2 types of bots, the regular one that only serves one purpose, such as retweeting, and the fake accounts that are able to mimic the actions of real humans in social media, they can have locations enabled, post random pictures, and even tweet just like humans. This makes it harder to detect than the regular bot accounts.

Bots have been around for a long time, in 2018. Facebook disclosed that they have 1.3 billion fake accounts and their monthly users, 5% of them are fake. For Twitter, they estimated that 40% of their users are not real. These high numbers surely are not healthy for the social ecosystem (Lielacher & Pickering 2020). We found some bitcoin examples that experienced sudden huge upticks on user interests, Satoshi Vision Bitcoin ($BSV) and Ripple Bitcoin ($XRP). This happened through bots that spam contents about them repeatedly to gain real human's interest (Lielacher & Pickering 2020).

To approach this problem, we extracted several interesting user features from our own scraped Twitter API dataset such as ‘age of account’, ‘number of followers’, and many else, as we believe these kinds of features can help us navigate and detect what kind of pattern bots usually have. We used the Gephi Twitter Streaming Importer Plugin to find the edges between users, and we also used the Botometer API as the bot labeller for each user. We believe these kinds of applications and programs can help our research thoroughly and provide a strong foundation for further research.

3. Data Exploration

3.1. Data Gathering

Due to the relative novelty of graph-based approaches in detecting bots, we found it difficult to find existing datasets that contained both user information and network information (edge connections). As such, we utilized publicly available APIs such as the Gephi Twitter Streaming Importer Plugin, Twitter API and Botometer API to create our own dataset for analysis. The APIs, data mining process, and collected dataset will be discussed further in the following sections.

3.2. Data Mining Process

The data mining process involved three steps. First, we needed to scrape both user and network information using the Gephi Twitter Streaming Importer Plugin. Next, we extracted more information for each user using the Twitter API. Finally, we used the Botometer API to determine the likelihood that a user was a bot or human.

3.2.1. Gephi Twitter Streaming Importer Plugin

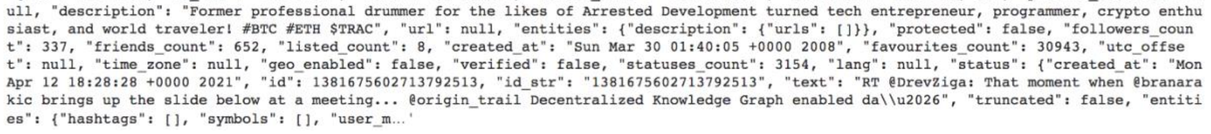
The Twitter Steaming Importer Plugin on Gephi was used to scrape 5001 Twitter users with 14573 connections between them. These connections represent hashtags present in Tweets between users. The hashtags used were based on the cryptocurrency topic. These included:

#crypto #cryptocurrency #blockchain #bitcoin #btc #ethereum #eth #dogecoin #coinbase.

The output of this step was two tables. A Nodes Table containing all the users and an Edges Table containing all connections between users.

3.2.2. Twitter API

The "get_user" endpoint was used on the Twitter API to extract more meaningful information for each user. This endpoint takes input in the form of a Twitter screen name and outputs a JSON file (Figure 2).



```
'{"id": 14254757, "id_str": "14254757", "name": "CK Nomad", "screen_name": "DrumKitt87", "location": "Atlanta, Ga", "profile_location": n
ull, "description": "Former professional drummer for the likes of Arrested Development turned tech entrepreneur, programmer, crypto enthu
siast, and world traveler! #BTC #ETH $TRAC", "url": null, "entities": {"description": {"urls": []}}, "protected": false, "followers_coun
t": 337, "friends_count": 652, "listed_count": 8, "created_at": "Sun Mar 30 01:40:05 +0000 2008", "favourites_count": 30943, "utc_offse
t": null, "time_zone": null, "geo_enabled": false, "verified": false, "statuses_count": 3154, "lang": null, "status": {"created_at": "Mon
Apr 12 18:28:28 +0000 2021", "id": 1381675602713792513, "id_str": "1381675602713792513", "text": "RT @DrevZiga: That moment when @branara
kic brings up the slide below at a meeting... @origin_trail Decentralized Knowledge Graph enabled da\\u2026", "truncated": false, "entiti
es": {"hashtags": [], "symbols": [], "user_m...'
```

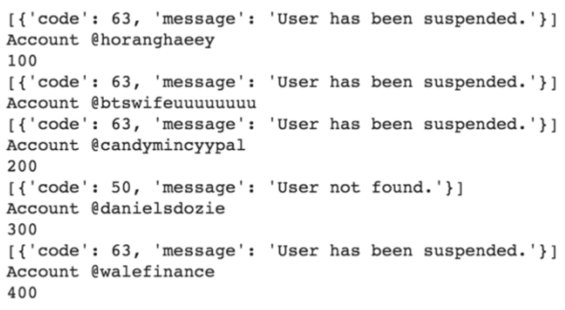
Figure 2: Example user JSON file

Specific features in the JSON file were extracted for the purpose of this analysis (Figure 3). These features were selected based on the Twitter policy on spam profile detection.

Extracted/Derived User Features	Assumptions
Age of Account (Years)	Bots generally have new accounts
Number of Followers	Bots have low number of followers
Number of Followings	Bots tend to follow a large number of users
Follower/Followings Ratio	This ratio is low for Bots
URL in User Description	Bots typically share links to malicious sites
Verified Account	Bot accounts aren't verified
Number of Tweets (including Retweets)	Bots post tweets more frequently
Number of Tweets User has Liked	Bots like tweets more frequently
Number of Public Lists User is a member of	Bots are members of many public lists

Figure 3: Selected Features Table

During the data collection process in this step, some users were either deleted or suspended by Twitter since the initial collection of users in step one (Figure 4). Therefore, these users were deleted in the data pre-processing phase (Section 3.3)



```
[{'code': 63, 'message': 'User has been suspended.'}]
Account @horanghaeey
100
[{'code': 63, 'message': 'User has been suspended.'}]
Account @btswifeuuuuuuuu
[{'code': 63, 'message': 'User has been suspended.'}]
Account @candymincyypal
200
[{'code': 50, 'message': 'User not found.'}]
Account @danielsdozie
300
[{'code': 63, 'message': 'User has been suspended.'}]
Account @walefinance
400
```

Figure 4: Sample error codes using Twitter API

3.2.3. Botometer API

The final step was to establish a ground truth label for each user. This was accomplished by thresholding the score given by the Botometer API. The Botometer API is a publicly available web application developed by Yang *et al*. (2019) that assigns a score between 0-5 of bot-like behaviour to Twitter users; 5 being a high likelihood that the account is a bot. As such, we arbitrarily chose to label all users with a score of 4 and above as a bot, human otherwise.

3.3. Data Pre-Processing

The raw Nodes Table and Edges Table were cleaned and pre-processed using Python in a Google Colab notebook.

3.3.1. Nodes Table

This table contains the list of all users and their features. The various alterations are outlined below.

- 77 users were deleted because additional features in step two were unable to be extracted.
- Attributes in Figure 3 were added as columns for extra features.
- During the collection of Botometer scores in this step three, some users were assigned a blank score. For the purpose of this analysis, we have assumed that these users were either deleted or suspended by Twitter for being a bot/spam account. These 89 users were given a score of 6. We chose a score of 6 instead of 5 so that we can omit them from analysis if needed.
- Human/Bot Label column was added. This was calculated by thresholding the Botometer score. A score of 4 or over was labelled 1, 0 otherwise.
- Screen names for each user were converted to lower case for uniformity.

3.3.2. Edges Table

This table contains the list of all users-user interactions and hashtag present. The various alterations are outlined below.

- 323 rows were deleted as they contained interactions between users that were deleted in the Nodes Table.
- Unwanted columns were removed.
- The @ and # symbols were removed from the beginning of screen names and hashtags respectively.
- Screen names for each user were converted to lower case for uniformity.

3.4. Imbalanced Dataset

The final dataset we obtained comprised of 696 bot labels (14%) and 4228 human labels (86%). This imbalanced dataset proved to decrease performance during the modelling phase due to the class imbalance. Traditional machine learning techniques such as Logistic Regression, Random Forest and Gradient Boosting allows users to up-sample the minority class by randomly adding additional training examples chosen from the minority class. Unfortunately, for the graph-based techniques, adding additional minority class (bot) training nodes to the network is infeasible as this will affect the underlying network structure. Therefore, we created an additional balanced dataset by down sampling the majority class (humans) so that we have a 50/50 ratio of bots to humans. The downside of this is a loss of training information. It should be noted that in the graph-based models, we will only be masking the unused training example so we can keep the underlying network structure.

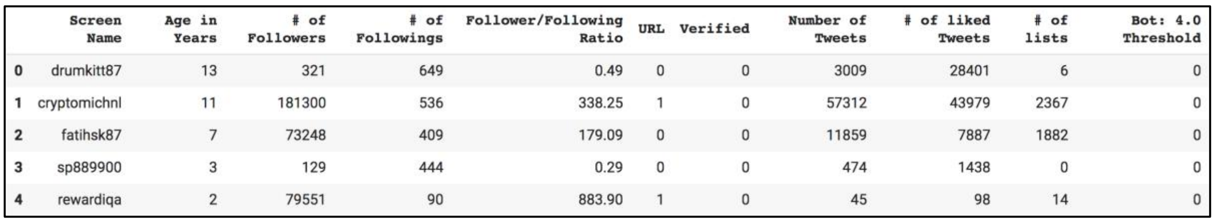
3.5. Final Cleaned and Pre-Processed Datasets

The final datasets comprise of an imbalanced and balanced dataset. These were split (60/20/20) into training, validation, and test splits. We used the same data splits for every model so that we can accurately compare results across each model.

The statistics of the final datasets we will be using for the modelling stage is shown in Figure 5 below. Figure 6 and 7 display the first 5 rows of the Nodes and Edges Tables respectively.

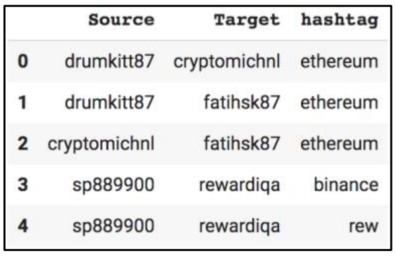
	Raw Data	Pre-Processed Imbalanced Data	Pre-Processed Balanced Data
Number of Users (Nodes)	5001	4924	1392
Number of Connections (Edges)	14573	14250	14250
Number of Attributes	3	11	11
Number of Human Labels	-	4228	696
Number of Bot Labels	-	696	696

Figure 5: Dataset Statistics Table



	Screen Name	Age in Years	# of Followers	# of Followings	Follower/Following Ratio	URL	Verified	Number of Tweets	# of liked Tweets	# of lists	Bot: 4.0 Threshold
0	drumkitt87	13	321	649	0.49	0	0	3009	28401	6	0
1	cryptomichnl	11	181300	536	338.25	1	0	57312	43979	2367	0
2	fatihsk87	7	73248	409	179.09	0	0	11859	7887	1882	0
3	sp889900	3	129	444	0.29	0	0	474	1438	0	0
4	rewardiqa	2	79551	90	883.90	1	0	45	98	14	0

Figure 6: Nodes Dataset Head



	Source	Target	hashtag
0	drumkitt87	cryptomichnl	ethereum
1	drumkitt87	fatihsk87	ethereum
2	cryptomichnl	fatihsk87	ethereum
3	sp889900	rewardiqa	binance
4	sp889900	rewardiqa	rew

Figure 7: Edges Dataset Head

4. Modelling

In this section we discuss the implementation and rational of the various models we explore. The goal is to compare the methods to better understand their viability in detecting social bots. The models we explore can be broadly categorised into traditional machine learning methods and graph-based machine learning methods. The main difference between these categories is in the input dataset. Traditional machine learning methods take input in the form of labelled user features and solve the problem of bot detection in a supervised or unsupervised manner. On the other hand, the graph-based machine learning methods take both the labelled user features, as well as the social connection or network between these users. The traditional machine learning methods we chose were Logistic Regression, Gradient Boosting, Decision Tree and Random Forest. Whereas the graph-based methods were Graph Convolutional Network and GraphSAGE. Additionally, experiments were conducted with unsupervised methods to determine its viability for bot detection.

