Department of Computer Science

Submitted in part fulfilment for the degree of BSc.

Bot Detection in Mastodon

Izz Abd Aziz

Version 3.0, 2025-May

Supervisor: Angus Marshall

To all students everywhere

Acknowledgements

I would like to thank my supervisor, Angus Marshall for the help he gave during the making of this project.

Contents

Executive Summary v

1 Introduction 1

2 Literature Review 3
- 2.1 Introduction . . . 3
- 2.2 Bots in Social media . . . 4
  - 2.2.1 Spam bots . . . 4
  - 2.2.2 Engagement bots . . . 4
- 2.3 Mastodon and Decentralised Social Networks . . . 5
- 2.4 Bot detection algorithm . . . 5
  - 2.4.1 Support Vector Machine . . . 6
  - 2.4.2 Naive Bayes . . . 6
  - 2.4.3 Decision Tree . . . 7
  - 2.4.4 CART algorithm . . . 8
- 2.5 Literature Summation/ Overview . . . 8

3 Methodology 9
- 3.1 Packages and training models . . . 9
- 3.2 Private server and data collection . . . 10
- 3.3 Data Preprocessing . . . 11
- 3.4 Decision tree model (CART Algorithm) . . . 12
  - 3.4.1 Model building . . . 13
  - 3.4.2 Model evaluation . . . 14

4 Implementation 16
- 4.1 Mastodon setup on a private server . . . 16
- 4.2 Bot in Mastodon . . . 17
- 4.3 Data extraction . . . 18
- 4.4 Bot detection model . . . 19

5 Result and Evaluation 21
- 5.1 Test and Result . . . 21

6 Conclusion 23

List of Figures

3.1 Decision Tree Graph. 12
3.2 Confusion Matrix representation. 15

4.1 Decision Tree for Mastodon dataset. 20

5.1 Confusion Matrix. 22

Executive Summary

The emergence of decentralised social networks, such as Mastodon, has introduced new challenges in bot detection. Unlike traditional platforms like Twitter or Facebook, Mastodon's architecture consists of independently moderated instances, making uniform moderation and bot detection more complex. This project explores the effectiveness of machine learning models in identifying bots within this decentralised framework.

A private Mastodon server was deployed to ethically simulate and observe bot behavior without interfering with public users. Using engagement-based metrics (such as favourites and bookmarks per day), a dataset was created and analyzed using the CART (Classification and Regression Trees) algorithm. The model was developed in Python, leveraging the Scikit-learn library, and trained on data collected from accounts within the private instance.

Results from the experiment showed the CART model performed moderately well, with an overall accuracy of 66.67% and high precision in identifying human accounts. However, bot classification was less consistent, indicating potential overfitting due to the small dataset size. Despite limitations, the study confirms that decision tree models can be a practical starting point for bot detection in decentralised platforms and lays the groundwork for future expansion with larger datasets and refined algorithms.

1 Introduction

Bots in social media have become both a powerful tool and a significant challenge. These automated programs can mimic human behaviour and are often used for various purposes, ranging from legitimate uses like customer service automation to more harmful ones like spreading misinformation, spam, or scams. [1][2] For example, there's an article that claims bots on Twitter highly influenced the 2016 US presidential election by amplifying political messages. [3] Other than that, there are so many bots that are sending malicious links to the comment section or chat box on this platform. [4] This creates an opportunity for online criminals to send viruses to the victim's devices. [5] In this project, I would like to study the bots that are being used by criminals within social media Mastodon. By using Artificial Intelligence, I will make a bot detection algorithm so that AI bots can be reduced within the apps Mastodon.[6]

To begin with, Mastodon is being selected as a testing platform because it is open source and decentralised social network.[7] This is a perfect nature for the project as it is not controlled by one company and experimenting on this platform can be made easier.[8] Also, Mastodon's API is generally praised for its clarity and ease of use than Twitter. [9] I could use this advantage to gather data, test applications and run some tasks. [10]Plus, decentralised social networks has gained more attention recently.[11] Thus, experimenting on this platform can offer early insights about bot in decentralised social networks .

To create a Mastodon bot, I need to connect to the Mastodon API to access instance data. This involves user profiles and post activity. [10] Once we have access to the API, we can write some code using the Python language to create a bot detection algorithm.[12] Then, we need some parameters that differentiate between real humans and bot accounts. For example, high post frequency, lack of profile information, and typical hashtags used. [13]Next, write a function to collect and analyse these parameters for each user. Then, create a scoring system based on the parameters collected. Bots could be labeled if the account exceeded certain values of the scoring system [13] For example, if the account is spamming the same comment on the different posts, it could be identified as bot.

Then, I will further my research on what bots could do to help criminal activity around Mastodon. For example, bots that spam the comment section with malicious links can give big advantages to the criminal to gain the personal information of the victim. This includes directing the victim to fake e-commerce platforms or banking sites. [5] Other than that, bots can also create fake accounts to promote nudity and pornography to users. [14]Some unfortunate users might fall into this trap and give their bank details to the criminal. Some unlucky victims experience sextortion that involves blackmailing the sexual videos or pictures to be published if demands are not fulfilled. [15]

It is common nowadays in social media platforms where bot accounts can be found everywhere around internet. While researchers have developed sophisticated detection methods for centralized platforms like Twitter, Mastodon's unique decentralized nature changes the game entirely.

This project addresses:

1. When we try to apply bot detection algorithm from centralised social networks, do they actually work in Mastodon?
2. What featuress that uniquely identify bots in decentralised network?
3. Is there any improvement that Mastodon could use to improve the bot issue.

