Fall 2021

Identifying Bots on Twitter with Benford's Law

Sanmesh Bhosale
San Jose State University

Identifying Bots on Twitter with Benford's Law

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Sanmesh Bhosale

Dec 2021

The Designated Project Committee Approves the Project Titled

Identifying Bots on Twitter with Benford's Law

Sanmesh Bhosale

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

May 2021

Prof. Fabio Di Troia	Department of Computer Science
Dr. Navrati Saxena	Department of Computer Science
Mahesh Jaliminche	SDE at Amazon

ABSTRACT

Over time Online Social Networks (OSNs) have grown exponentially in terms of active users and have now become an influential factor in the formation of public opinions. Due to this, the use of bots and botnets for spreading misinformation on OSNs has become a widespread concern. The biggest example of this was during the 2016 American Presidential Elections, where Russian bots on Twitter pumped out fake news to influence the election results.

Identifying bots and botnets on Twitter is not just based on visual analysis and can require complex statistical methods to score a profile based on multiple features and compute a result. Benford's Law or the Law of Anomalous Numbers states that in any naturally occurring sequence of numbers, the first significant leading digit frequency follows a particular pattern such that they are unevenly distributed and reducing. This principle can be applied to the first-degree egocentric network of a Twitter profile to assess its conformity to Benford's Law and classify it as a bot profile or normal profile.

This project focuses on leveraging Benford's Law in combination with various Machine Learning (ML) classifiers to identify bot profiles on Twitter. In addition, the project also discusses various statistical methods that are used to verify the classification results.

Keywords **– Benford's Law, Twitter, Machine Learning, Social Bots**

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my mentor and project advisor Prof. Fabio Di Troia for guiding me and providing valuable suggestions in the project. He encouraged me throughout the course of the project with constant feedback and support to fulfil the requirements.

I would also like to thank my committee members Dr. Navrati Saxena and Mahesh Jaliminche for their time and valuable guidance on completion of the project.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. Introduction

Online Social Networks (OSNs) or Social Media Platforms (SMPs) as we know them have accumulated millions of users worldwide [9]. With the exponential growth in the number of accounts and active users on SMPs, it is becoming harder and harder to moderate the content and account activities. While a genuine user and malicious user are being considered in this scenario, we also need to consider informational bots and malicious bots. OSNs have been plagued with many types of malicious bots in recent years. Twitter is a popular microblogging and social networking service with millions of users worldwide. Twitter account holders have the option to follow other accounts i.e., make friends, each account can have any number of accounts following them i.e., followers, and each account can post status updates with a character limit of 280 characters in the form of tweets. Twitter has gained popularity due to its adaptation by numerous influential figures and regular political coverage. These services offered by Twitter have become a target of social media bots for spreading fake and malicious content online. One of the biggest examples of bots spreading fake news and malicious misinformation was during the 2016 presidential elections where Russian bots tried to interfere in the election. Since then, Twitter has taken numerous measures for content moderation by suspending suspicious accounts that spread misinformation and flagging baseless or questionable tweets.

In context of the OSNs a social bot or a suspicious user account is a computer algorithm or script that will automatically interact with other accounts and produce content without human input or intervention. There are different types of bots or sybil accounts, but we will only consider two scenarios where either a bot is malicious i.e., it violates the Twitter community guidelines, or it is an informational bot which is not involved in malicious

activities. There can also be levels of bots i.e., fully automated bots, partially automated bots, and hacked real user accounts for malicious activities. Bots or suspicious accounts participate in activities that can seriously harm the integrity of online communities. Previously, there have been numerous studies which tackle the social bots on Twitter with the help of ML techniques. There are also some real-time Twitter bot detection platforms like Bot Sentinel and BotOrNot, but they have their limitations. Due to these efforts to tackle bots, the bot accounts have started changing their patterns and they are now able to better camouflage themselves such that previous methods are not enough to identify them [8]. This project focuses on identifying these camouflaged bots with the help of Benford's Law, Machine Learning (ML) classifiers, and Statistical Analysis.

The Benford's Law or Newcomb-Benford's Law states that in any naturally occurring sequence of numbers, the First Significant Leading Digit (FSLD) frequencies follow a particular pattern such that they are unevenly distributed and reducing in nature [1], [2]. The astrologer Simon Newcomb in 1881 first observed that the logarithmic tables in the library had their initial pages more worn out and dirtier than the latter ones [1]. He concluded that the initial digits are more commonly to appear or used than the latter digits. Physicist Frank Benford re-discovered this lost phenomenon after 50 years and later published a paper titled "The Law of Anomalous Numbers" [2]. For experimentation he researched on 20 sets of naturally occurring sequences with more than 20,000 samples which included data from sources like river areas, population, newspapers, addresses, and death rates [2]. The different dataset tested by him followed the Benford's Law and can be calculated with P predicted for any digit d can be obtained by using the following formula:

$$\mathbf{P(d) = \log_{10}(1 + 1 / d)}$$

Benford's Law is not obeyed by all datasets, there are certain conditions that a dataset must fulfil to follow it [18]. Let use compare a few general conditions and compare Twitter datasets with them below:

- All digits from 1 to 9 should occur in leading position: In our Twitter datasets when we consider following_counts all digits from 1 to 9 can be possible FSLDs.
- There should be more smaller numbers than large numbers: In our Twitter datasets when we consider status_counts the small numbers are more likely to occur than larger numbers.
- The dataset should be natural: Twitter relationships where users follow each other should form organically. There are botnets which will follow a particular user to inflate their followers_counts when paid for the service.
- There should be no sequence in numbers: Every individual Twitter account has different number of status_counts, following_counts, and followers_counts.
- No predefined boundaries: Twitter has no maximum or minimum number set for the parameters like favorite_counts, likes_counts, and status_counts.
- Orders of magnitude: Twitter has numbers in tens, hundreds, thousands, and even in millions so this condition is satisfied.
- Dataset should be large: Twitter has millions of users, so a large dataset of users is accessible for research.