4.1. Logistic Regression

Originated from a population growth model in the 19th century (Cramer, 2002), the modern basic form of Logistic Regression analysis statistically models the classification possibilities with binary dependent variables. Due to the nature of the model, variables used in Logistic Regression comes with multiple selection criteria, including but not limited to their relevance as well as independence of errors, their linearity in the logit for continuous variables, their independence from one another (i.e. absence of multi-collinearity/high correlation) and their lack of influential outliers (Stoltzfus, 2011). When combined with the results found in the dataset's correlation matrix, these assumptions prove the model suitable to serve as the baseline model in this project.

The model used in this project was implemented from the Linear Model package available in the sklearn Python library. It is then customised in several model parameters in order to be more suitable with the data shape, with the maximum number of iterations taken for the solvers to converge set at 500, and the inverse of regularisation strength set at 0.001 for stronger regularisation. This Logistic Regression model uses the SAGA solver, which is an extension of the Stochastic Average Gradient descent that allows for L1 regularisation, giving the model faster training speed and exceptional suitability for dataset with high dimensionality. Then it is validated using Cross-Validation score and tested with the Confusion Matrix and Classification Report.

The model was trained, validated and tested on different datasets taken from both the imbalanced and balanced datasets to improve the quality of the analysis. Results and findings of the model are summarised in Section 5.

4.2. Unsupervised Methods

Unsupervised learning can also be applied in our problem to investigate whether Bots may be represented by any patterns or outliers. The process involves extracting insights and patterns from the data without any given labels, then comparing the points obtained with the Bot labels. The two main methods we have experimented with is clustering and anomaly detection. Clustering involves discovering any natural groupings in the data, whereas anomaly detection is the process of finding outliers that differ from the majority of the data.

In order to proceed with this problem, we will have to assume that the outliers detected and clusters formed are in fact representative of Bot accounts, however this is very unlikely to be the case as outliers are simply points that are different, with clusters representing the formation of patterns. Nevertheless, by performing this experiment, we can test our hypothesis and determine whether there is any value to using outliers and clusters as a feature. Our experiments involved KMeans clustering, followed by anomaly detection methods with Isolation Forest and Local Outlier Factor.

The first person to use the KMeans algorithm for clustering data was Forgy in 1965. KMeans clustering works by grouping the data samples into fixed (k) number of clusters. The data points are assigned to the closest centroid, which is a point in the middle of the cluster based on its distance to the point. The centroid is initially formed through random initialization, selecting a random point and computing the middle point of each cluster iteratively, until it stabilizes, with the final points used for classification.

With Local Outlier Factor (Breunig *et al*. 2000), the algorithm computes the local density of a given number of data points based on its neighbours. It considers outliers as samples with substantially lower density than that of its neighbours.

Lastly, Isolation Forest (Liu *et al*. 2008) works on the same principle as decision trees, but through isolating anomalies. To isolate a data point, the algorithm recursively generates partitions by randomly selecting a feature from a given set of features and generating a split value from the maximum and minimum values of the feature.

4.3. Gradient Boosting

Gradient Boosting was introduced in the 1990s through Freund and Schapire's work on weighted iterative classification (Schapire, 1990 & Freund, 1995), then later developed through by Breiman and Friedman's work in the early 2000s (Breiman, 1999 & Friedman, 2001). Gradient Boosting algorithm sequentially constructs a linear combination of trees (Biau, Cadre & Rouviere, 2019), where mistakes and errors of the previous tree will be learned and used to adjust the following tree in line. In other word, each new tree is built based on previous trees, which proves to work better than bagging models where trees operate in isolation to one another (Wade & Glynn, 2020).

The model used in this project was implemented from the Ensemble package available in the sklearn Python library. The function GradiendBoostingClassifier is also customised to fit the dataset and the purpose of this, with the learning rate set at 0.5, number of boosting stages set at 10 for trade-off and maximum tree depth of 25. In this model we decided to set the subsample fraction at 0.8 to perform Stochastic Gradient Boosting reduce the correlation among the Gradient Boosting trees generated.

Similar to the previous models, this model was trained, validated and tested on both the imbalanced and balanced datasets using Confusion Matrix, Classification Report and Cross-Validation. Results and findings of the model can be found in Section 5.

4.4. Decision Tree

First introduced in 1960s, Decision Trees are one of the most effective methods for data mining; a commonly used data mining method for establishing classification systems based on multiple covariates or for developing prediction algorithms for a target variable (Song & Lu, 2015). Decision Tree works by classifying a population into branch-like segments that will construct an inverted tree with a root node, internal nodes, and leaf nodes. Both discrete and continuous variables can be used either as target variables or independent variables, and if the dataset is large enough, the data can be divided into training and validation datasets.

Several major advantages of Decision Tree method are as below:

- Simplifies complex relationships between the input and target variables by dividing the original input variables into significant subgroups.
- Easy to understand and interpret.
- Non-parametric approach without distributional assumptions.
- Easy to handle missing values instead of removing them.
- Easy to handle heavy skewed data instead of transforming them.
- Robust to outliers.

But Decision Tree method also has its limitations, which it is often to be overfitting or underfitting, particularly when using a small data set. This kind of problem could lead to limit the generalizability and robustness of the resultant models. Decision Tree could also unintentionally select the strongly correlated input variables to improve the model statistics even though it is not related to the desired outcome of interest.

4.5. Random Forest

Initially introduced by Ho in 1995, Random Forest set out to improve and extend on the Decision Tree algorithm. By constructing a multitude or commonly known as an ensemble of Decision Trees, Random Forests are designed to outperform their predecessor by correcting their tendency to overfit on training data. The model makes use of the idea of bagging, where a combination of learning models is combined to get a more accurate and stable prediction. Due to the high stability and consistency of the performance of Random Forest with little configuration, it has been included in our analysis as an initial comparison.

The implementation of the model was performed using the scikit-learn library in Python. Using random search with 3-fold cross validation, the model was lightly tuned to find more optimal hyperparameters. Training is then performed on both the imbalanced and balanced dataset with evaluation using accuracy and f1-score as the metrics.

The result of the tuning on the imbalanced dataset had a max tree depth of 40, min samples per leaf of 4, min samples to split of 5 and number of trees at 1200. In terms of training on the balanced dataset, the tuning had decreased the accuracy, hence the model chosen for evaluation contained default parameters provided by scikit-learn. It consists of 100 trees, with a min samples per leaf of 1 and min samples to split of 2. Refer to Section 5 for the results and findings of the model.

4.6. Graph Convolutional Network (GCN)

The Graph Convolution Network (GCN), first proposed by Kipf & Welling (2016), considers the network structure of the social network by using a message passing algorithm to learn user embeddings. These user embeddings are then used as training input for the model to solve the binary node classification problem. We chose this method to detect social bots because we hypothesise that including the extra network information can improve the accuracy of the model.

The model was coded in Python and PyTorch using a library for deep learning on graphs called Deep Graph Library (Wang *et al*., 2019). A neural network consisting of two GCN layers was used with an input feature size of 9, hidden feature/dimension size of 100 and an output feature size of 2. Adam Optimizer was used with a learning rate of 0.001. Cross Entropy loss was chosen, and the model was trained over 3000 epochs. The model was evaluated using the accuracy, F1-score, precision, recall and loss.

The model was trained on both the imbalanced and balanced datasets to improve the quality of the analysis. Training, validation and test masks were used to split the graph into training, validation and test nodes. Results and findings of the model are summarised in Section 5.

4.7. GraphSAGE

The GraphSAGE model, first proposed by Hamilton *et al*. (2017), improves on GCN by proposing an inductive framework that learns a function that generates embeddings by sampling and aggregating features from a node's neighbourhood. Similar to GCN, we chose this method to detect social bots because we hypothesise that including the extra network information can improve the accuracy of the model. Additionally, the ability for the message passing algorithm to incorporate the features of the node in the hidden layers and generalize to completely unseen graphs has shown to outperform models such as GCN in the literature.

Deep Graph Library was also used to train and test the model. A neural network consisting of two GraphSAGE layers was used with an input feature size of 9, hidden feature/dimension size of 100 and an output feature size of 2. Adam Optimizer was used with a learning rate of 0.001. Cross Entropy loss was chosen, and the model was trained over 3000 epochs. The model was evaluated using the accuracy, F1-score, precision, recall and loss.

Similar to GCN, the model was trained on both the imbalanced and balanced datasets. Training, validation and test masks were used to split the graph into training, validation and test nodes. Results and findings of the model are summarised in Section 5.