2 Literature Review

2.1 Introduction

Decentralised Social Networks (DSNs) like Mastodon, Lens Protocol and BlueSky have emerged as a solution against privacy and control issue over users' data. [16] This can be made possible thanks to two keys of DSN: no centralisation because the availability of open-source software that allows everyone to make their own server, and the use of ActivityPub protocol that allow users to interact with each other even from different instances. [17] Mastodon, in particular, has encountered greatest attention within users and researchers, making it most suitable Fediverse for this project. [17] Moreover, Mastodon is the alternative version of Decentralised Social Networks for Twitter/X . [18]

As DSNs become more popular among internet users, high number of bot activity within online social networks has become the main concern around the globe. Research has shown that bot can impact political discussion, creating virtual communities within specific conversation, and influencing network sentiment. [19] Even though bots in Twitter/X being in smaller numbers, they have a profound impact on the popularity of the content and activity on Twitter/X. [20]

Existing bot detection algorithm shows magnificent impact to tackle down threats posed by social media bots. Current bot detection methods employ a set of technique that includes behavioural analysis, network representation analysis, and graph embedding approaches. For example, MRLBot employ unsupervised learning, combining behavioural representation and relationship representation learning models to create fused representations for bot detection. [21]On the other hand, BotChase uses both unsupervised and supervised machine learning to perform graph-based bot detection system. [22] In conclusion, there are so many methods had been used for bot detection in social media. However, there are little to no research for bot detection in Mastodon since every bot created needs to be flagged as a bot. Despite the bot declaration procedure in Mastodon, the bot created was not flag by mastodon moderators, highlighting the opportunity to dig more about bot detection in Mastodon. [23]

2.2 Bots in Social media

Social media like Instagram, X (Twitter) and Facebook are famously known with high bots account in the platforms. There is various type of bots in social media with different purposes, behaviours and impacts. For instance, spam bots, engagement bots, propaganda bots and fake account bots are one of the most common bots in the platforms. Twitter is the one of the highest percentage of bots account which estimated around 9% to 15 %. [24] Although some bots perform beneficial and efficient task, they can also creates political, social and economic problems due to uncertainty and behaviour of the them. [25]

2.2.1 Spam bots

Spam bots have become a critical threat to online social media, especially in centralised social network. These bots posed a threat to social media with their malicious behaviour, spreading misinformation messages and posting unwanted content. [26] [27]. Spam bots becoming more advance which creates a challenge to detect them even in this advanced era of technology. [28]Surprisingly, there are some studies that shows that not all spam bots are malicious. They can also generate benign and informative tweets (news and blog), which amplify information dissemination and increasingly becoming useful to the users. [29]

Despite that, benign bots that perform valuable services did not received enough attention due to low threat in Internet. [30] These bots usually distinguished as a real account because there is no master list defining which Twitter accounts are real users. [25] Thus, spam bots and fake profile bots are usually the same because fake profile bot will be created to make it looked like real person. Then, this bot will spam like and posts in social media making it look like it is done by real account.

2.2.2 Engagement bots

Social media influencers is one of the digital content creators who has a huge follower on social media and use this advantage to build online presence, promotes brands and earn money. [31] Usually, these people on social media will use engagement bots to manipulate their followers by automatically gets the likes or comments on social media making a significant impression on their posts. [8] Other than that, some use engagement bots to influence cryptocurrency market by increasing like and retweet about any tweets regarding crypto, creating a pump and dump schemes. [32]

2.3 Mastodon and Decentralised Social Networks

Mastodon's architecture works on decentralised social networks, which consist of interconnected communities located in different servers (instances) that form a federated network. Each instance is independently owned and moderated, which follows their own specific content and policies. [8]. This federation policies enable administrators to create rules who may apply different strategies to manage their communities. [32]. This decentralised architecture can influence how the bot will be detected within the instances because rules are not the same in every instances. Despite that, Mastodon focuses on interest-based community engagement and niche communities [33]where bot within the instances is very unlikely.

2.4 Bot detection algorithm

Bot detection algorithm like Support Vector Machines (SVM), Naïve Bayes and Neural Networks for supervised learning are commonly used these days. [34]Researchers categorise existent algorithm into three classes; structure-based bot detection, crowdsourcing-based bot detection, and machine learning-based bot detection.[35]

For machine learning-based bot detection, it is proved that Decision Tree shows the highest accuracy when detecting botnet. [36] Decision Tree obtained a mean score of 85% in F1 score – evaluation metrics to measure model accuracy. [36] On the other hand, structure-based bot detection is better in detecting structural anomalies in a graph-based representation of data [37] while crowdsourcing-based bot detection are highly cost because it requires interactions with human user.[38]Plus, this approach is not suitable for this project because detecting bots are done in account level, not behaviour of bot itself. [38]

On another studies, machine learning techniques for phishing detection shows that Random Forest (a collective of Decision Tree in a model) outperform Logistic Regression, Classification and Regression Trees, Bayesian Additive Regression Trees, Support Vector Machines and Neural Networks. [39] [40] Surprisingly, different studies have found that using Ensemble Generalised Multiclass Support Vector Machine (EGMSVM) achieve high efficiency compared to traditional SVM and ANN-based algorithm for health evaluation. Unfortunately, this model is more time-consuming than traditional models making it complicated and less reliable for this study. [41] Based on these findings we could narrow down the choice of bot detection algorithm which only consist of traditional models only.

2.4.1 Support Vector Machine

Support Vector Machine (SVM) has been widely used in different research, including face recognition, diseases diagnostic, text recognition, sentiment analysis, plant disease identification and intrusion detection system for network security application. [42] SVM has gained the popularity among researchers because it proved the efficiency and effectiveness working with small training sets. [43] SVM can be used to identify network traffic or user behaviour patterns to differentiate between botnet and human in internet. This is certainly applicable for detecting the intrusions in the network, where Intrusion Detection System (IDS) uses SVM to identify suspicious activity using anomaly detection. [44] Although most research are not specifically point to the bot detection in social media, it shows that SVM algorithm achieved a high level of accuracy in different real life practice, especially in face recognition, text recognition and disease recognition. [30] This gives the opportunity to explore the potential of the algorithm in bot detection scenarios

In summary, the studied papers are not directly addressing the bot detection using SVM. Instead, they show the algorithm's strength in different related domain they are closely related to bot detection such as anomaly recognition, pattern recognition, and network security. [42] The low amount of bot detection research in social media gives the opportunity in this project to study more about the technique, specifically in Mastodon.