Table 1: Benford's Distribution FSLD Frequencies [2]

Digit:	**Frequency: (%)**
1	30.103
2	17.609
3	12.494
4	9.691
5	7.918
6	6.695
7	5.799
8	5.115
9	4.576

The experimental findings of Prof. Jennifer Golbeck in [3], [4] and Lale Madahali & Margeret Hall in [10] have paved the way for the use of Benford's Law on the first-degree egocentric network of any social media profile for its Benford Analysis. It has been experimentally proved that first significant leading digits of friend counts of a social media account follow the Benford's distribution given above [3], [4]. If any account doesn't follow the Benford's distribution it can be a suspicious account or malicious bot.

In Chapter 2, we review the background with various approaches and machine learning techniques used in the past for bot detection on Twitter with the help of multiple research papers, journals, and articles. In Chapter 3, we go over the methodology, experimental setup, and datasets used to implement the project. In Chapter 4, we discuss the results and observations of our experiments. In Chapter 5, the conclusion of our work is presented, and any possible future scope of the project is explored with clarification.

2. Background

In this chapter, we discuss about the background of other related works in the field of social bot detection on twitter with and without application of Benford's Law. We also show some of the drawbacks of previous works, which make it difficult to tackle the problem of bot detection. In addition, this chapter also discusses various classification techniques used in this project.

2.1 Related Work

This section discusses the previous works of social bot detection on Twitter and analyzes their performance and drawbacks. Twitter was launched in 2006 merely as a simple SMS mobile app but has grown into a full-fledged communication platform. Most of the previous works tackle the problem of social bot detection with supervised machine learning [12]. The main issue to be addressed here is that there is no standard definition of a social bot. Hence, the labelled datasets used to train the classifier are created by researchers after manual analysis which can have human error and bias.

2.1.1 Botometer Service

The Botometer service was formerly known as BotOrNot service. It is a popular publicly available bot detection tool which gives out a real-time social bot score for Twitter accounts [14]. The BotOrNot service was released in May 2014, and it has been developed by researchers from Indiana University at Bloomington. The service is based on a supervised machine learning classifier which leverages over 1,000 features of the target account to produce a classification score also called as the social bot score. According to the algorithm of Botometer, the higher the social bot score is the more likely that target account is being

controlled by a software. To get the features required by the classifier the target account's 200 most recent tweets and 100 recent mentions from other users are required. Its features can be grouped into six main classes: Network, User, Friends, Temporal, Content, and Sentiment. The classifier has been trained on 15k manually verified bots and 16k human accounts with millions of tweets. It uses Random Forest classifier which is an ensemble supervised machine learning technique to run seven classifiers (one for each feature and one for overall score). Since, some features are based on English language the social bot score is for accounts in English language. Botometer is accessible through both a web interface (botometer.org) and an API endpoint. Botometer service does not have a browser plugin and requires Twitter authentication and permissions.

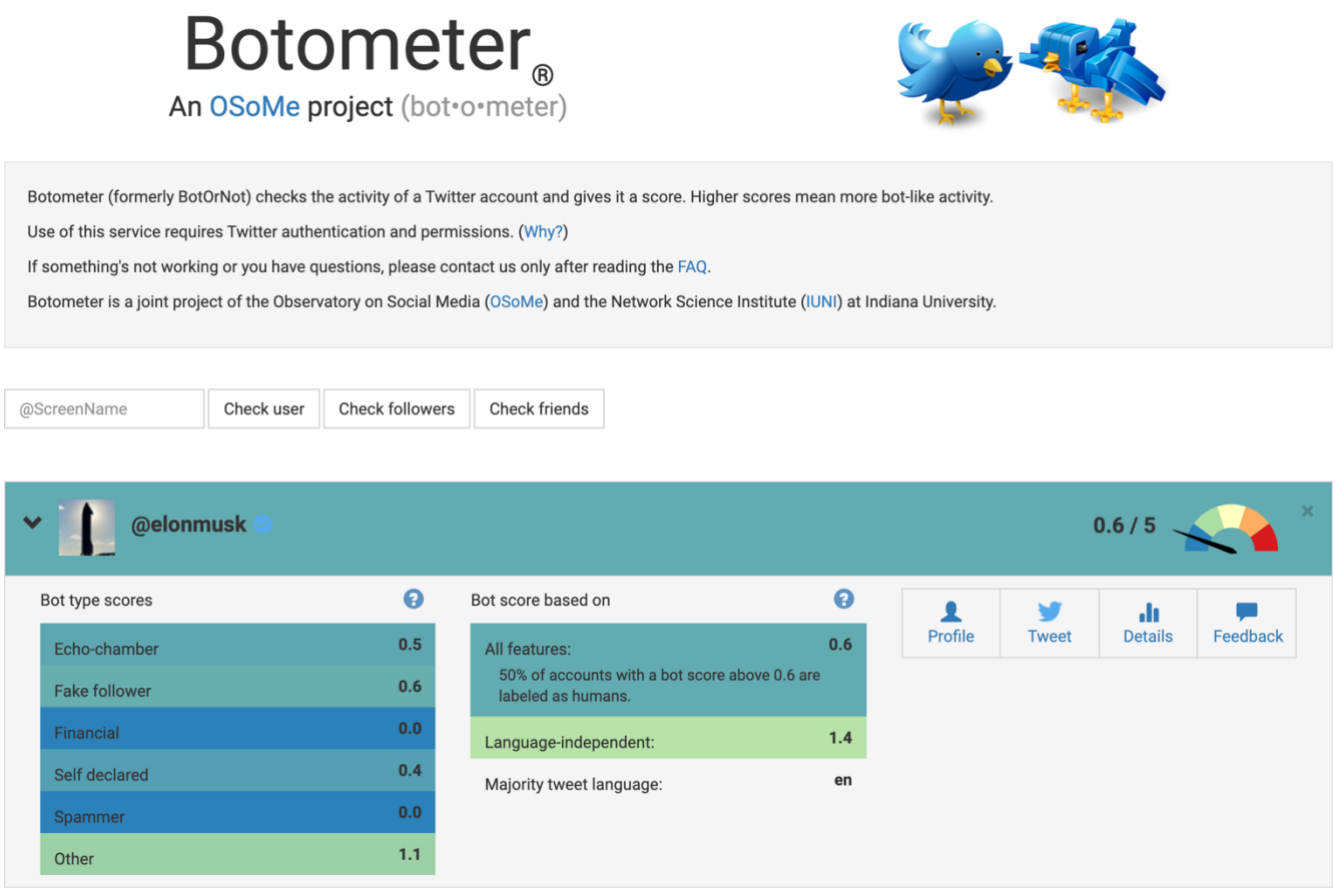

Fig.1 Botometer Service Bot Score [14]