5. Results

5.1. Logistic Regression Results

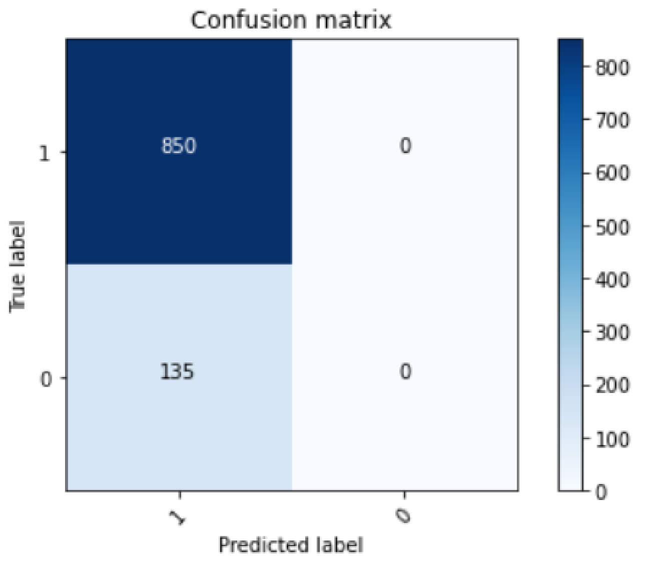


Figure 8: Logistic Regression imbalanced confusion matrix

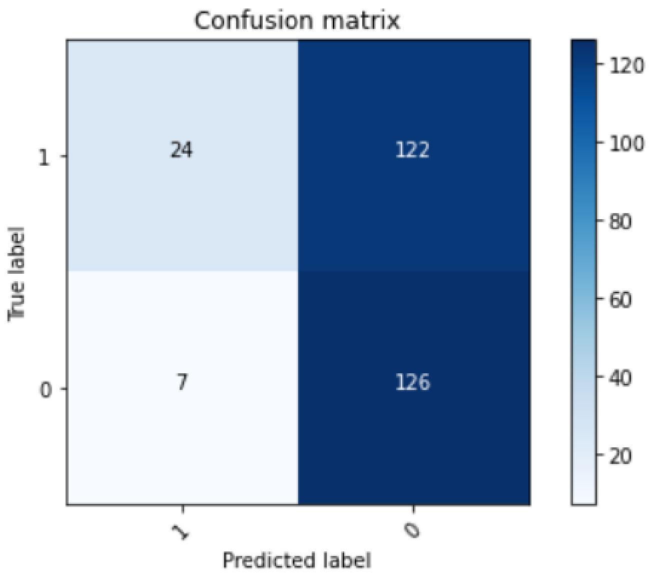


Figure 9: Logistic Regression balanced confusion matrix

5.2. Gradient Boosting Results

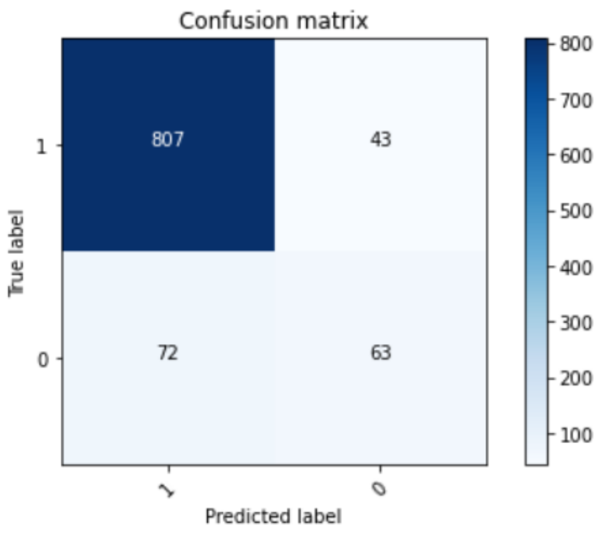


Figure 10: Gradient Boosting imbalanced confusion matrix

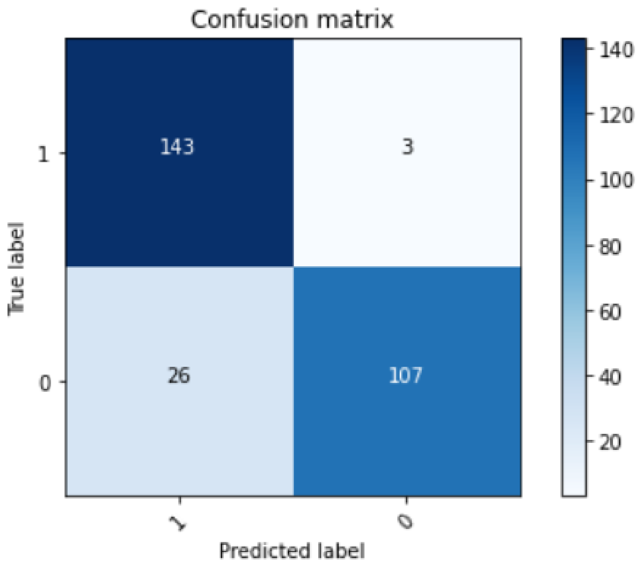


Figure 11: Gradient Boosting balanced confusion matrix

5.3. Decision Tree Results

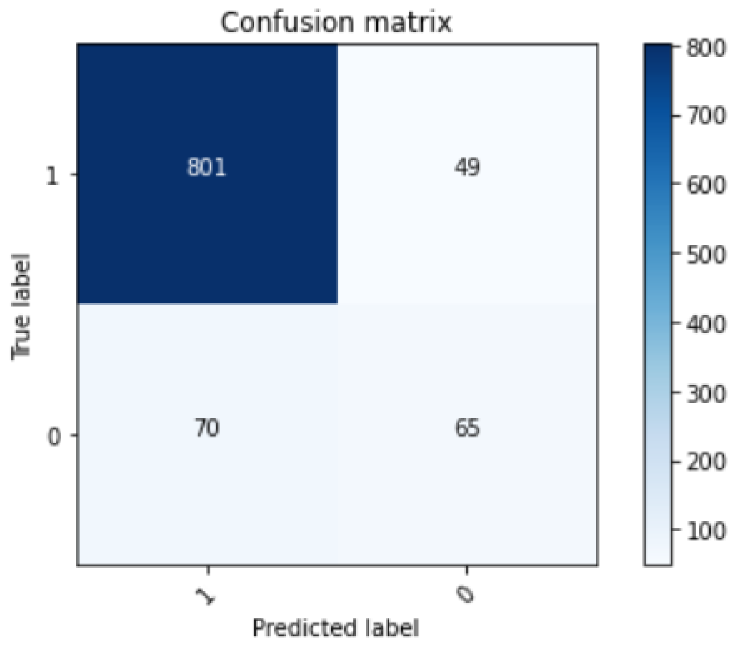


Figure 12: Decision tree imbalanced data confusion matrix

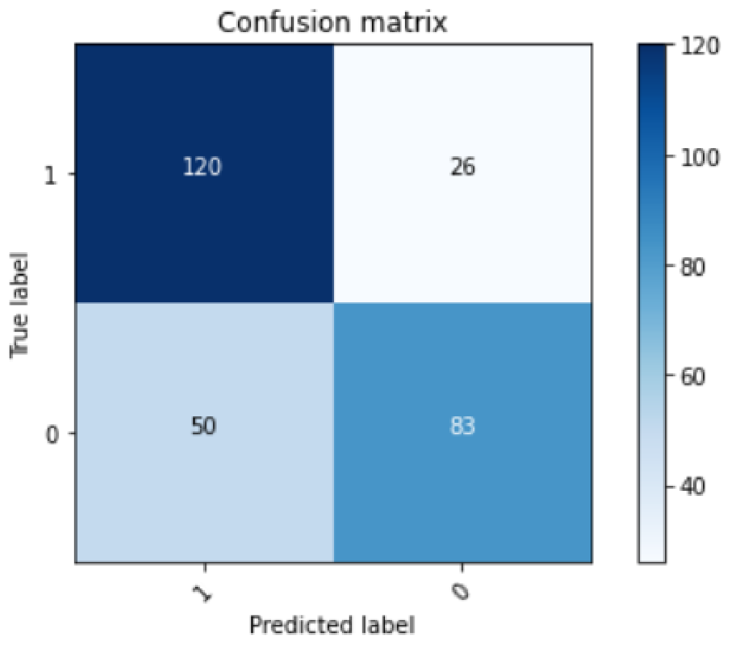


Figure 13: Decision Tree balanced data confusion matrix

5.4. Random Forest Results

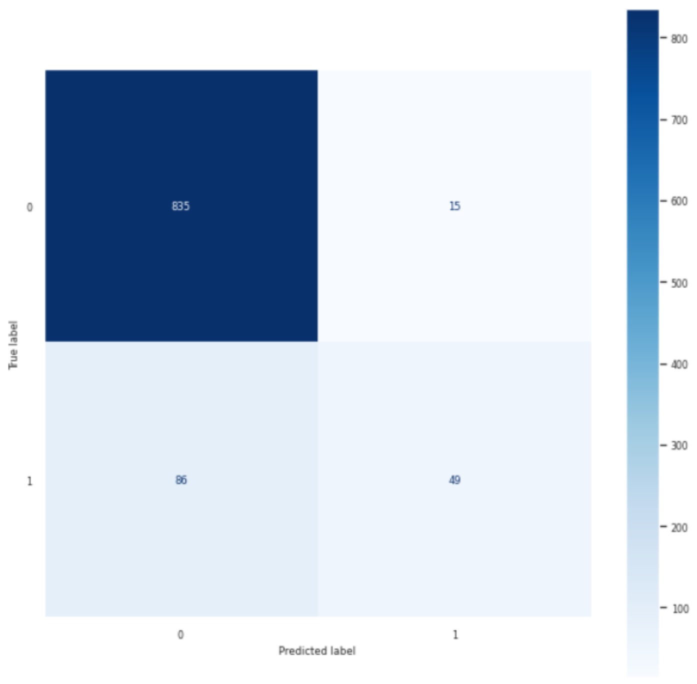


Figure 14: Random Forest imbalanced confusion matrix

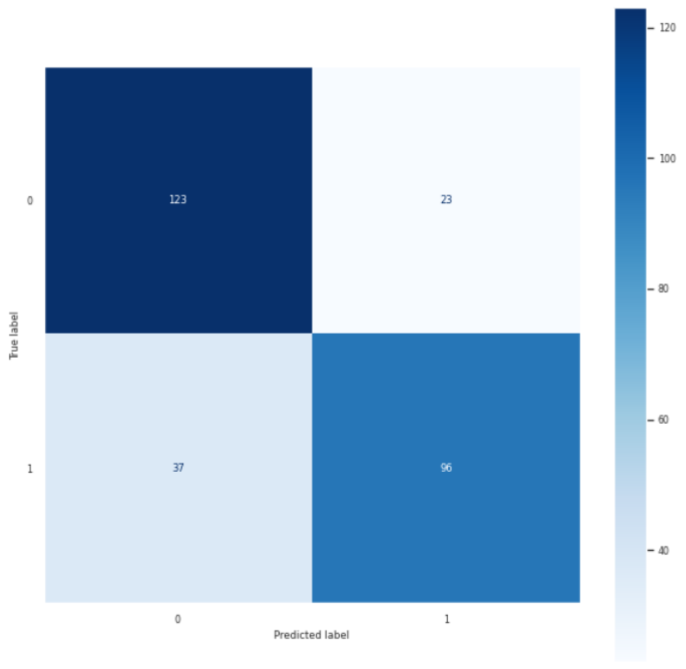


Figure 15: Random Forest balanced confusion matrix

5.5. GCN Results

5.5.1. Imbalanced Dataset

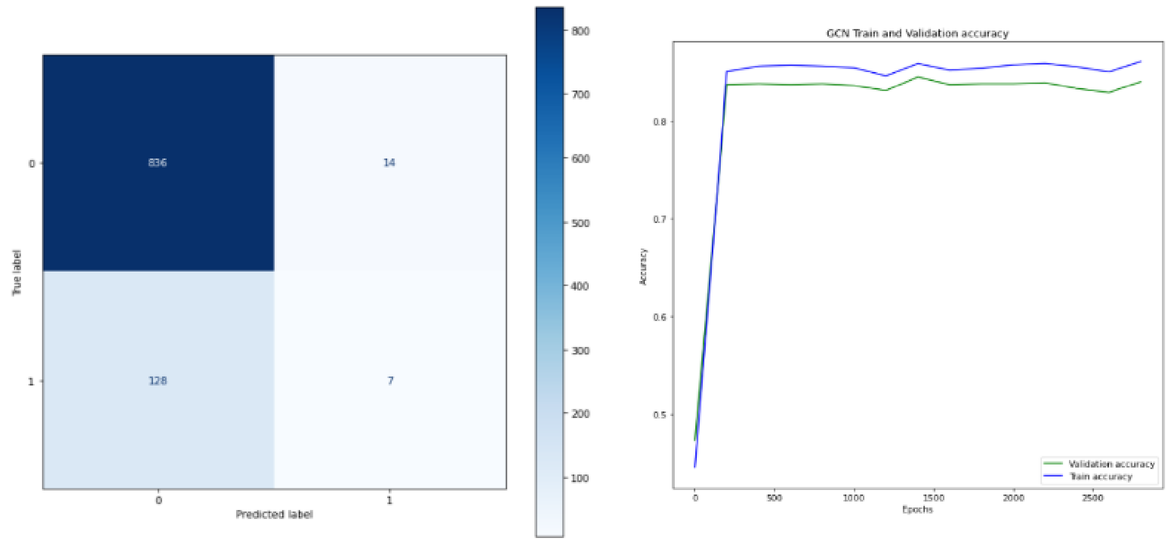


Figure 16: GCN imbalanced confusion matrix (left) and GCN imbalanced train and validation accuracy (right)

5.5.2. Balanced Dataset

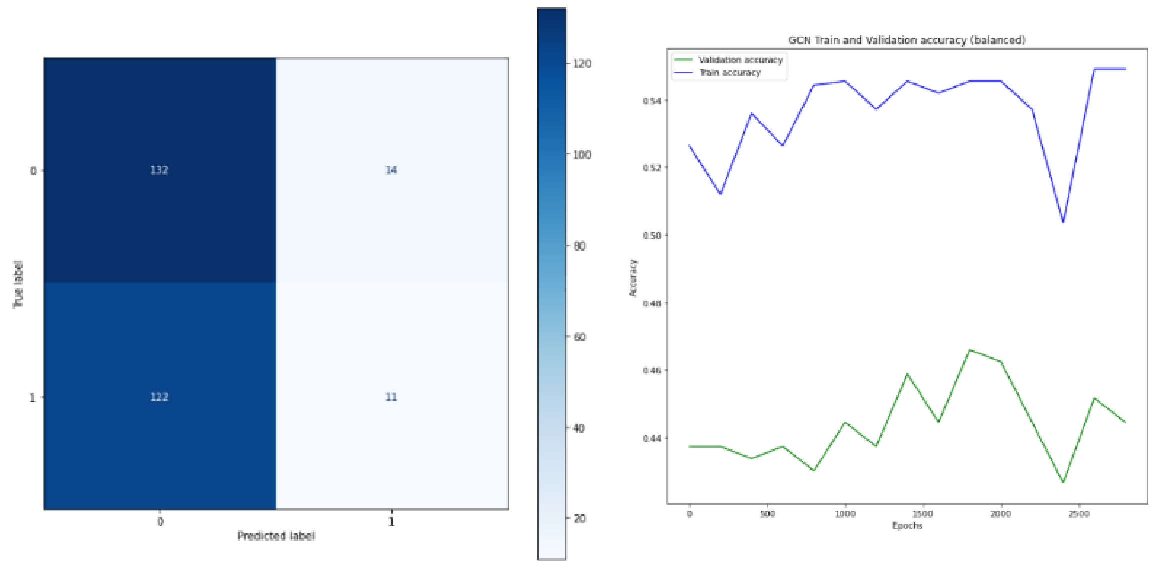


Figure 17: GCN balanced confusion matrix (left) and GCN balanced train and validation accuracy (right)