2.4.2 Naive Bayes

Naïve bayes algorithm are one of the most effective bot detections in social media platforms, showing their ability in recognising automated bot accounts. This classifier is one of the most remarkable for data mining algorithms. [45] Interestingly, this algorithm has been used in various field of study which consist of medical disease detection,[46] text and document classification [35] and email classification. [47]Some studies have utilised this technique, showing positive results in distinguishing between bots and human account. [44] [48] For the study of bot detection in social media, Naïve Bayes shows incredible accuracy while testing in Twitter dataset with

89% accuracy. [12]

While Naïve Bayes shown an excellent accuracy, this classifier is still not the best in overall performance. One study show that a convolutional neural network (CNN) achieved overall accuracy of 98.71% while 97.6% accuracy using artificial neural network (ANN). [49] However, these models' performance is not suitable for comparison because this experiment was conducted without including the Naïve Bayes algorithm. Despite that, there is one study that compares the performance of decision tree, k-Nearest Neighbours and Naïve Bayesian. According to this study, decision tree outperformed the other algorithm with less error rate and easier algorithm. [50]

In conclusion, Naïve bayes classifiers have shown positive outcome for bot detection in social media, offering a high accuracy and computational efficiency. However, the quality of dataset can differentiate the effectiveness of bot detection methods. For example, the Naïve Bayes performs the best in large datasets in comparison to other classification technique. [51]

2.4.3 Decision Tree

Decision tree algorithm is one of the most important models that we studied in this project. This is because multiple studies have shown that decision tree outperform different models, including Naïve Bayes, k-Nearest Neighbours, Random Tree and SVM. [36] [12][51] Plus, decision tree are regarded as the most well-known algorithms for data classification. [44]

Decision tree consists of different algorithms such as ID3(Iterative Dichromotimiser 3), C4.5, CART (Classification and Regression Tree), CHAID (CHI- squared Automatic Interaction Detector), MARS. Also, decision tree is split into two type which is classification tree and regression tree. [52] Classification tree is suggested to use when task requires classification or prediction of outcomes. Meanwhile, regression tree is used to estimate a continuous variable's value. [53] In the study about prediction of flood, the combination of classification and regression tree (CART) were implemented. [54] Although the predictive accuracy are lowest between models, the new CART algorithm are executed for the first time. [54] This indicates that improvement could be made in the future enhance the capability of CART algorithm. Thus, this project will use CART algorithm to implement bot detection in Mastodon server.

2.4.4 CART algorithm

CART, which stands for Classification and Regression Trees, was introduced by Breiman in 1984. This algorithm is a type of decision tree that is build using both classification and regression trees. For classification part, CART constructs trees by performing binary splits on attributes. Gini index is used to find the best split of the trees. Meanwhile, CART builds regression trees to forecast a dependent variable based on a set of predictor variables over a given time period. The algorithm operates at an average processing speed and is capable of handling both continuous and nominal attribute data.

The ability of CART that can automatically handle the missing values using surrogate splits makes it more ideal with the dataset of our Mastodon. This is because the user experience while using the Mastodon was entirely by their own action. We did not provide any instruction or baseline when they were using the Mastodon. So, it is expected for any missing values to be present in the dataset.

2.5 Literature Summation/ Overview

Based on the literature review, Decentralised Social Networks (DSNs) like Mastodon offer users greater control over their data by removing centralised authority and using protocols like ActivityPub for cross-instance communication. Despite the benefits, bots remain a pressing issue in social networks, influencing political discourse, spreading spam, and manipulating engagement metrics. Various machine learning techniques have been applied to detect bots, including Decision Trees, Support Vector Machines (SVM), and Naïve Bayes classifiers, each with their own strengths and limitations. Mastodon presents a unique challenge for bot detection due to its decentralised structure and inconsistent moderation policies across instances. While SVM and Naïve Bayes have shown strong performance in related tasks like intrusion detection and email classification, Decision Trees have consistently demonstrated excellent accuracy in detecting bots on platforms like Twitter. As bot research in decentralised platforms like Mastodon remains scarce, there is significant potential for further exploration and development in this area.

3 Methodology

Research in social media bot detection has identified the decision tree algorithm as one of the most accurate and efficient models for this task. [55] However, this finding primarily applies to centralised social networks like Twitter and Facebook, as little to no studies have explored bot detection in decentralised social networks such as Mastodon. [55] To address this gap, this study will focus on implementing supervised learning-based bot detection in Mastodon, given its structural similarities to Twitter. The research will involve analysing the Mastodon API and simulating bots on a private server to follow the ethical and privacy considerations. Then, the decision tree algorithm will be employed to detect these newly generated bots within the controlled environment (private server).

3.1 Packages and training models

In this project, all models studied and evaluated will be implemented using Python for programming language and the Scikit-Learn (sklearn) for the library. Scikit-Learn is a versatile machine learning library that supports supervised learning for tasks such as classification and regression. Unlike more complex frameworks like TensorFlow, Scikit-Learn provides a user-friendly interface and seamless integration with popular Python libraries such as pandas and NumPy, enabling efficient data analysis and deeper insights.

Additionally, Scikit-Learn is particularly well-suited for small to medium-sized datasets, ensuring optimal performance. The project will also leverage Matplotlib for data visualization and graphs, along with pandas for data manipulation and analysis. It is important to note that training deep learning models typically requires large datasets and often involves hundreds of training epochs. To overcome slower training time and resource issue while training the machine learning, project will be done in Viking cluster.

3.2 Private server and data collection

Due to privacy and ethical issues, Mastodon instance for this study was created within the computer science building. This instance was created purposely for the study of this project and it is not connected to the public network (Internet) to ensure this instance is not accessible by other parties. For this project, the CSE/168X hardware lab was chosen as the ideal location for this project because it equipped with high security cyber computer.

In order to use Mastodon, users need to change to the Linux operating system. Then, they need to sign up to the Mastodon instance using the terminal. Once their account was approved, they can do anything inside the secured instance. This include all the feature that is available in Mastodon public network.Once the data is ready to be extract, features need to be collected in order to distinguish between human and bot account.