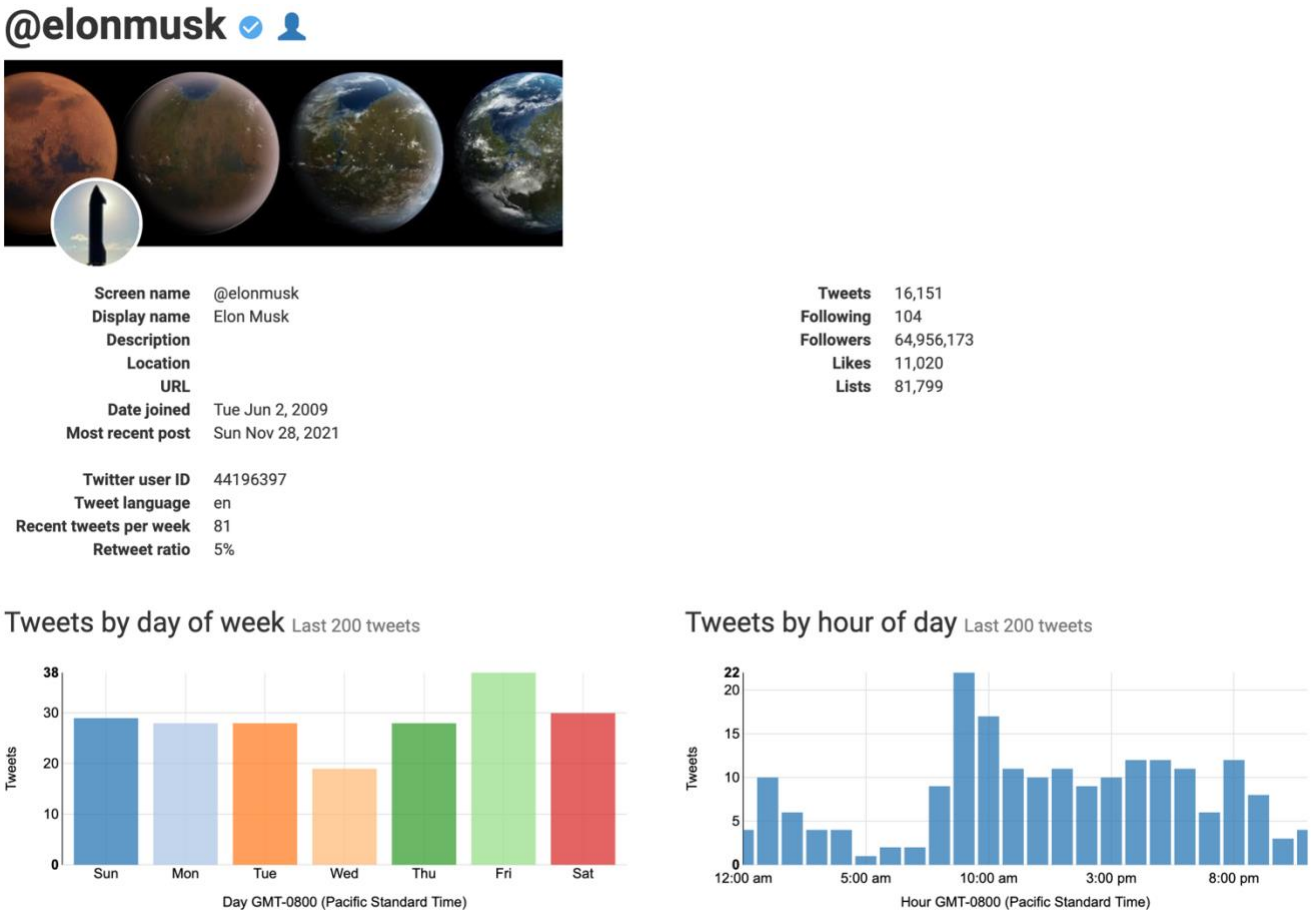

Fig.2 Botometer Detailed Profile Analysis [14]

2.1.2 Human, Bot, or Cyborg?

The researchers Chu et al. have designed a supervised machine learning classifier to distinguish a target account into three different groups: human, bot, and cyborg [16]. An account classified as human is said to have no automated activity, whereas an account classified as bot will be fully automated. An account with a mix of automated and non-automated activity is classified as a cyborg. Their classifier is based on four components: entropy, machine learning, account properties, and decision maker. The entropy

component is used to detect automation by detecting a periodic or regular timing for tweeting. The machine learning component is based on a Bayesian classifier to identify text patterns of social spambots on Twitter. The account properties component analyses account information to differentiate humans from bots. Finally, the decision maker component employs Linear Discriminant Analysis on the features shortlisted by other three components to make a classification. The researchers collected their data by crawling on Twitter using the Twitter API and found that data constitutes of 53% human, 36% cyborg, and 11% bot accounts. For the creation of ground truth, the researchers chose random samples from collected data and classified them manually by going through their homepages and user logs. For the training set each class of humans, cyborgs, and bots have 1000 samples and the classifier is trained on total three thousand samples. The test set is also created with the same method and contains three thousand samples. The researchers have used a very small dataset for the training of the classifier and have changed a binary classification problem into multi-class classification problem by introducing cyborgs. As the results show, their classifier makes no mistakes in identifying humans and bots apart but gets confused between human and cyborg accounts or bot and cyborg accounts.