5.6. GraphSAGE Results

5.6.1. Imbalanced Dataset

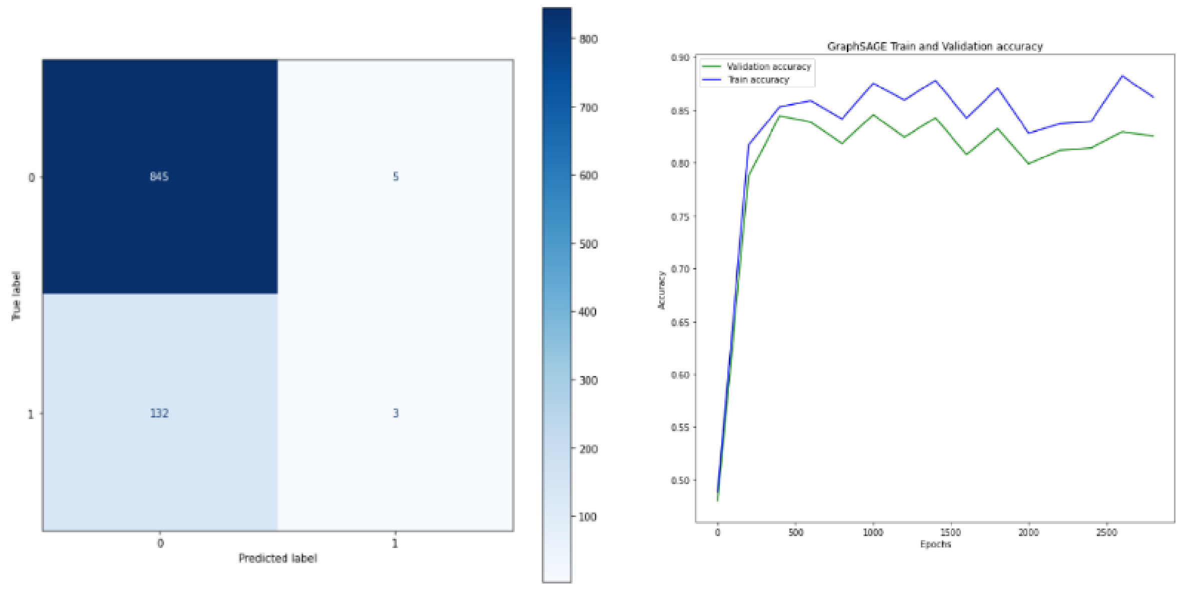


Figure 18: GraphSAGE imbalanced confusion matrix (left) and GraphSAGE imbalanced train and validation accuracy (right)

5.6.2. Balanced Dataset

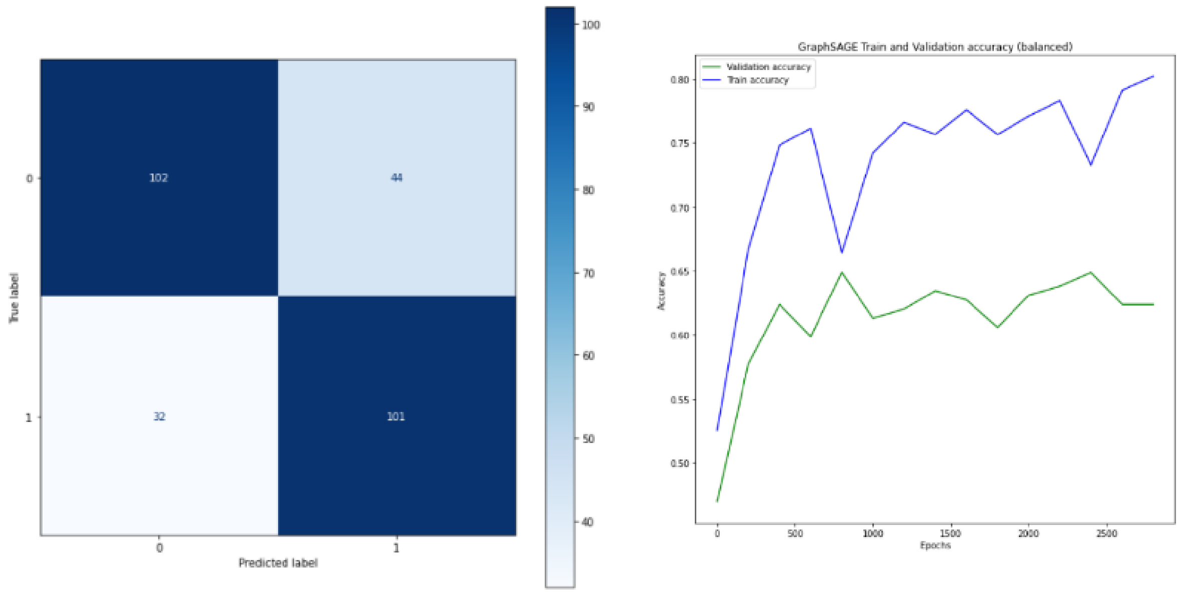


Figure 19: GraphSAGE balanced confusion matrix (left) and GraphSAGE balanced train and validation accuracy (right)

5.7. Results Summary

The results of the modelling are summarised in Section 5.7.1 and 5.7.2. We found that the Decision Tree model had the highest test accuracy and F1-score for the imbalanced dataset and the Gradient Boosting model has the highest test accuracy and F1-score for the balanced dataset. Therefore, in both instances, we see that the traditional machine learning methods outperform the graph-based machine learning methods. Additionally, we found that training the models with the balanced dataset significantly increases the F1-score. This suggests that training a model on a large class disparity will significantly increase the amount of false positives/false negatives.

5.7.1. Imbalanced Dataset Summary

Model	Test Accuracy	F1-Score
Logistic Regression	0.86	0.000
Gradient Boosting	0.87	0.490
Decision Tree	**0.88**	**0.530**
Random Forest	0.90	0.492
GCN	0.86	0.090
GraphSAGE	0.86	0.420

Figure 20: Imbalanced Dataset Summary Table

5.7.2. Balanced Dataset Summary

Model	Test Accuracy	F1-Score
Logistic Regression	0.54	0.660
Gradient Boosting	**0.89**	**0.880**
Decision Tree	0.73	0.690
Random Forest	0.78	0.762
GCN	0.51	0.139
GraphSAGE	0.73	0.727

Figure 21: Balanced Dataset Summary Table

5.7.3. Unsupervised Modelling Summary

Model	Accuracy	F1-Score
KMeans Clustering	0.86	0.003
Isolation Forest	0.80	0.094
Local Outlier Factor	0.84	0.083

Figure 22: Unsupervised Modelling Summary Table

6. Findings and Explainability

In this section, we examine the findings of our data mining and modelling process. With explainability in mind, from our findings, we aim to provide an analysis on the user features that contribute to the likeliness a Twitter account is a bot. Additionally, we provide various visualisations of our collected dataset to obtain a qualitative understanding of the social bot distribution in the network.

6.1. Network Visualisation

First, we visualise the entire dataset on Tableau with the bot accounts coloured in red and the human accounts coloured in blue. Figure 23 depicts the network with only the bots visible, Figure 24 depicts the network with only human nodes visible and Figure 25 contains all nodes in the network. We find that bots are generally dispersed all around the network. The bot clusters tend to aggregate around the human clusters suggesting that the bot accounts try to imitate human behaviour to evade detection.