Feature to be collected.

1. Number of post
2. Account age
3. Followers-to-following ratio
4. Use of hashtags or mentions
5. Language complexity
6. Engagement metrics
7. Posting frequency
8. Response time

Since the experiment requires a little bit of Linux experience with strict requirement for participation, we are expected to receive a low participant count among the University of York. In result, we also expect that the dataset will be a little lower for bot detection training. For the sake of security of the lab, we list out the requirements that need to be met in order to join the experiment.

Participant requirement:

1. Participant have to be 18 years old and above.
2. Participant need to be familiar with X/Twitter

3. Participant need to be among the member of University of York (staff or student)

3.3 Data Preprocessing

Data preprocessing is the process of evaluating, filtering, manipulating and encoding data. The main objective of data preprocessing is to make a machine learning to understand the data thus producing the best output. During this process, data will experience the elimination of data issues (missing values), improvement of data quality and making the data useful for machine learning purposes. In a simpler word, process of data preprocessing can be divided into 3 categories which is cleaning, feature engineering and encoding.

Cleaning steps:

1. **Remove duplicates**: Check whether there are any duplicate entries in the dataset. Remove the duplicate to ensure each account only appear once per user.
2. **Handle missing value**: Value could be replace or fill in with placeholder depending on the feature priority.
3. **Consistent format**: Adjust all data fields according to their standard formats.
4. **Remove unrelated feature**: Only use relevant feature or column that is crucial for bot detection.
5. **Outlier check**: Find extreme outliers that could skew the result of model. Outlier could be remove or cap them using clipping method.
6. **Consistency check**: Check whether data are correct to the feature. For instance, if the number of post is negative, correct or remove the data.

Feature Engineering:

1. **Calculate frequency of post**: Create a columns that measure the frequency of post per user. Bot usually have high frequency of posts.
2. **Calculate follower-following ratio**: Create a column that represent the ratio of followers and following of the account. Bot usually have unbalance ratios..

3. **Examine profile metadata**: Create a column that indicate the presence of URL. Bot usually have a URL to link to phishing website.
4. **Calculate engagement ratio**: Sum the post interaction (bookmark, favourite and reblog) and calculate the ratio. Bots usually have high engagement rates especially engagement bots.
5. **Collect text-based feature**: Some feature like hashtag frequency or mention frequency should be counted because bots often overuse hashtags or mentions to allow more presence in social media.

Data encoding:

1. Binary column is created for each distinct category in the variable.
2. Set to 1 when category is present in a sample while set to 0 when it is not present.
3. Conversion of column after data encoding will be as diagram below.

3.4 Decision tree model (CART Algorithm)

For the model, decision tree algorithm will be implemented to detect bots in the Mastodon instance. The detection in this study will be trained on a dataset comprising approximately 17 of user's profile with different amount of bookmarks, toots, favourite and reblog. The dataset follows a split of 70 % for training, 20% for validation and 10% for testing. This distribution aligns with machine learning best ratio - a 70:30 ratio for training vs testing validation.The decision tree diagram can be represented as figure.

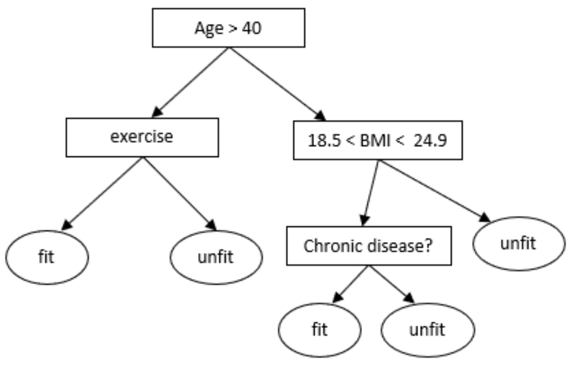


Figure 3.1: Decision Tree Graph.

CART algorithm will be use in this project because it is effective working in a simpler model as long as there is a high availability of quality and good data [].

3.4.1 Model building

To split the nodes of the tree, Gini Index is usually used for the CART algorithm. The choice of using the Gini Index shows the improvement in the CART algorithm's performance. In contrast, Information Gain is more suitable with other algorithm such as ID3 and C4.5.

This is because Gini Index provides the binary split for each attribute. This is obtained from the computation of Gini Index that split the attributes based on the measurement of the impurity. Meanwhile, Information Gain is the value of difference between the original information requirement and the new requirement. In other word, CART algorithm uses Gini Index because the architecture of the CART algortihm uses the minimum value of Gini Index, while other algorithm uses the maximum value of Information Gain.

Other than that, Gini Index makes the CART algorithm run faster because it is mathematically simple and easier to calculate even in large datasets. This is because Gini Index only use squares, not log like Information Gain. Overall, Gini Index leads to better balance and excellent classification accuracy. With the ability to measure node purity successfully, it is suitable for binary splits. Gini Index formula can be seen as equation. Gini Index also be known as Gini Impurity.

$$GiniImpurity = 1 - \sum_{i=1}^{c} P_i^2$$

Where Pi is the probability of an object being classified to a particular class[].

Based on this formula:

- The Gini Impurity can take on values ranging from 0 to 0.5.
- When Gini Impurity is 0, the node is perfectly pure. However, if the node reaches 0.5, the node is most impure node.
- The lower the Gini Impurity, the better the split. This indicates that index are more pure when the value is low.

How to select a Decision Node:

1. For each attributes, the Gini Index is calculated.
2. Then, weighted sum of Gini Indexes is calculated for feature.
3. Select the attributes with purest Gini Index (lowest Gini Index value).
4. Repeat all three steps above until a generalised tree has been created.

In summary, Gini Index measures the likelihood of incorrect classification of an element when both the element and its class are chosen randomly based on the distribution of class probabilities.

3.4.2 Model evaluation

To validate the model's effectiveness, we must deliberately test its classification performance on bot vs. human accounts. This could be done through quantitative metrics and qualitative analysis.