		Classified			Total	True Pos.%
		Human	Cyborg	Bot		
Actual	Human	949	51	0	1000	94.90%
	Cyborg	98	828	74	1000	82.80%
	Bot	0	63	937	1000	93.70%

Fig.3 Confusion Matrix on Human, Cyborg, and Bot Classification [16]

2.1.3 Benford's Law Applies to OSNs

Prof. Jennifer Golbeck from University of Maryland College Park was the first to apply Benford's Law on the data from OSNs in 2015 [3]. The author experimented with five major OSNs namely: Facebook, Google Plus, LiveJournal, Pinterest, and Twitter. They were able to discover that certain features of OSNs like the friend's following_counts conformed to Benford's Law i.e., Benford's Law was applicable to the first-degree egocentric networks of a target profile. The research findings on Twitter dataset indicate that accounts which strongly deviated from Benford's Law were engaged in malicious or unnatural behavior. This Twitter dataset used for analysis of Benford's Law has been made public by the author and can be accessed at https://github.com/jgolbeck/BenfordData.

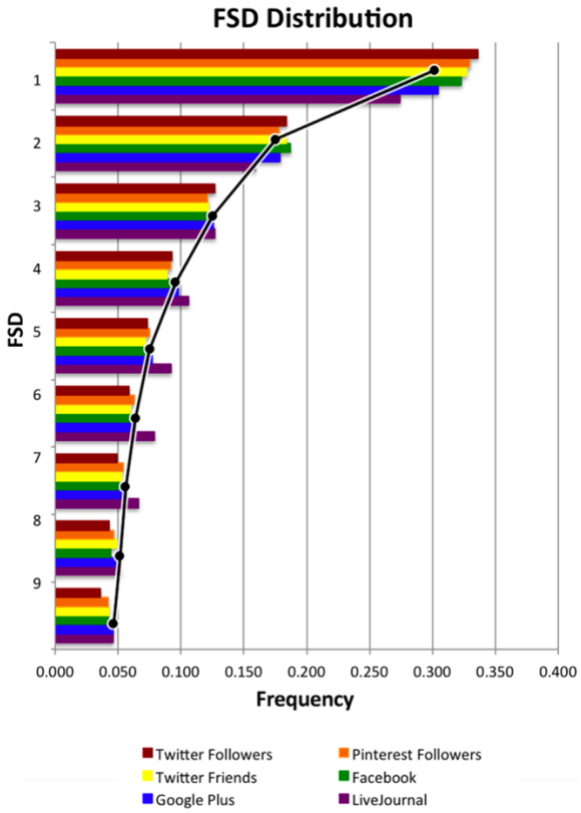

Fig.4 FSLDs for Twitter, Google Plus, Pinterest, Facebook [3]

This discovery from [3] led the author to test a hypothesis that the social connections made by bots are unnatural in nature and they tend to violate Benford's Law [4]. The author reinvestigated the previously discovered Russian bot accounts from 2015 and uncovered a larger Russian botnet with about 13,609 Twitter accounts out of which 99.6 percent did not conform to Benford's Law. This study concluded that first significant leading digits of a friend's following_counts can be utilized to identify anomalous behavior of malicious bots and it is a significant feature to differentiate between humans and malicious bots. Unfortunately, the author has not made the Russian botnet dataset used in this research public.

2.2 Machine Learning Techniques

In this section, the different machine learning techniques used in the project have been discussed in detail. We have experimented using different techniques like Logistic Regression, Naïve Bayes, Support Vector Machine, Random Forest, AdaBoost, and MLP and evaluated the models with Confusion Matrix, Accuracy, Precision, Recall, F-Measure, and AUC-ROC Curve. We have also validated our classification results with statistical tests like Pearson's chi-squared test, Kolmogorov-Smirnov test, Mean Absolute Deviation (MAD).

2.2.1 Naïve Bayes

The Naïve Bayes classifier is an efficient and very simple supervised learning model which performs well for many different types of applications [20]. Naïve Bayes works on the assumption that all the features of the model are independent and do not have any correlation. It is based on the Bayes Theorem of probabilities which can be

given as:

$$p(A|B) = \frac{\mathbf{p(A).p(B|A)}}{\mathbf{p(B)}}$$

Fig.5 Bayes' Theorem

Treating all the features as independent helps the naïve bayes algorithm to be very fast but this means speed is preferred over accuracy and it works well with high-dimensional data compared to other complex algorithms.

2.2.2 Logistic Regression

Logistic Regression is one of the basic machine learning models based on a liner classifier with an objective to predict the influence of different features based on the probability of an event [19]. It is a supervised machine learning algorithm that can be used to predict a binary classification problem. It has a complex cost function called as a Sigmoid Function which is very different from the liner function used in linear regression. The sigmoid function is used to map any value into a value between zero and one. To reduce the cost in Logistic Regression Gradient Descent algorithm is employed to optimize the model parameters.

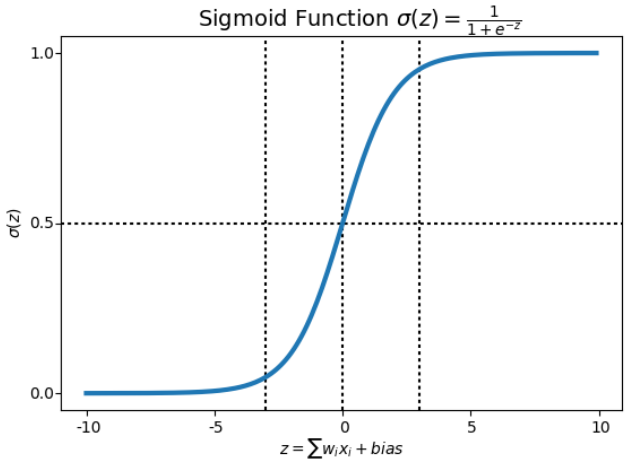

Fig.6 Sigmoid Function Graph

$$f(x) = \frac{1}{1 + e^{-(x)}}$$

Fig.7 Formula of Sigmoid Function

2.2.3 Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for classification as well as for regression [21]. It works well with smaller datasets as processing takes a lot of time. SVM divides the data with the help of a hyperplane which slices the data into different parts. Support vector points are the points closest to the hyperplane and their distance from the hyperplane is called the margin width. SVM has three types of kernels: linear, polynomial, and radial basis function / gaussian. SVM is well suited for binary classification and high dimensional data with more features than training data. Since it has three kernels using the right kernel trick makes all the difference.