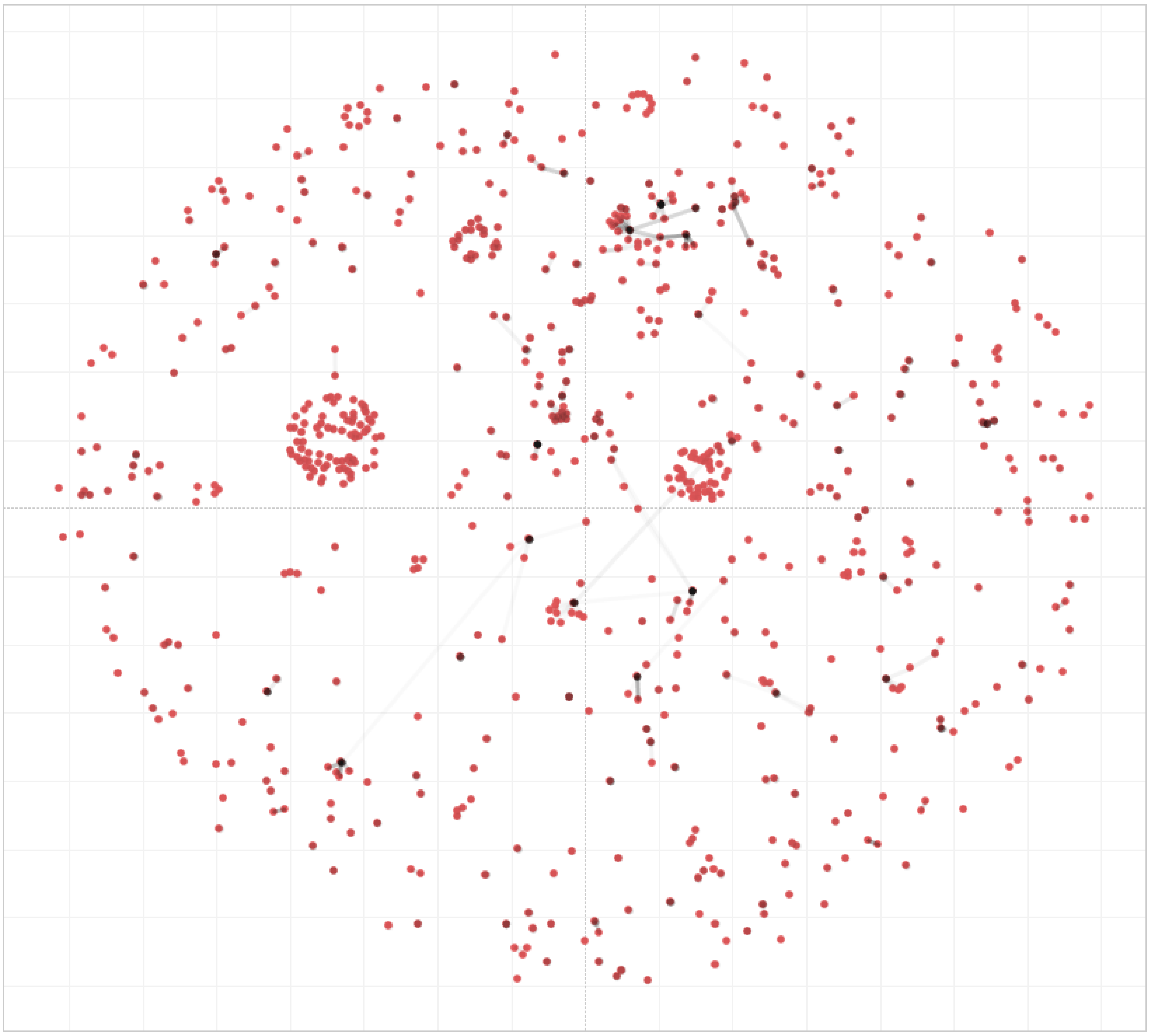


Figure 23: Bot Network Visualization

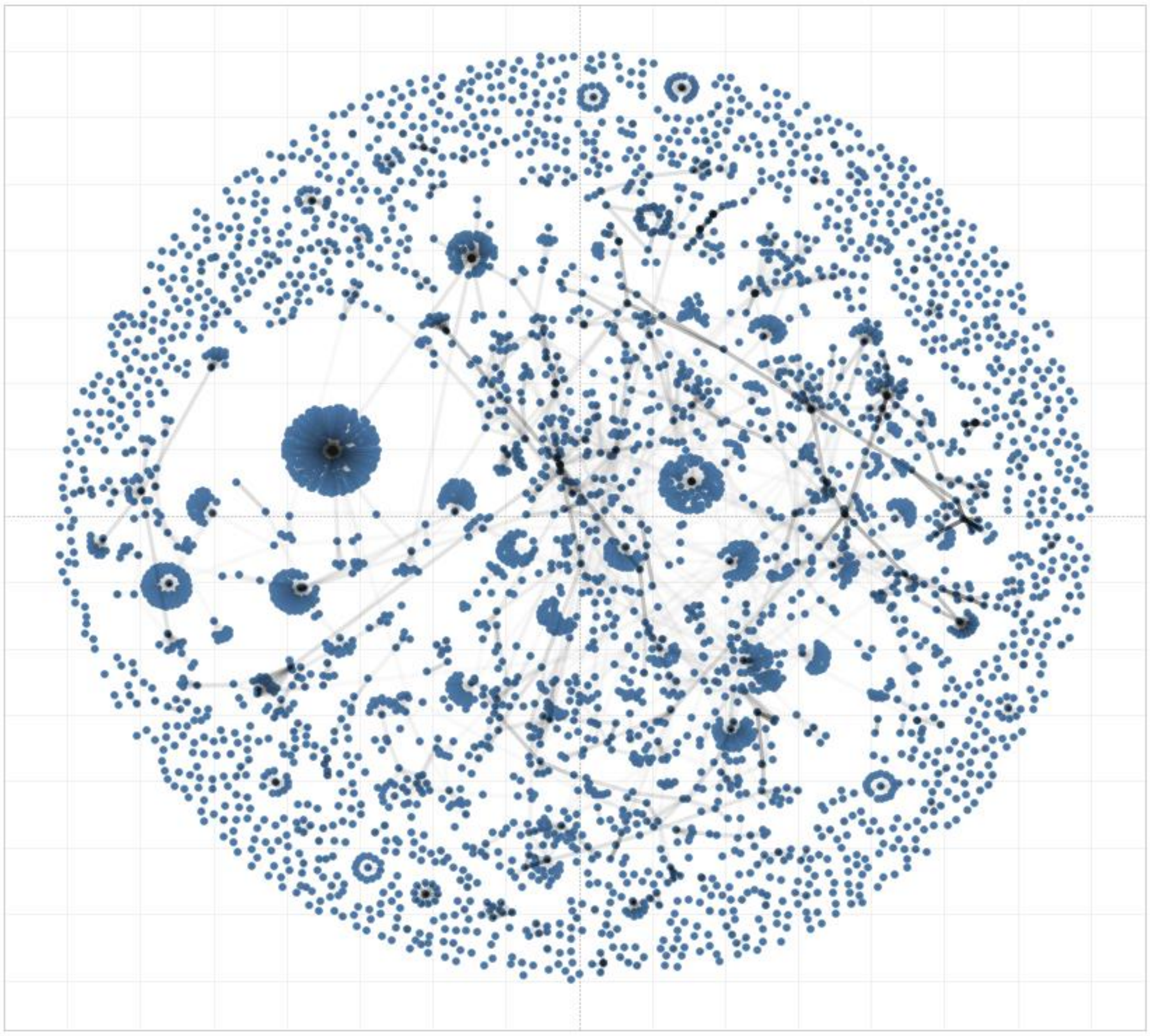


Figure 24: Human Network Visualization

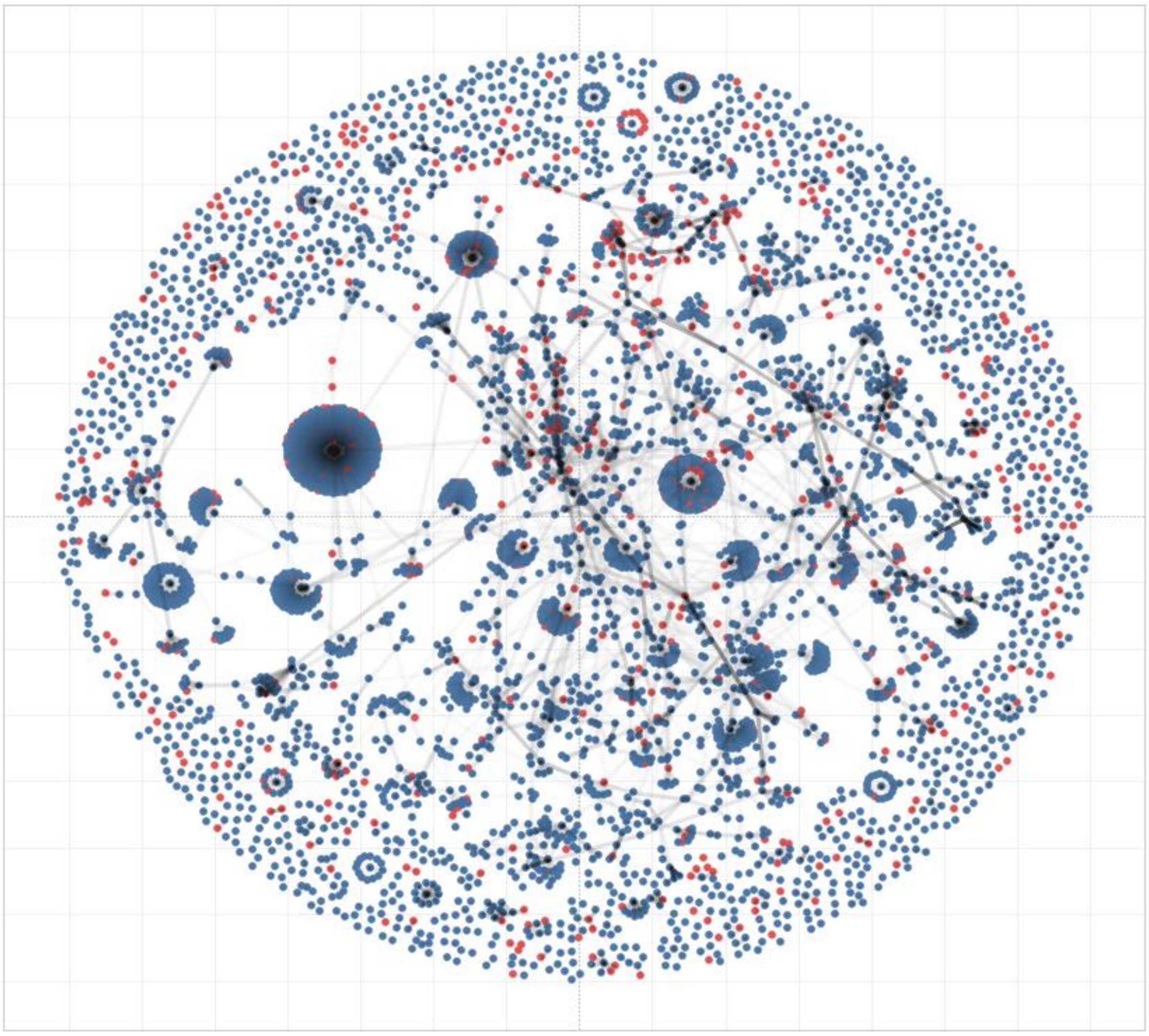


Figure 25: Full Network Visualization

6.2. Cluster Detection

Similarly, cluster detection was performed on Gephi to help visualise the communities present in the network (Figure 26). The modularity of the dataset was calculated using a community detection algorithm. It can be seen that multiple communities/clusters are present in the dataset and verify the use of graph-based algorithms for analysis. This is due to the strong network structure made available by connecting users via hashtags present in Tweets. Additionally, an average clustering coefficient of 0.482 was calculated meaning that several of the users in the network are connected to many other users in the network. This displays the strength of analysing the users as a network as opposed to individually.

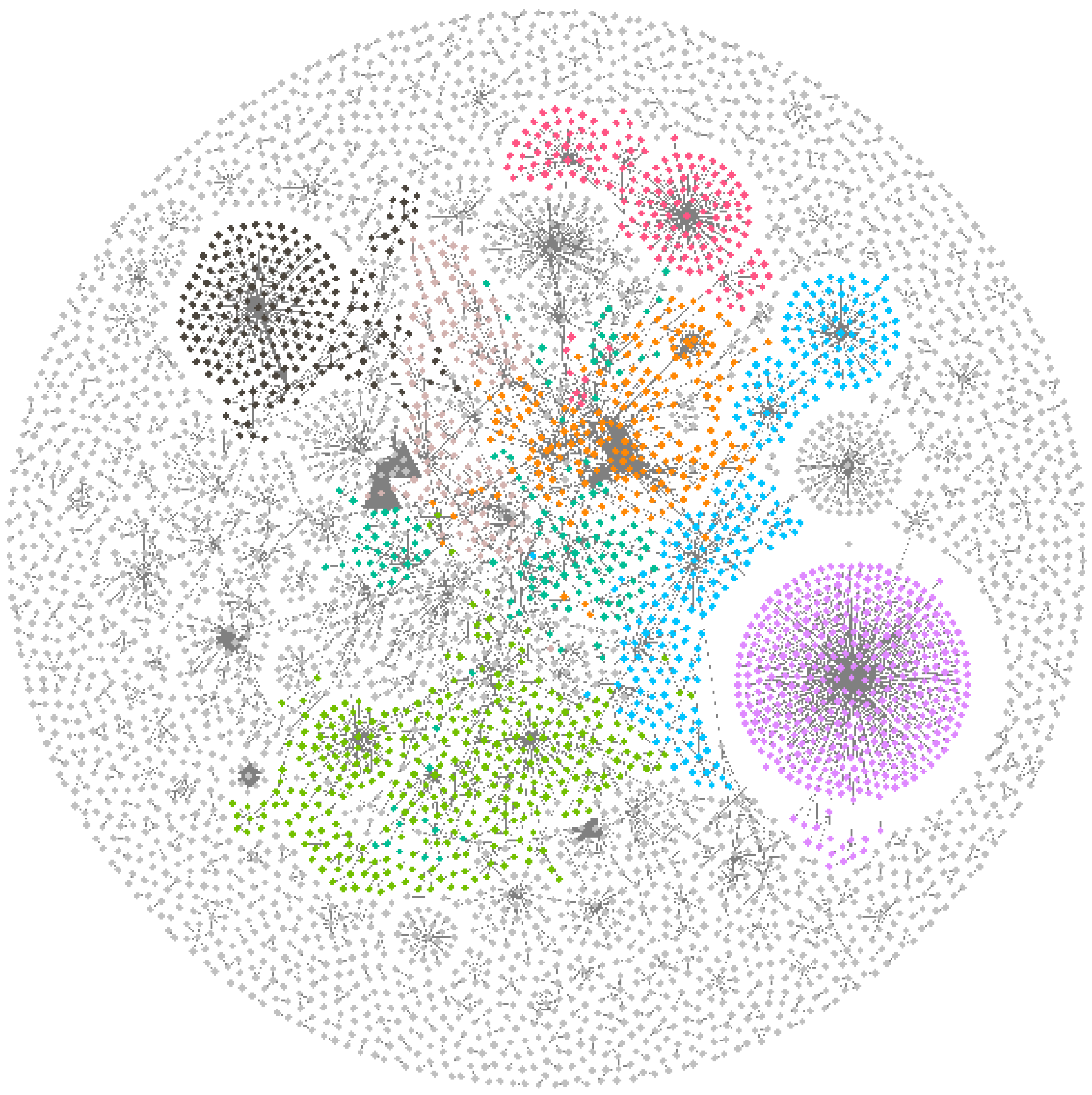
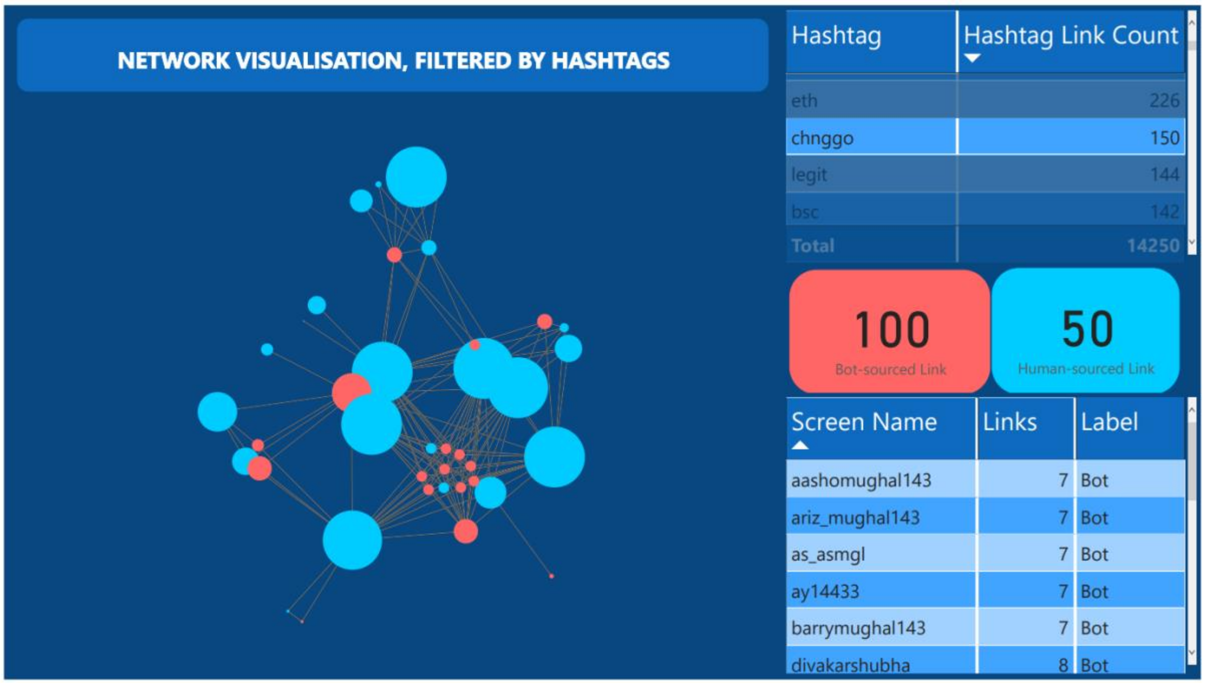


Figure 26: Clustering Visualisation of Dataset

6.3. Possible Hashtag-triggered Botnet

Plotting our network graph on Power BI, we have been able to identify the hashtag with most connections from bots (out of the most popular 20). Figure 27 is a visualization for accounts connected together via the hashtag #chnggo, with two thirds of its activities started by accounts categorized as bots (red). The weight of the nodes was calculated by the number of connections it has (from either a source or a target position). Since bot accounts are smaller in circle size, meaning that it has less interaction with other accounts, we can assume that the activities by bot accounts happen after the human one was posted, attempting to mimic human retweeting behavior. Additionally, we see that 'Screen Name' table in the bottom right suggests that these bot accounts all have very similar names, giving further indication that this cluster of bots originated from the same source, possibly the 'botmaster'. To address this, future research can include timestamps of Tweets and NLP techniques on Twitter Screen Names to help improve bot detection process.