There are various metrices that could be use to assess the performance of the models. For example, accuracy, precision, recall (sensitivity) and F1 score. These metrics could be calculated using this formula.

Metrics formula:

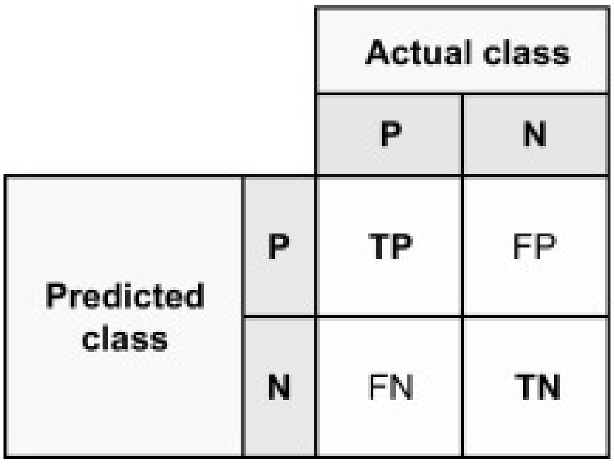
- **Accuracy**: (True Positive + True Negative) / Number of samples
- **Precision**: True Positives / (True Positives + False Positives)
- **Recall**: True Positives / (True Positives + False Negatives)
- **F1 score**: 2 * (Precision * Recall) / (Precision + Recall)

True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) values can be gained by counting the amount of bot and human accounts being identified in this model. These values calculated are called confusion matrix which each term can be identified below.

Confusion Matrix:

- True Positive (TP): Bots account are correctly classified as bots
- True Negative (TN): Human account are correctly classified as human

- False Positive (FP): Human account are wrongly classified as bot
- False Negative (FN): Bot account are wrongly classified as human



		Actual class	
		P	N
Predicted class	P	TP	FP
	N	FN	TN

Figure 3.2: Confusion Matrix representation.

In conclusion, model evaluation helps us to see how accurate our CART algorithm identifies bot accounts on Mastodon. This technique gives insight for the hyperparameter tuning to improve performance before deployment.

4 Implementation

In this project, we divided implementation into 3 main task which is setting up Mastodon on a private server, extracting dataset from PostgreSQL and implementing the bot detection models. The project will be conducted on CSE/168X with cyberlab05 computer selected as a server. Mastodon account creation can only be done in Linux terminal and if any problem occurs during the project, participants are allowed to contact me or my supervisor as a IT services.

4.1 Mastodon setup on a private server

In order to get the dataset of Mastodons' users in a private server, we need to host a Mastodon instance on an Ubuntu server. To build this server, a few requirement needs to be fulfilled. A step-by-step process to setup the Mastodon can be seen in the process below.

Process to set-up a Mastodon server:

1. **Preparing the environment**. To prepare the server, log in as user with sudo privileges. Then, enable and configure the firewall to allow communication within network.

2. **Install required software**. This includes APT dependencies and Node.js dependencies

3. **Set up PostgreSQL.**

4. **Install Ruby and rbenv**. Install Ruby using rbenv

5. **Download and Configure Mastodon**. Clone the Mastodon repository to the server into new directory. Now, Ruby dependencies can be install. Then, install the Mastodon's JavaScript dependencies.

6. **Run Mastodon setup** In this study, local DNS name was set up to 'https://cyberlab05.cyber.local/'. The email address will be in format username@cyberlab05.cyber.local.

7. **Configure systemd services.**

8. **Configure Nginx for Internal Domain**. We set our server to the IP address "192.168.100.105".

9. **Final check**. To check if the Mastodon is working, we visited the website of Mastodon's instance "https://cyberlab05.cyber.local/". Then, we try to use Linux terminal to register as a user in Mastodon.

4.2 Bot in Mastodon

Based on the capability of the machine that running the server, engagement bots are selected to be deployed into the Mastodon instance. This bots perform activities like bookmark, favourite and reblog which are similar to bookmark,likes and retweet for Twitter/X. The original idea is to implement spam bot but the machine needs to keep on running to make sure it can detect recently tooted within the live feeds. To combat this issue, engagement bots is created, which also carry almost the same problem to the Mastodon user. For example, engagement bot make a presence within the social media by bookmark, favourite and reblog other users to influence more people into their malicious act.

To create a engagement bot in Mastodon, Python programming language is used to connect to a Mastodon account using the Mastodon.py. Before we could create a bot using Python, these steps needs to be completed so that bot created can be approved.

Creating a bot in Mastodon:

1. **API credentials.** Create one account that will be a bot. Navigate to Preference > Development > New Application. Fill in application name and select all the scopes. Once applied, client_id, client_secret and access_token will be shown on New Application.

2. **Write the bot script.** Bot script file should consist of accountid.py and main.py. Accountid.py are generally use to get the account id of the bot account, while main.py is the file that will automate actions towards his following lists.

4.3 Data extraction

PostgreSQL is a highly capable, open source object-relational database system that is freely available to use. It builds on standard SQL while adding advanced functionality to handle complex data storage and scaling needs securely. This system is used in the project to store database of Mastodon instances. By using data in PostgreSQL, models are able to perform bot detection.

The process of extracting data can be split into these steps:

1. **Exporting a PostgreSQL Database to .sql file.** First, check if the pg_dump command is installed in the machine. Typically, this command was included when PostgreSQL was installed. Then, use basic syntax of exporting the database from username postgres in mastodon database. This will output mastodon.sql file.

2. **Transfer .sql file to another device** File extracted from the database is transferred to another device to allow more control to .sql file.

3. **Convert .sql file to .csv file.** Once the .sql file is in another computer, internet access is needed to convert .sql file to .csv file through website rebasedata.

4. **Delete unrelated tables in the database.** After examining the tables in database, there are so many empty tables that does not needed for this experiment. For example, public.relays, public.blocks, public.ip_blocks and more.

5. **Data cleaning.** Clean up the entries by replacing, removing or reformat the element in the table.