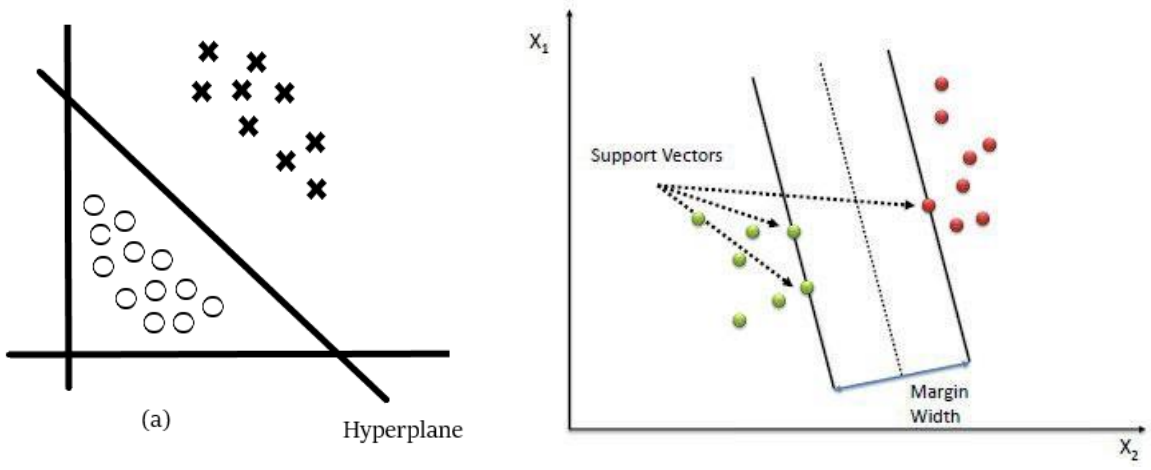

Fig.8 SVM Hyperplane with support vector points and margin width

2.2.4 Random Forest

Random Forest is an ensemble learning technique which is an extension of bagging approach [22]. Random Forest Classifier is a tree-based classifier which consist of individual decision trees that work on ensemble. So, the prediction accuracy increases with the number of trees in the model. With the help of bagging the trees can be trained on different sets of data and can also use different features for making the classification. This helps to create an uncorrelated forest of trees where the nodes are split on random subset of features for each tree. The difference between node splitting of decision trees and random forest model is shown in the diagram below:

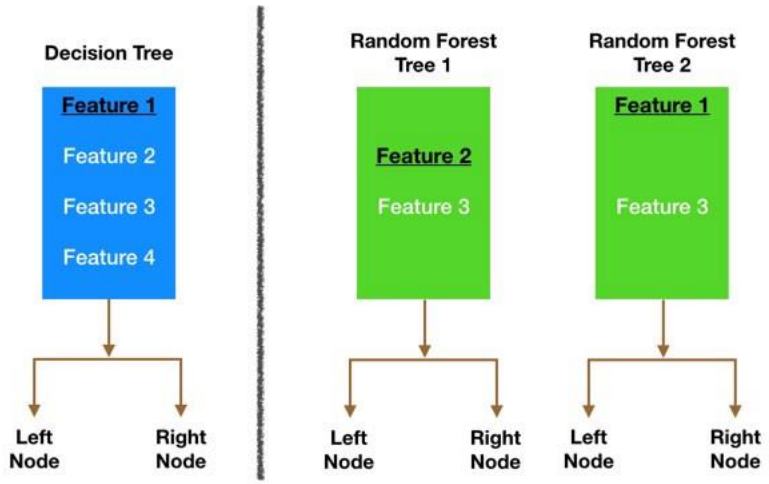

Fig.9 Node splitting on random features in Random Forest vs Decision Trees

2.2.5 AdaBoost

Adaptive Boosting works on the principle of making a high accuracy classifier by using many weak accurate classifiers [23]. It is also an ensemble learning technique and can be used for classification and regression. If each weak classifier satisfies the accuracy

condition of 50 % it will be accepted for aggregation of results. With each round weights are assigned, and the misclassification rate drops down, but it can also lead to overfitting. AdaBoost is widely used in face detection but is also useful as a binary classifier with high accuracy and speed.