Hashtag	Hashtag Link Count
eth	226
chnggo	150
legit	144
bsc	142
Total	14250

100
Bot-sourced Link

50
Human-sourced Link

Screen Name	Links	Label
aashomughal143	7	Bot
ariz_mughal143	7	Bot
as_asmgl	7	Bot
ay14433	7	Bot
barrymughal143	7	Bot
divakarshubha	8	Bot

Figure 27: Possible hashtag-triggered Botnet visualization

6.4. Correlation Matrix

The Correlation Matrix for our Twitter dataset is created using existing Python library. Each attribute of the data table is listed on every row and column. The value in a cell indicates the correlation between the horizontal row and the corresponding vertical row properties in the chart (note that cells on the diagonal line top left to bottom right are always 1.0 indicating a perfect correlation between an attribute and itself):

- The value is closer to 0, indicating the low correlation between the two properties.
- The value is closer to 1, representing two properties with a proportional relationship.
- The value is closer to -1, representing two properties with an inverse relationship.

From the correlation matrix results, many attributes seem to portray a low correlation to one another (approximately 0.0) as the information they convey is independent of other attributes. There is a visible relationship between Number of Followers versus Number of Lists. However, there is no evident relationships between the class label (Botometer) and any other attributes. This suggests that more sophisticated analysis, such as modelling can provide more insight into this relationship, if any.

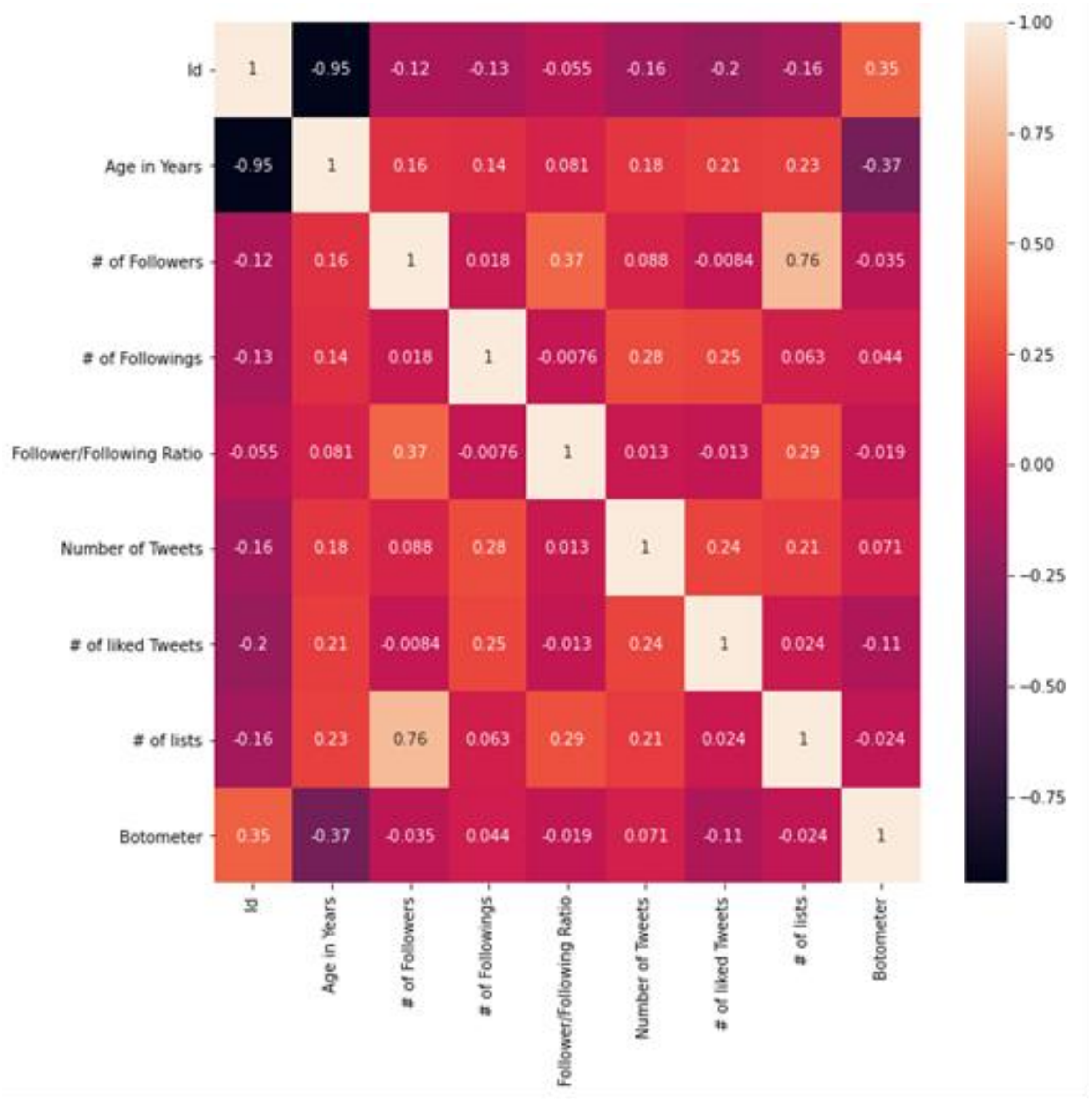


Figure 28: Data Features Correlation Matrix

6.4. Age of Account Distribution

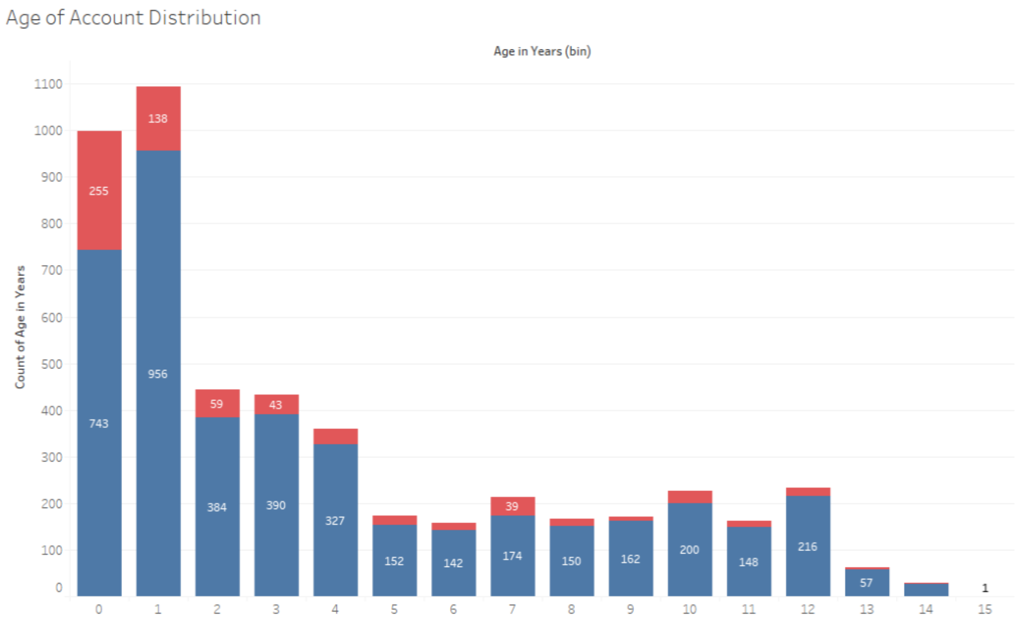


Figure 29: Age of Account Distribution

The Age of Account distribution reveals that more than 70% active bots tweeting about cryptocurrency are “fresh”, meaning that they have only been created around a year or less. Therefore, this feature can be a good indication or explanation as to whether a Twitter user is a human or a bot. It should be noted that this might be partially due to the fact that discussions on cryptocurrency trading have only been around for the last 3 years.

6.5. Verification Check

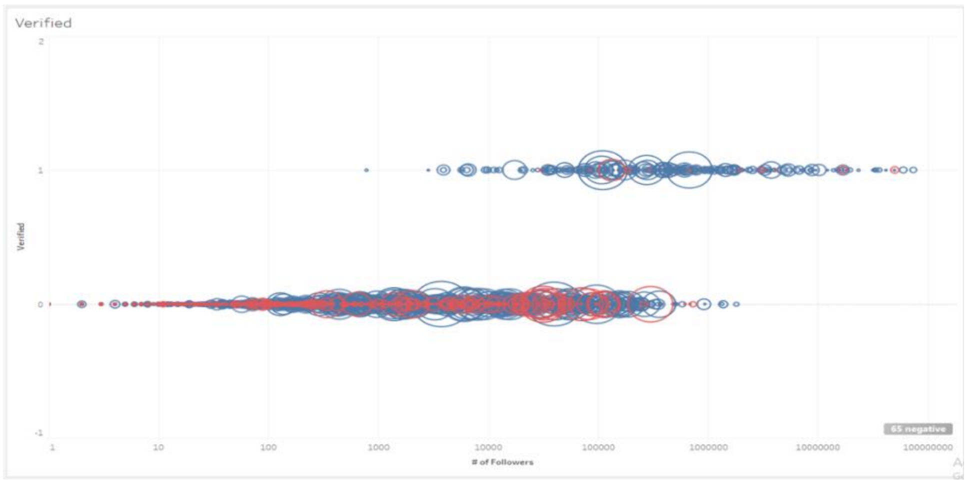


Figure 30: Verified Check versus Number Followers (Number of Like Tweets – Circle size)

This observation shows that almost all bot accounts are not verified. When plotting it against the number of liked Tweets, indicating the users’ interactive activities on Twitter (size of red/blue circles), it is clear that bot accounts have trouble being verified despite the effort of mimicking human-like interactivity. Visually, verification is a good indicator of a human user.