After extracting the data into the CSV file, tables need to be observed repeatedly. Some tables that are not relevant and does not have any values should be removed and ignored. Also, since sql file are dump file from PostgreSQL, insertion between tables should be done so that reading a file will be easier. Since this project focuses on detecting engagement bot, important feature such as favourite and bookmarks are collected and kept in on file so that we could observe the behaviour of bots with these two feature.

Header of dataset after data preprocessing:

- **account_id**
- **username**

- **bot_boolean**
- **status_id_favourite**
- **status_id_bookmarks**
- **created_date**
- **favourites_per_day**
- **bookmarks_per_day**

4.4 Bot detection model

The models use the CART algorithm to detect bots in Mastodon private server. During the experiment we collected over 17 accounts in Mastodon while 2 of them are labeled as bots. In dataset, we accumulate 17 rows and 8 columns after combining 3 table which is public.bookmarks, public.accounts and public.favourite.

In Jupyter Notebook, where we implement the CART model, the detection bot algorithm was implemented based on these main steps:

1. **Import libraries.** For data visualisation and handling, pandas and matplotlib are used. Meanwhile, train_test_split, DecisionTreeClassifier and classification_report are used for model training and evaluation.
2. **Read CSV file**. Path to read the file could be different on local device. Adjustment are needed in order to read the file correctly.
3. **Adjust column name.** Since first row is used for the column header in CSV file, it need to be remove from the data.
4. **Train-Test Split.** Split the data into training and testing sets with 70:30 ratio respectively.
5. **Train CART model.** A decision tree is trained on the training data.
6. **Prediction and Evaluation** Model are predicted based on test set. Then, classification metrics values such as precision, recall, f1-score are printed in order to give the baseline of model performance.
7. **Visualisation.** A diagram of a decision tree with gini index, samples, value and class is generated.

Upon completing the CART algorithm training and testing process, decision tree can be seen as figure4.1.

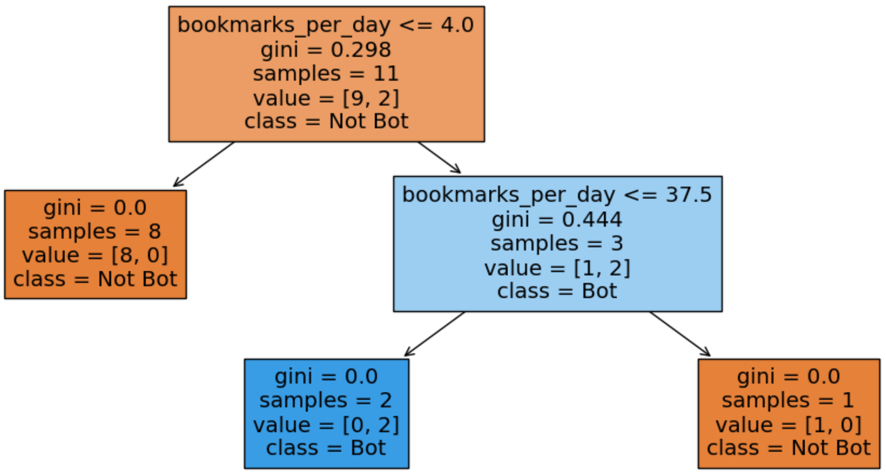


Figure 4.1: Decision Tree for Mastodon dataset.

5 Result and Evaluation

Throughout the training process, results of the bot detection model shows a positive outcome. However, we did make a slight adjustments—such as altering the random state value to observe how these changes would impact its performance. Suprisingly, it shows a significant changes in precision value. Despite that, this result is not desirable because this whole experiment does not replicate the real situation of Mastodon in Internet.

5.1 Test and Result

During implementation, the CART model was trained on datasets containing accounts with bot-like activity patterns, based on bookmark and favourite count per day. The dataset contains 17 rows and 8 columns, with a mixture of user identifiers, account activity features, and a label (bot_boolean) to show that whether an account is a bot. After data preprocessing, only 11 rows of data suitable for model training and evaluation. This is due to the split of dataset into 70% training set and 30% testing sets.

Evaluation metrics:

- Accuracy: 66.67%
- Precision: (Class 0 - Not Bot): 80%
- Recall: (Class 0 - Not Bot): 100
- F1-score: (Class 0): 89%
- Precision: (Class 1 - Bot): 100%
- Recall: (Class 1 - Bot): 50
- F1-score: (Class 1): 67%

Meanwhile, training set score is 100% and test set score is 83.3%. Although the training set score seems desirable because it achieve highest value, but it could also mean that overfitting occurs. This will cause poor performance on unseen data.

For confusion matrix, the model correctly predicted 4 human accounts and misclassified 2 cases (one false positive, one false negative). Despite that, no true positive for bot detection.