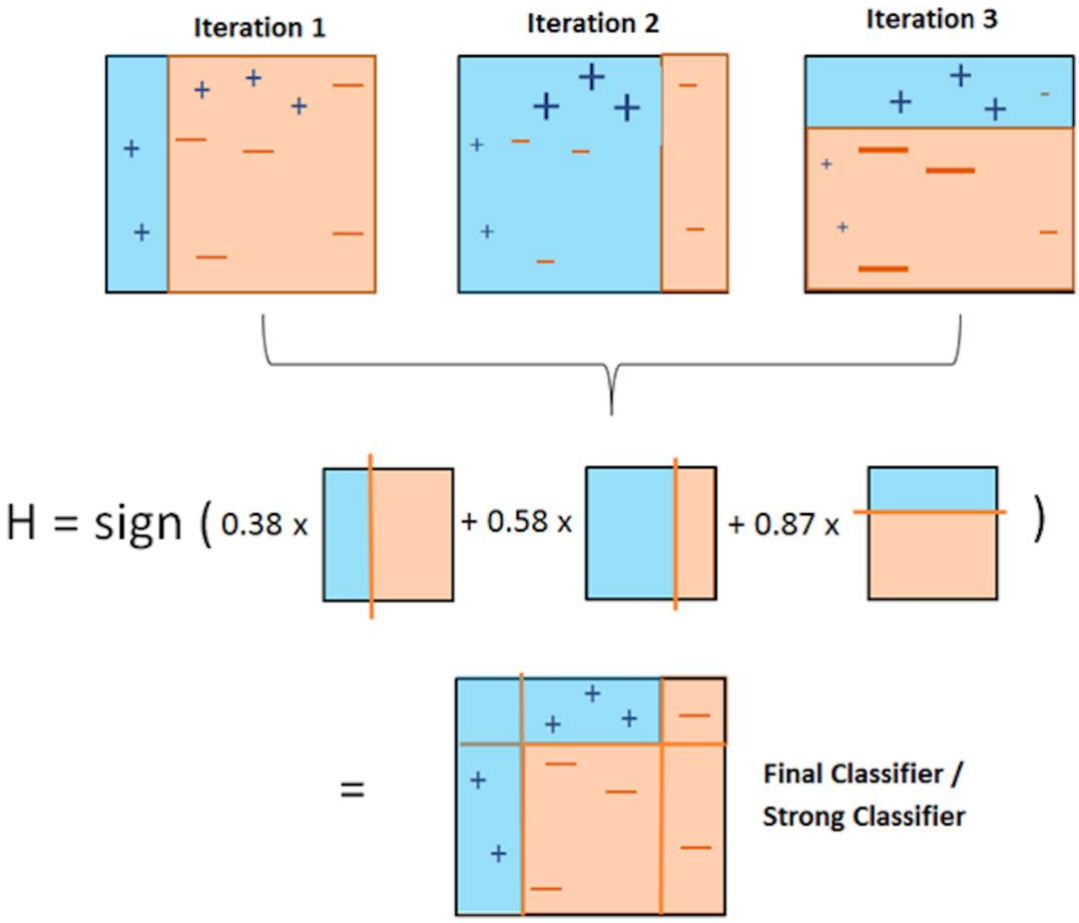

Fig.10 Adaptive Boosting assigning weights and generating final classifier

2.2.6 MLP

Multilayer Perceptron is a feedforward Artificial Neural Network also based on supervised machine learning [24]. The algorithm learns a non-linear function for classification or regression from the given set of features and labels. It is like the Logistic

Regression with the difference that we can have multiple intermediate layers instead of just input and output layers. These intermediate layers are called hidden layers and they contain multiple neurons which use a nonlinear activation function. All the features are the first input layer and for the following layers the input is the previous layer's processed output. MLP uses backpropagation for training, and it is sensitive to feature scaling.

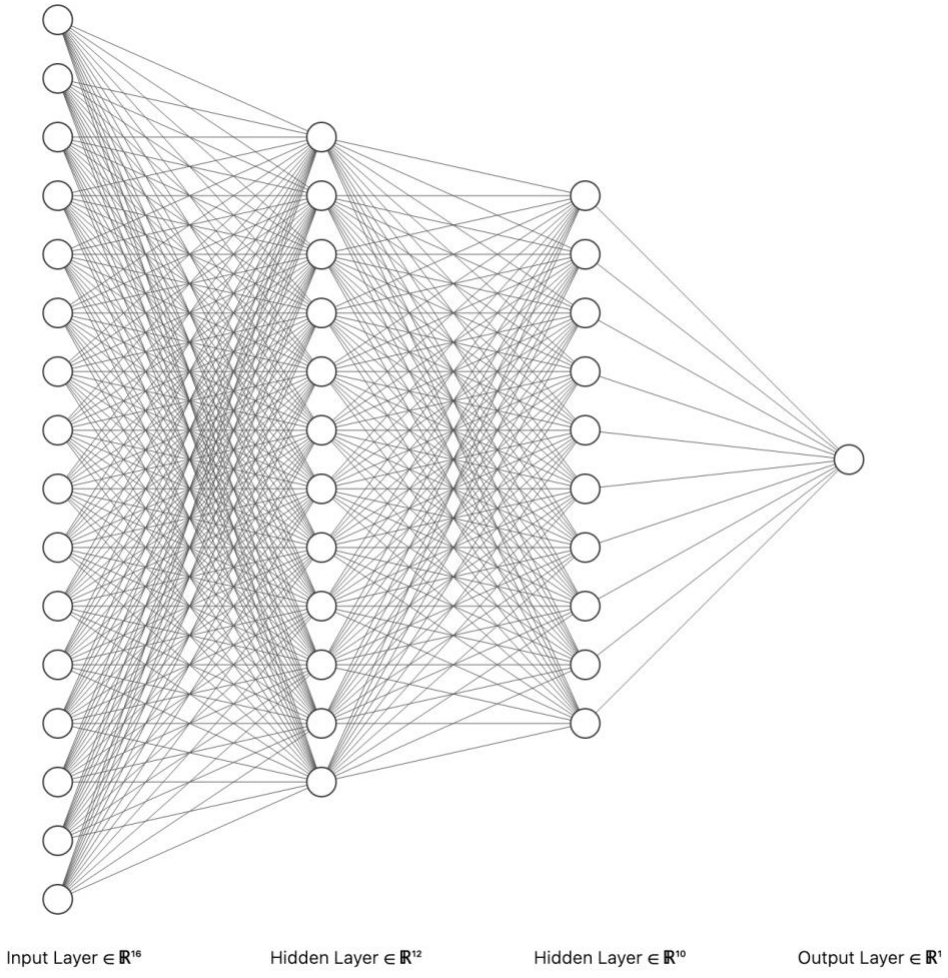

Fig.11 Multilayer Perceptron with 2 hidden layers

2.2.6.1 Activation

The Activation function is also known as the transfer function. There are many different activation function types: Sigmoid, ReLU, Leaky ReLU, and Tanh. These are used to determine the output of the internal hidden layers. In our project we use the default activation function of MLP i.e., ReLU.

2.2.6.2 Optimization

Optimization is the method used to reduce the loss of the neural network by changing parameters like weight and learning rate. There are many different optimizer types: lbfgs, sgd, and adam. The default optimization function is adam but in our project we use lbfgs instead.

2.2.6.3 Regularization

Regularization ensures that the neural network is not overfitting by using penalties while training the model. There is also dropout method where outputs are randomly selected and dropped to reduce overfitting. The alpha is the regularization parameter in the MLP model, and we set the alpha to 0.00001 value.

2.3 Evaluation Metrics

In this section, the different evaluation metrics used in the project have been discussed in detail. Evaluation metrics help us better understand the performance of our machine learning classifier.