6.6. Follower/Following Ratio

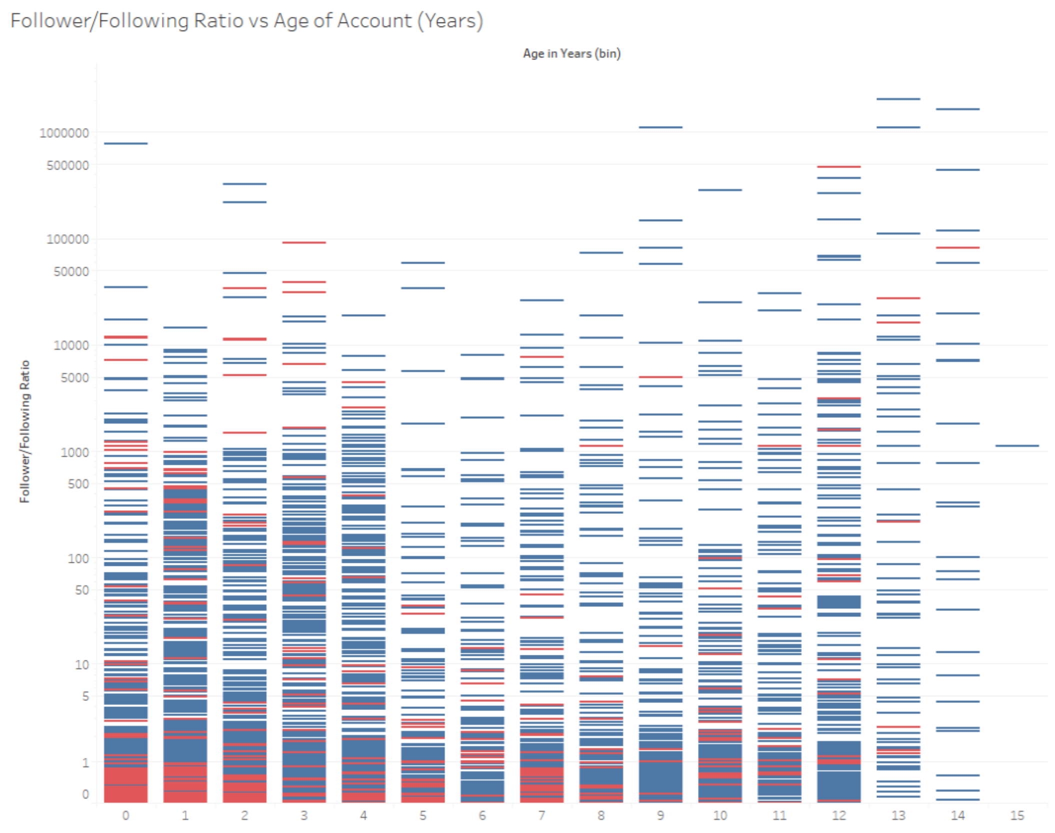


Figure 31: Follower/Following Ratio versus Age of Accounts (Years)

When plotting accounts' Follower/Following Ratio throughout the Age of the Accounts (in Years), we noticed that there were a large number of bot accounts (red) located in the bottom left corner of the graph. This contradicts with Chu *et al.* (2012) and previous studies that a high follower/following ratio could indicate a spamming behaviour. In fact, recently created bots tend to have a lower follower/following ratio (closer to 1). As Twitter have imposed a limit on the follower/following ratio for bot concerns, we could explain this as an attempt to imitate human accounts.

6.7. Number of Liked Tweets versus Follower/Following Counts

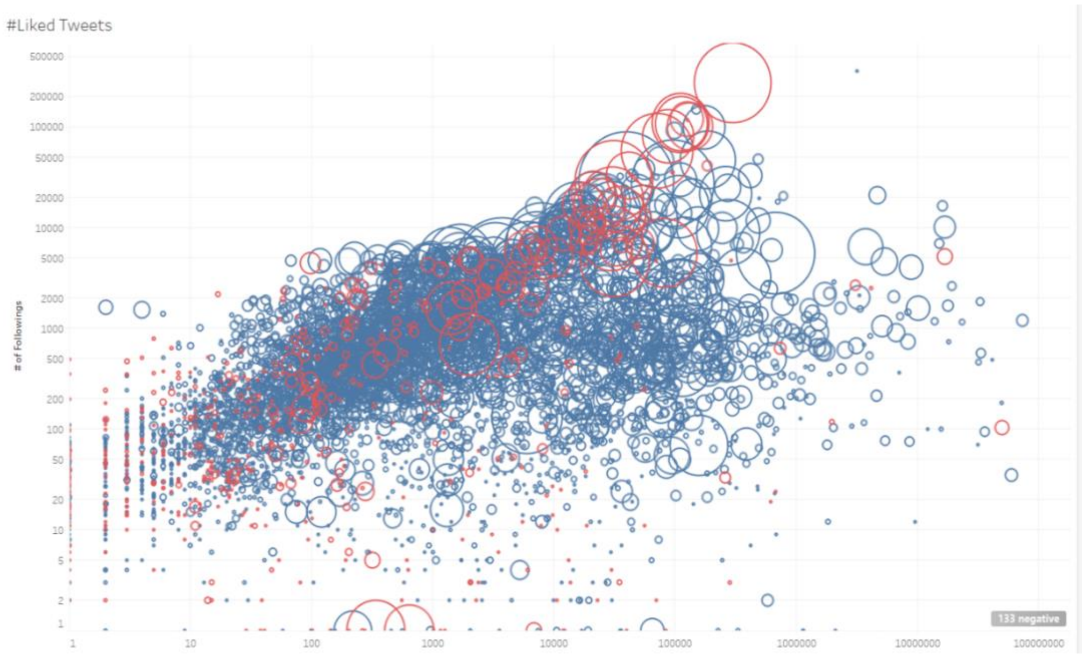


Figure 32: Follower vs. Following counts (Number of liked Tweets – circle size)

Similarly, bot accounts with higher number of both followers and followings (high follower/following ratio) tend to like more Tweets. Again, this seems to be an attempt to imitate a high interaction on Twitter to avoid being "bot-busted".

6.8. Unsupervised Learning

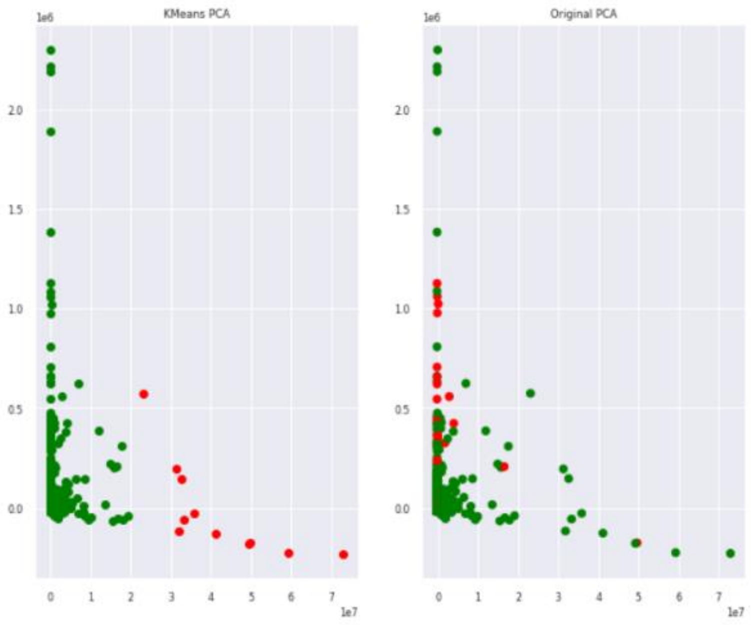


Figure 33: Visualisation of KMeans Prediction

From the results generated by our experiments, we find accuracies that are comparable to the supervised learning models used in Section 5.7.1, ranging from 0.80 to 0.90. This falls within our expectations as the dataset is heavily imbalanced. However, the results begin to differ when examining the F1 Score of the unsupervised models. The F1 Score combines the precision and recall of the models to provide a better measure of accuracy on unbalanced datasets. From the F1 Score's observed, the models are performing very poorly and have no difference from simply predicting all classes as the majority label. Overall, this confirms our hypothesis that the clusters and outliers from unsupervised learning with our dataset are not a good feature to examine for Bot detection.

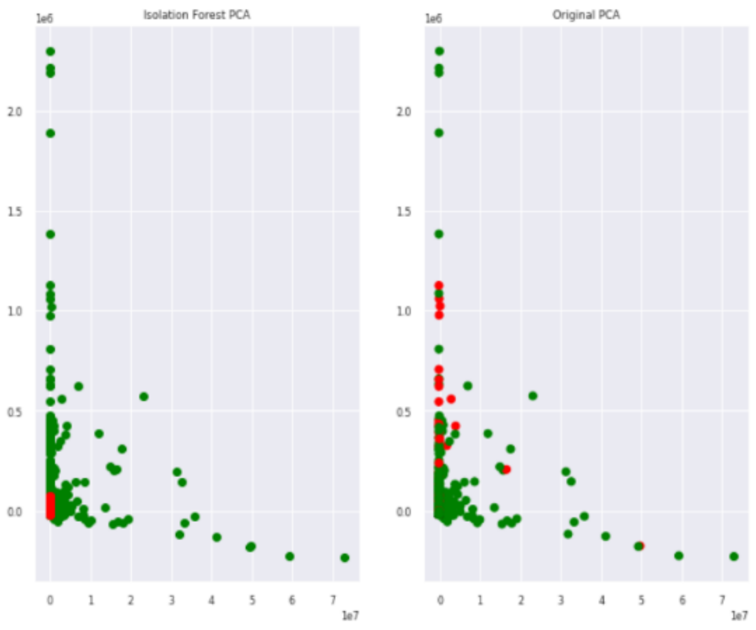


Figure 34: Visualisation of Isolation Forest Predictions

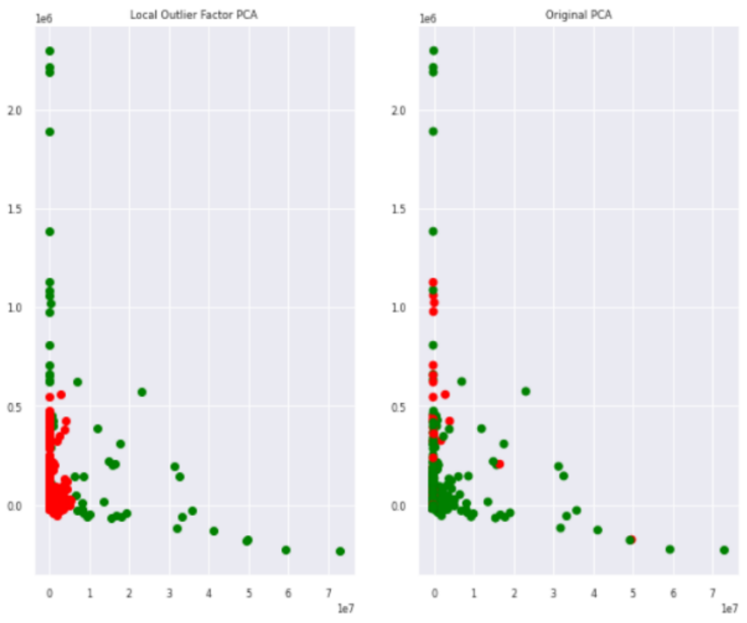


Figure 35: Visualisation of Local Outlier Factor Predictions

To visualise the results of the unsupervised learning models in an interpretable manner, the number of dimensions had to be reduced using Principal Component Analysis (PCA) dimensionality reduction. The resulting series of graphs depict the identified cluster and outliers relative to other points in the dataset, contrasted with the distribution of the labels in the imbalanced dataset. In figures 33-35, the Bot labels are represented by red points and human labels by green points. From the figures shown, we can confirm that there seems to be little relation in terms of clusters and outliers with the detection models used.

6.9. Modelling Findings

As we touched upon earlier in the results summary (Section 5.3). We find that the traditional based machine learning algorithms outperform the graph-based machine learning methods. This was different to our initial hypothesis that the addition of network features would improve the accuracy of graph-based models over traditional based models. This may be because traditional methods have been shown to outperform graph-based methods in simple classification tasks such as ours. Graph-based methods are still in their infancy and the addition of mechanisms such as attention (Graph Attention Networks; Veličković *et al.* 2017) in the future can aid in our goal of explainability.

We have also come to the conclusion that an ideal bot detective model must be supervised as unsupervised models don't seem to perform well with the features as well as the given dataset. We have also identified # of liked Tweets and # of Tweets as the most important feature used in ML models, meaning that these models categorise these bots based on their interaction activities online.

6.10. Explainability from Modelling

Feature importance was measured when using the Logistic Regression, Gradient Boosting, Decision Tree and Random Forest models. It can be seen that the "Number of Tweets" and the "Number of Liked Tweets" are the most important features when determining social bots in the model. An interesting data point to note is that the "Verified" feature is the least important feature, which is contrary to our analysis in Section 6.5.