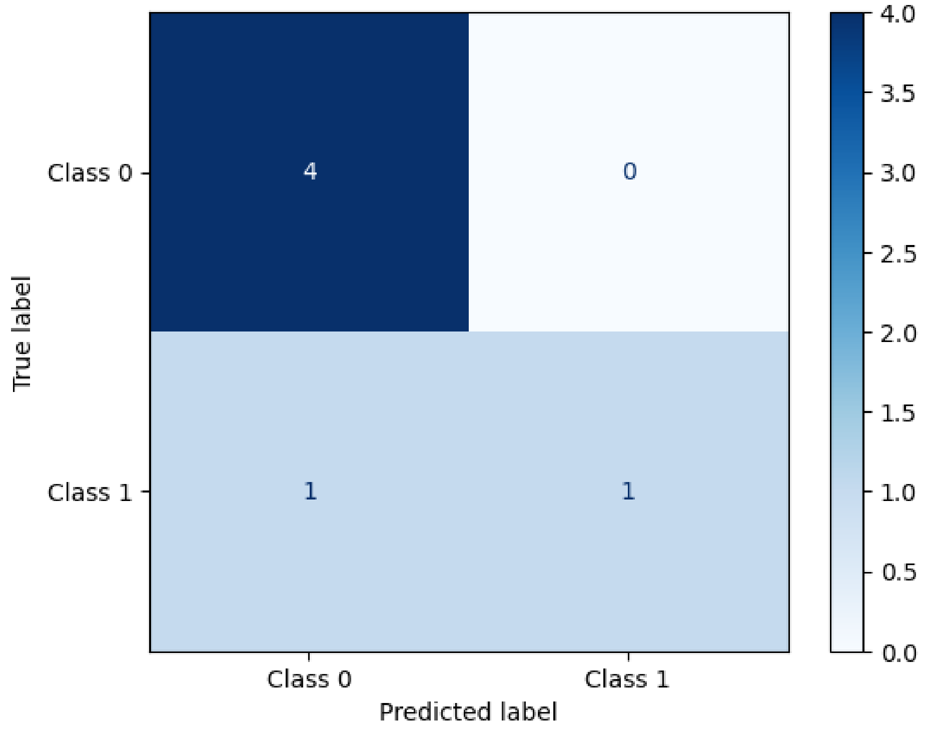


Figure 5.1: Confusion Matrix.

6 Conclusion

This project demonstrated that it is possible to detect bot behavior in decentralised platforms like Mastodon using machine learning models, particularly decision trees. By deploying a private server, we were able to simulate user behavior and ethically collect data without impacting the wider network. The application of the CART algorithm yielded useful insights into bot patterns, despite performance constraints due to a small dataset.

However, several limitations surfaced during the study. The dataset was relatively small, and the private server environment could not fully replicate the complexities of the public Mastodon ecosystem. Additionally, real-world bots often display more diverse and adaptive behaviors than those simulated in this controlled setting. These factors limited the model's generalizability and point to the need for further testing with larger and more representative datasets.

Results show that bot detection models in Mastodon can be adapted to decentralised social networks to some extent. This is due to the unique architecture of Mastodon, particularly its federated moderation. As Mastodon and other decentralized networks grow in popularity, tools that adapt to their unique architectures will be critical. Bots in decentralised social networks can have a specific behaviour based on the instance it resides. Despite the solutions to label a bot when a bot is deployed in a server, this study proves that bot can still act as a real account without any label.

In summary, this study provides a foundational step in that direction and opens up future opportunities to enhance detection accuracy using more scalable and dynamic approaches.

Bibliography

[1] C. Shao, A. Flammini, O. Varol and F. Menczer, 'The spread of low credibility content by social bots,' 2018, Unpublished or journal unspecified.

[2] E. Ferrara, O. Varol, F. Menczer and A. Flammini, 'The rise of social bots,' 2016, Unpublished or journal unspecified.

[3] A. Bessi and E. Ferrara, 'Social bots distort the 2016 u.s presidential election online discussion,' 2016, Unpublished or journal unspecified.

[4] R. R. Rout, G. Lingam and Somayajulu, 'Detection of malicious social bots using learning automata with url features in twitter network,' Conference name not specified, 2020.

[5] R. S. Kunwar and P. Sharma, 'Social media: A new vector for cyber attack,' in *IEEE*, 2016.

[6] S. Kudugunta and E. Ferrara, 'Deep neural networks for bot detection,' *Information Science*, vol. 467, pp. 312–322, 2018.

[7] E. D. Cristofaro, A. Raman, S. Joglekar, G. Tyson and N. Sastry, 'Challenges in the decentralised web: The mastodon case,' in *Proceedings of the Internet Measurement Conference*, 2019, pp. 217–229.

[8] M. Zignani, S. Gaito and G. P. Rossi, 'Follow the "mastodon": Structure and evolution of a decentralized online social network,' in *Proceedings of the International AAAI Conference on Web and Social Media*, Milan, 2018.

[9] Brandur. 'Brandurr.org.' [Online; accessed 12-January-2021]. (Jan. 2021), [Online]. Available: https://brandur.org/fragments/mastodon-cross-posting.

[10] Mastodon. 'What is mastodon?' [Online; accessed 14-January-2024]. (Jan. 2024), [Online]. Available: https://docs.joinmastodon.org/.

[11] T. Palpanas, S. Bortoli and P. Bouquet, 'Decentralised social network management,' *International Journal of Web Based Communities*, vol. 7, no. 3, 2011.

[12] N. Narayan, 'Twitter bot detection using machine learning algorithms,' in *International Conference on Electronics, Communication and Computing Technologies (ICECCT)*, 2021.

[13] S. Barhate, R. Mangla, D. Panjwani, S. Gatkal and F. Kazi, 'Twitter bot detection and their influence in hashtag manipulation,' in *Centre of Excellence in Complex and Nonlinear Dynamical Systems*, Mumbai, 2020.

[14] A. M. M. Flores, E. Pilipets and S. P. Caldeira, ''24/7 horny&training': Porn bots, authenticity, and social automation on instagram,' 2024, Journal not specified.

[15] K. Lancaster, 'Non-consensual personified sexbots: An intrinsic wrong,' vol. 23, pp. 589–600, 2021, Journal name not provided.

[16] M. Zignani, C. Quadri, S. Gaito, H. Cherif and G. P. Rossi, 'The footprints of a "mastodon": How a decentralized architecture influences online social relationships,' 2019.

[17] L. L. Cava, S. Greco and A. Tagarelli, 'Understanding the growth of the fediverse,' 2021.

[18] S. d. Mönnink, 'From network to platform to protocol: Mastodon's ethos and the sociotechnical imaginary of the fediverse,' 2024.

[19] L. Hagen, S. Neely, T. E. Keller, R. Scharf and F. E. Vasquez, 'Rise of the machines? examining the influence of social bots on a political discussion network,' *Sage Journal*, 2022.

[20] Z. Gilani, R. Farahbakhsh and J. Crowcroft, 'Do bots impact twitter activity?' In *Proceedings of the 26th International Conference on World Wide Web Companion*, Perth, 2017.

[21] F. Zeng, Y. Sun and Y. Li, 'Mrlbot: Multi-dimensional representation learning for social media bot detection,' 2023.