2.3.1 Confusion Matrix

Confusion Matrix is a measure of performance of a machine learning classifier [25]. It is useful for calculating Precision, Recall, Accuracy, and AUC-ROC curves. The

matrix is divided into four parts, and we will explain this with our human vs bots example:

- True Positive (TP): the number of bots recognized as bots
- True Negative (TN): the number of humans recognized as humans
- False Positive (FP): the number of humans recognized as bots
- False Negative (FN): the number of bots recognized as humans

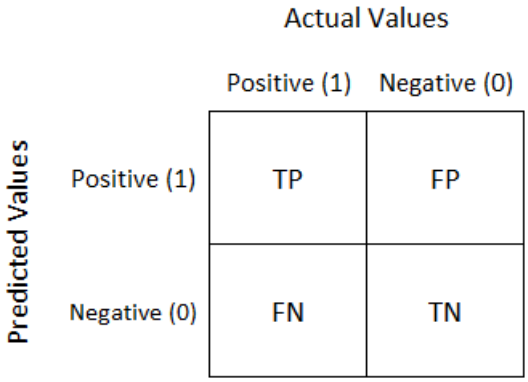

Fig.12 Evaluation Metric: Confusion Matrix

2.3.2 Accuracy

Accuracy is the proportion of true results by the total number of samples in the dataset. The formula for accuracy is given as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Fig.13 Evaluation Metric: Accuracy

2.3.3 Precision

Precision is the proportion of predicted positive cases that are actually positive. The formula for precision is given as:

$$Precision = \frac{TP}{TP + FP}$$

Fig.14 Evaluation Metric: Precision

2.3.4 Recall

Recall is the proportion of predicted positive cases that we predicted correctly. The formula for recall is given as:

$$Recall = \frac{TP}{TP + FN}$$

Fig.15 Evaluation Metric: Recall

2.3.5 F-Measure

F-measure is the harmonic mean of the precision and recall values. The formula for F-measure is given as:

$$F - measure = \frac{2*Recall*Precision}{Recall + Precision}$$

Fig.16 Evaluation Metric: F-measure

2.3.6 AUC-ROC Curve

The Area Under the Curve Receiver Operating Characteristic is one of the most important evaluation metrics to measure the performance of our classifier. ROC can be defined as the probability curve and the AUC can be defined as degree of separability [26]. The higher the AUC the better the model is at predicting our classes correctly.

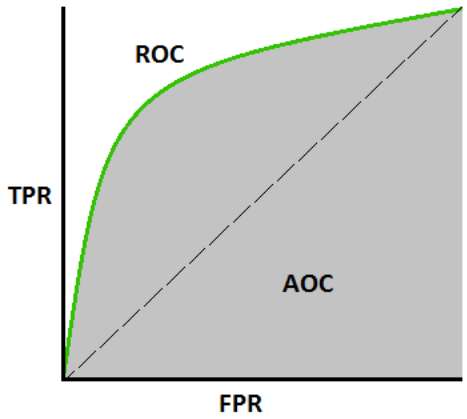

Fig.17 Evaluation Metric: AUC-ROC Curve

2.4 Statistical Tests

In this section, the different statistical tests used in the project have been discussed in detail. In machine learning, it is important to understand if the results of the classifier are statistically distinguishable. We have used three statistical tests to get the majority voting and verify the results of our classifier.

2.4.1 Pearson chi-squared test

The first step is to determine the chi-squared test statistic which is normalized sum of squared deviations between observed and desired values [27]. Second step is defining the degrees of freedom and since all 9 digits are possible in our data, the degrees of freedom are 8. The Pearson chi-squared test is given by the following formula:

The Formula for Chi Square Is

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

where:

c = degrees of freedom

O = observed value(s)

E = expected value(s)

Fig.18 Formula: Pearson Chi-Squared Test

2.4.2 Kolmogorov-Smirnov test (K-S test)

The Kolmogorov-Smirnov test is popularly known as goodness of fit test. The two-sample K-S test is a general nonparametric method which compares the distribution of two independent samples and gives a statistic and p-value [28]. The p-value is interpreted the same as other tests where you can reject the null hypothesis that two samples are identical if the p-value is less than level of significance. Our level of significance is 0.05 or five percent.

2.4.3 Pearson Correlation Coefficient test

Pearson Correlation Coefficient test is also known as Pearson's r test. It measures the linear correlation between 2 sets of sample values. The test is essentially the normalization of covariance or the ratio of covariance of two values divided by the product of their standard deviation [29]. The test will always give a correlation value between -1 and 1.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

r = correlation coefficient
x_i = values of the x-variable in a sample
$\bar{x}$ = mean of the values of the x-variable
y_i = values of the y-variable in a sample
$\bar{y}$ = mean of the values of the y-variable

Fig.19 Formula: Pearson Correlation Coefficient Test

3. Implementation

In this chapter, we will discuss in detail the databases and step-by-step pipeline for the implementation of the project. This chapter will explain the setup used to train the ML models and statistical techniques used in detail.

3.1 Setup

Each part of implementation of project has been done by using multiple conda virtual environments. Conda can run on many operating systems, and it is an open-source package and environment management system. The host machine has the following configurations:

- Model: MacBook Pro
- Processor: 2.3 GHz 8-Core Intel Core i9
- Memory: 16 GB 2400 MHz DDR4
- Graphics: Intel UHD Graphics 630 1536 MB
- Operating System: macOS Monterey Version 12.0.1

All the tests, trainings, and experiments have been executed on the host machine only.

3.2 Dataset

Twitter is a global communication network available to the public in real-time. It is used by millions of users daily who end up generating lots of metadata in the form of short bio, location, @handle, display name, number of followers, number of statuses, number of friends, etc. Twitter metadata can be accessed and retrieved programmatically

with the help of the Twitter API which has the latest version Twitter API v2 launched in November 2021. Twitter has recently announced rate limits to the API service which has in turn reduced the access to the metadata and slowed down data collection and retrieval. Due to the rate limits, this project has been built with the help of four publicly available datasets [2], [11], [12], [13]. All the datasets have been discussed in detail below.

Prof. Jennifer Golbeck conducted an analysis over five social networking websites to study if Benford's Law applies to online social networks [2]. For analysis of Benford's Law on Twitter they collected egocentric data of 21,135 Twitter users by randomly generating user IDs. This dataset called anonymizedTwitter dataset is not labelled so the labelling approach is explained below. This is the only dataset that provided us with first-degree egocentric network data, and it is available at https://github.com/jgolbeck/BenfordData.