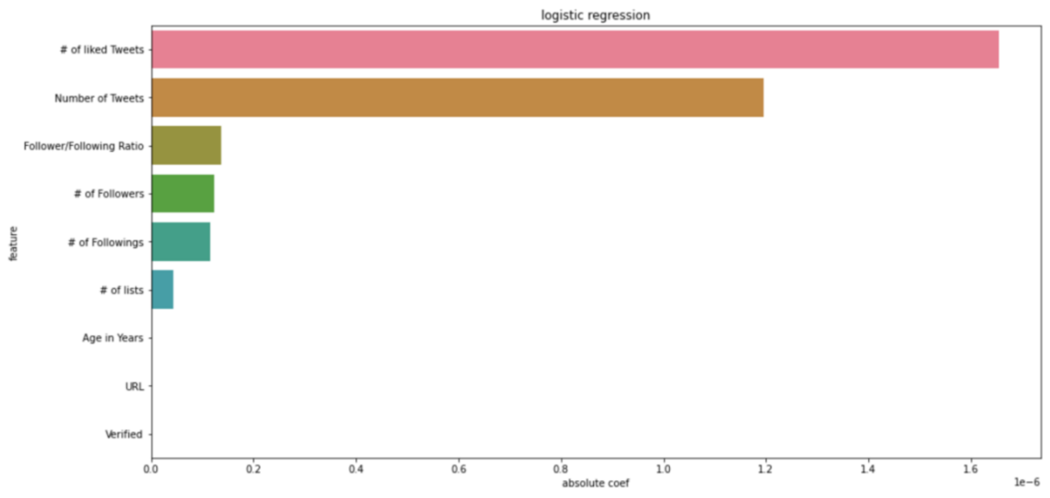


Figure 36: Feature Importance – Logistic Regression

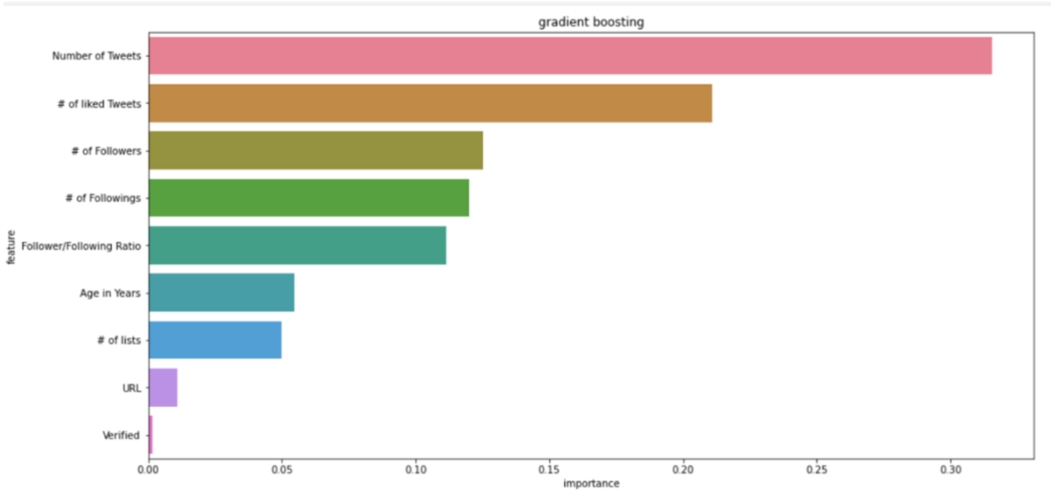


Figure 37: Feature Importance – Gradient Boosting

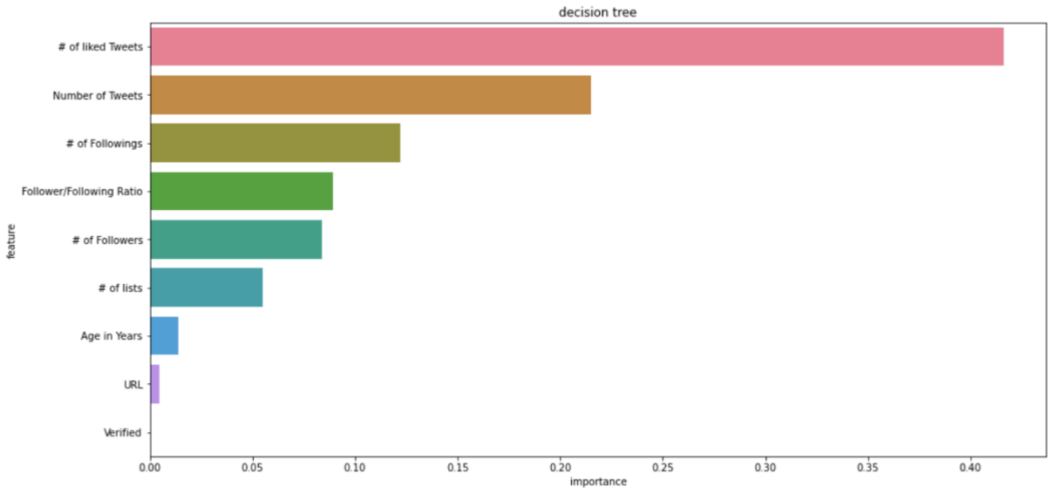


Figure 38: Feature Importance – Decision Tree

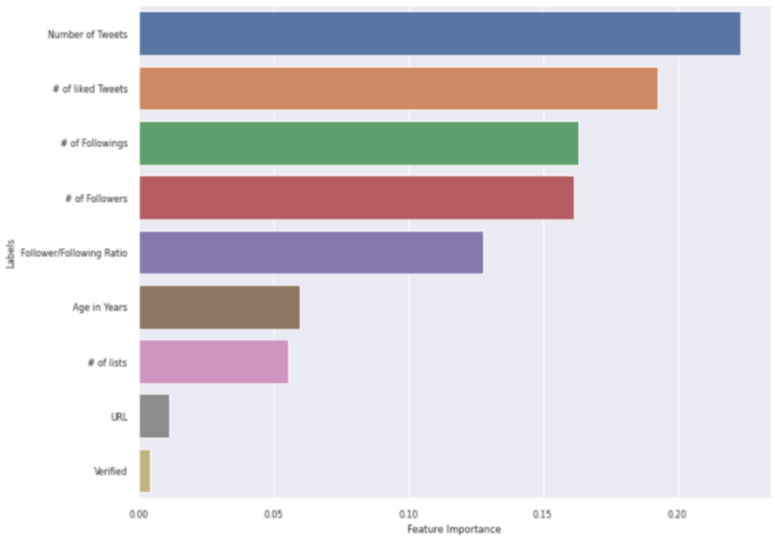


Figure 39: Feature Importance – Random Forest

Although we have not been able to develop a novel methodology using explainable artificial intelligence for bot detection, through our research we have been able to address key findings for future development of such model.

The findings on Correlation Matrix and dataset analysis provide us with an understanding of the fields used in Twitter bot detections, meaning that we are provided with an initial understanding of which field values could indicate a bot account (Age of Account, Verification, and how active the account is on Twitter via Tweets postings/liked Tweets and Follower/Following). These features are then double checked and ranked by the predictive models to determine the most influential features (that we now learned as # of liked Tweets and # of Tweets).

By running the models, we have also noticed that the implementation of an explainable method must be a supervised model for the best result (as Gradient Boosting and Decision Tree in our model). Graph-based models might be suitable for detecting bot clusters, however, not only they provided a lower accuracy, but also they would not be as explainable as the tradition machine learning models.

From the network visualisations, bot clusters have also proved to be triggered by certain hashtags, meaning that in an XAI model the ratio between bot-sourced connections and human-sourced connection can also be taken into consideration. Other identified aspects that could be included in explainability are NLP with the Tweet contents, hashtags and usernames, as well as timestamps of Twitter activities.

However, our visualisations and evaluation also illustrate the propensity for bots to mimic human behaviour leading to a widespread distribution of bot accounts within the network. This makes the task of bot detection increasingly complicated and difficult to generalise. We recommend further analysis into this behaviour as it is likely an ongoing issue due to the sophistication of the available methods.

7. Recommendations & Deployment

In summary, the model that obtained the highest accuracy and F1-score was the Gradient Boosting model with a bot detection accuracy of 89% and an F1-score of 0.88. This finding suggests that traditional machine learning models perform better than the graph-based machine learning models when detecting social bots. The Gradient Boosting model was trained on a balanced dataset, meaning there was no class imbalance. This may propose some problems going forward because in real world datasets, it may be difficult to obtain a dataset with such a large number of bots. K-fold validation, up sampling the minority class (bots) and down sampling the majority class (humans) is recommended when training models to ensure a higher accuracy in bot detection.

Another conclusion we obtained from the modelling is that unsupervised models trained on our dataset did not perform as well as the supervised models. This indicates that more sophisticated unsupervised techniques are required or that the dataset features do not translate well to an unsupervised task.

With regard to explainability, our findings suggest that newer, unverified Twitter accounts with a low Follower/Following ratio are more likely to be bot accounts. Additionally, the "Number of Tweets" and the "Number of Liked Tweets" are the most important features when determining the likelihood of bot behaviour. However, our visualisations and evaluation of botnets also illustrate the propensity for bots to mimic human behaviour leading to a widespread distribution of bot accounts within the network. We find that bot accounts can copy a human accounts initial hashtag and propagate this hashtag in the network. These bot accounts that 'piggyback' off the human account to disguise themselves as legitimate users, display very similar behaviour and Screen Names. Inclusion of activity timestamps and Natural Language Processing (NLP) techniques used on the Screen Names can help detect this botnet behaviour. Together, these findings make the task of bot detection increasingly complicated and difficult to generalise. We recommend further analysis into this behaviour as it is likely an ongoing issue due to the sophistication of the available methods. Finally, future work on graph-based methods such as Graph Attention Networks (Veličković *et al.* 2017) and attention mechanisms in general should be explored to aid in the goal of explainable detection.

8. Difficulties Encountered

1. The main issue we encountered overall, was with the distribution of the dataset, where there were many more samples of human accounts than bot accounts, resulting in an imbalanced dataset.
    - To counteract this problem, we applied under sampling and K-fold validation to the dataset, reducing the number of human labels (majority class) and up sampling the bot labels (minority class) to have a more balanced dataset.

2. Rate limits were also encountered on the data collection procedure, limiting the number of samples that could be collected in a given time period, causing delays to our modelling process whenever we adjusted parameters and settings.
    - We made sure to plan and prepare in advance, by completing our data collection well ahead of deliverable dates, the effects of the delays were minimized.

3. In the modelling phase of the project, the implementation of several different models were performed by different team members, this resulted in separate pre-processing methods and evaluation metrics that made comparison of results difficult.
    - Through conducting a critical meeting to resolve the issues, one common pre-processed dataset was generated with agreed upon evaluation metrics to measure model performance.

4. The process of selecting appropriate Twitter tags for data collection that could appropriately represent our area of study was another major concern.
    - We addressed this issue by selecting popular and trending tags with the help of publicly available websites.

5. After the data collection procedure, we had found some collected accounts were removed by Twitter, resulting in removal of features that were crucial for our Bot scoring process.
    - Although we are uncertain as to the reason behind the account removal, to address this issue, we assumed the deleted accounts were Bots that Twitter had identified and removed from their network.

9. References

BitInfoCharts (2021). *Bitcoin Tweets historical charts*. https://bitinfocharts.com/comparison/bitcoin-tweets.html

Breunig, M.M., Kriegel H.P., Ng, R.T. & Sander, J. (2000). LOF: identifying density-based local outliers. *Association for Computing Machinery.* 93-104. https://doi.org/10.1145/342009.335388

Forgy, E.W. (1965) Cluster Analysis of Multivariate Data: Efficiency vs Interpretability of Classifications. *Biometrics. 21*, 768-780.

Hamilton, W., Ying, R., & Leskovec, J. (2017). Inductive Representation Learning on Large Graphs.

Ho, T.K. (1995). Random decision forests. *Proceedings of 3rd International Conference on Document Analysis and Recognition*. *1*, 278-282. https://doi.org/10.1109/ICDAR.1995.598994

Lielacher, A., and Pickering, A. (2020). *Fake views: How social media bots distort the crypto narrative*. Brave New Coin. https://bravenewcoin.com/insights/fake-views-how-social-media-bots-are-distorting-the-crypto-narrative

Liu, F.T., Ting, K.M. & Zhou, Z. (2008). Isolation Forest. *2008 Eighth IEEE International Conference on Data Mining,* 413-422, https://doi.org/10.1109/ICDM.2008.17

K.-C.Yang, O.Varol, C.A.Davis, E.Ferrara, A.Flammini and F.Menczer (2019) "Arming the public with artificial intelligence to counter social bots" Human Behaviour and Emerging Technologies.

Kipf, T., & Welling, M. (2016). Semi-Supervised Classification with Graph Convolutional Networks.

Song, Y. Y., & Lu, Y. (2015). Decision tree methods: applications for classification and prediction. Shanghai archives of psychiatry, 27(2), 130–135. https://doi.org/10.11919/j.issn.1002-0829.215044

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2017). Graph Attention Networks. https://arxiv.org/abs/1710.10903

Zi Chu, Gianvecchio, S., Haining Wang, & Jajodia, S. (2012). Detecting Automation of Twitter Accounts: Are You a Human, Bot, or Cyborg? *IEEE Transactions on Dependable and Secure Computing*, *9*(6), 811–824. https://doi.org/10.1109/TDSC.2012.75

10. Appendix

All datasets, notebooks, and reports for this project can be found in the following GitHub Repository and Google Drive links.

1. GitHub Repository
2. Notebook Drive