[22] A. A. Daya, M. A. Salahuddin, N. Limam and R. Boutaba, 'Botchase: Graph-based bot detection,' *IEEE Transactions on Network and Service Management*, vol. 17, 2020.

[23] K. Radivojevic, C. McAleer, C. Conley, C. Kennedy and P. Brenner, 'Social media bot policies,' 2024.

[24] O. Varol, E. Ferrara, C. A. Davis, F. Menczer and A. Flammini, 'Online human-bot interactions: Detection, estimation, and characterization,' 2017.

[25] M. Moore, 'Fake accounts on social media, epistemic uncertainty and the need for an independent auditing of accounts,' *Internet Policy Review*, vol. 12, no. 1, 2021.

[26] A. H. Wang, *Detecting spam bots in online social*.

[27] S. A. Alhosseini, P. Najaf, R. B. Tareaf and C. Meinel, 'Detect me if you can: Spam bot detection using inductive,' 2019.

[28] C. Zhao, Y. Xin, X. Li, H. Zhu, Y. Yang and Y. Chen, 'An attention-based graph neural network for spam bot detection in social networks,' vol. 10, no. 2, 2020.

[29] R. J. Oentaryo, A. Murdopo, P. K. Prasetyo and E.-P. Lim, ‘On profiling bots in social media,’ 2016.

[30] I. Mbona and J. H. Eloff, ‘Classifying social media bots as malicious or benign using semi-supervised machine learning,’ *Journal of Cybersecurity*, vol. 9, no. 1, 2022.

[31] T. Zaman and K. Qureshi, ‘Social media engagement and cryptocurrency performance,’ 2023.

[32] I. H. Anaobi, A. Raman, I. Castro, H. B. Zia, D. Ibosiola and G. Tyson, ‘Will admins cope? decentralized moderation in the fediverse,’ 2023.

[33] D. Zulli, M. Liu and R. Gehl, ‘Rethinking the “social” in “social media”: Insights into topology, abstraction, and scale on the mastodon social network,’ 2020.

[34] A. Ramalingaiah, S. Hussaini and S. Chaudhari, ‘Twitter bot detection using supervised machine learning,’ India, 2021.

[35] A. Karataş and S. Şahin, ‘A review on social bot detection techniques and,’ 2017.

[36] J. VELASCO-MATA, V. G. CASTRO, E. F. FERNÁNDEZ and E. ALEGRE, ‘Efficient detection of botnet traffic by features selection and decision trees,’ 2021.

[37] W. Eberle and L. Holder, ‘Discovering structural anomalies in graph-based data,’ *IEEE*, 2024.

[38] Y. Feng, J. Li, L. Jiao and X. Wu, ‘Towards learning-based, content-agnostic,’ 2021.

[39] P. T. R, ‘A comparative study on decision tree and random forest using r tool,’ *International Journal of Advanced Research in Computer and Communication Engineering*, 2015.

[40] S. Abu-Nimeh, D. Nappa, X. Wang and S. Nair, ‘A comparison of machine learning techniques for,’ 2007.

[41] J. Wu, P. Guo, Y. Cheng, H. Zhu, X.-B. Wang and X. Shao, ‘Ensemble generalized multiclass support-vector-machine-based health evaluation of complex degradation systems,’

[42] D. M. Abdullah and A. M. Abdulazeez, ‘Machine learning applications based on svm classification: A review,’ May 2021.

[43] K.-C. Lin, S.-Y. Chen and J. C. Hung, ‘Botnet detection using support vector machines with artificial fish swarm algorithm,’ *Journal of Applied Mathematics*, 2014.

[44] M. V. Kotpalliwar and R. Wajgi, ‘Classification of attacks using support vector machine (svm) on kddcup’99 ids database,’ in *Fifth International Conference on Communication Systems and Network Technologies*, 2015.

[45] G. I. Webb, L. Liu, X. Ma and S. Chen, ‘A novel selective naïve bayes algorithm,’ 2020.

[46] K. Vembandasamy, R. and E., ‘Heart diseases detection using naive bayes algorithm,’ *International Journal of Innovative Science, Engineering Technology*, vol. 2, no. 9, 2015.

[47] N. F. Rusland, N. Wahid, S. Kasim and H. Hafit, ‘Analysis of naïve bayes algorithm for email spam filtering across multiple datasets,’ in *IOP Conference Series: Materials Science and Engineering*, Melaka, 2019.

[48] S. Smys and W. Haoxiang, ‘Naïve bayes and entropy based analysis and classification of humans and chat bots,’ *Journal of ISMAC*, vol. 3, no. 1, pp. 40–49, Apr. 2021.

[49] M. U. Khan, S. Mohammad, L. Liu, M. Shardlow, R. Nawaz and M. Ali, ‘Bot detection using a single post on social media,’ in *World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, 2019.

[50] S. D. Jadhav and H. P. Channe, ‘Comparative study of k-nn, naive bayes and decision tree classification techniques,’ *International Journal of Science and Research (IJSR)*, 2016.

[51] D. R. Thareja and K. Yadav, ‘Comparing the performance of naive bayes and decision tree classification using r,’ 2019.

[52] B. Gupta, A. Arora, A. Rawat, N. Dhami and A. Jain, ‘Analysis of various decision tree algorithms for classification in data mining,’ *International Journal of Computer Applications*, vol. 168, no. 8, 2017.

[53] GeeksforGeeks, *Cart (classification and regression tree) in machine learning*, https://www.geeksforgeeks.org/cart-classification-and-regression-tree-in-machine-learning/, [Accessed 27 April 2025], Sep. 2024.

[54] F. Sajedi-Hosseini, M. Golshan, J. Adamowski, E. Moradi and B. Choubin, ‘An ensemble prediction of flood susceptibility using multivariate discriminant analysis, classification and regression trees, and support vector machines,’ 2018.

[55] J. Velasco-Mata, V. González-Castro, E. F. Fernández and E. Alegre, ‘Efficient detection of botnet traffic by features selection and decision trees,’ in *International Conference on Networking and Information Technology*, 2010.