The botometer-feedback dataset was constructed by researchers Kai-Cheng Yang et. al. in 2019. The Botometer service formerly known as BotOrNot is a bot detection tool which was developed by the researchers at Network Science Institute of Indiana University. It has been live since May 2014 and has been used significantly over the years. The botometer-feedback dataset was created by manually labeling the Twitter accounts flagged by Botometer service. The dataset has 143 'bot' and 386 'human' labelled accounts.

The third dataset is called cresci-2017 and was collected by Stefano Cresci et. al. [12]. To create this dataset the authors made a more fine-grained classification to group three different categories: traditional spambots, social spambots, and fake followers. Traditional spambots tend to just tweet out the same content repeatedly while social

spambots will try to disguise themselves like normal profiles. Fake followers are just part of a botnet that follows an account for money. The cresci-2017 dataset is annotated with 9391 'bot' and 3474 'human' labelled accounts.

The last dataset gilani-17 was collected by Zafar Gilani et. al. [13]. The researchers used the Twitter streaming API to group accounts into four categories based on followers counts. They then sampled the accounts and got them annotated from four undergraduate students according to key information. The dataset is comprised of 1090 'bot' and 1413 'human' labelled accounts.

Table 2: Datasets Bot and Human label counts

Dataset	**#Bot**	**#Human**
anonymizedTwitter	317	20,818
botometer-feedback	143	372
cresci-2017	9391	3474
gilani-17	1,090	1,413
Total	**10,941**	**26,077**

3.3 Approach

The approach to implement this project is divided into two easy steps: preprocessing each dataset and combining them, training and testing multiple classifier models and selecting the best model.

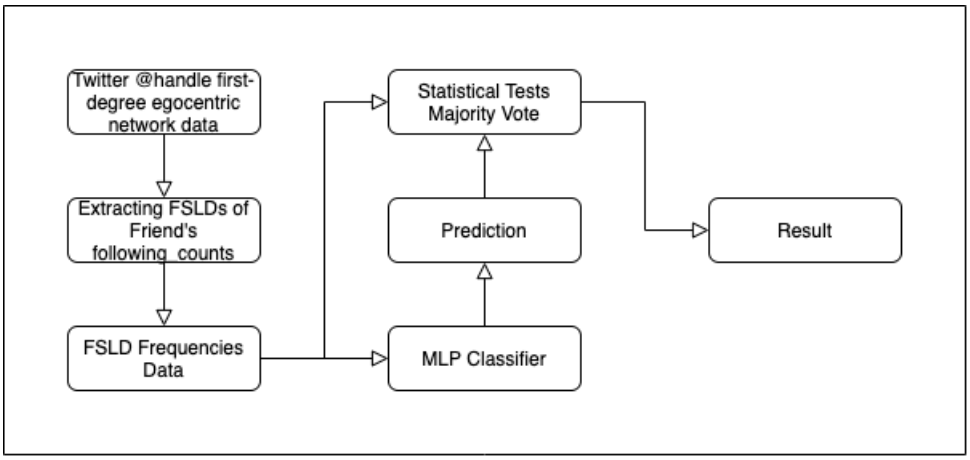

Fig.20 Overview of our approach

3.3.1 Data Preprocessing

Since the first dataset i.e., anonymizedTwitter did not have labels the labelling was done by the author of the project. For the task of labeling, the FSLDs of each of the 21,135 data samples were extracted from their following_counts and then their frequencies were visualized in the form of a histogram against the Benford's Law distribution one sample at a time. Exploratory Data Analysis was performed on each data sample and a bot or human label was assigned to all samples manually.

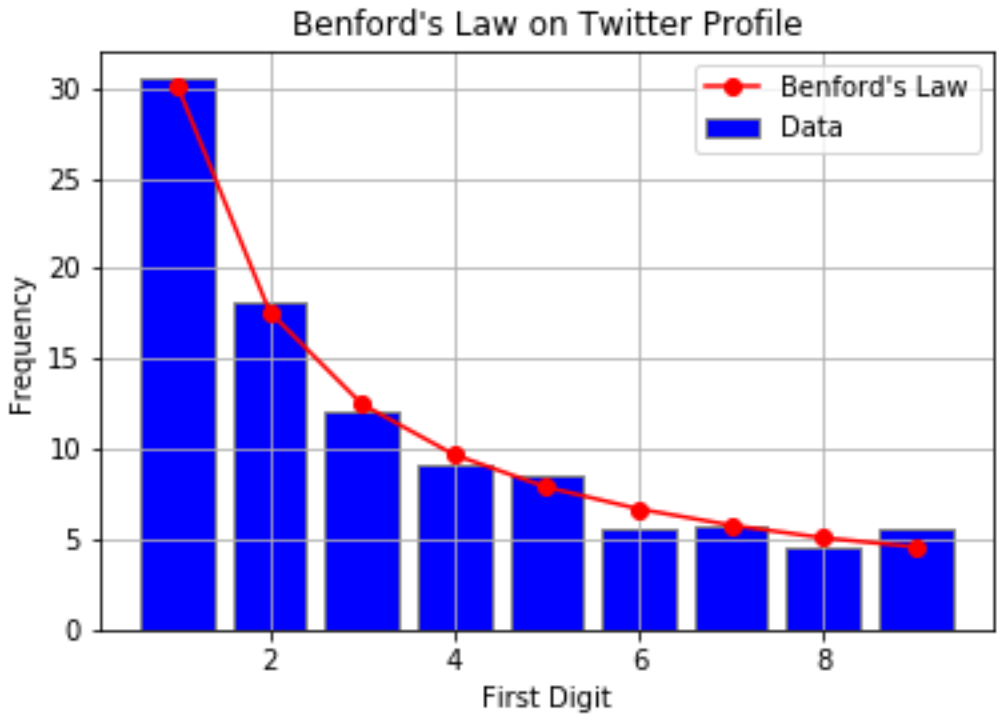

Fig.21 EDA Data Sample that follows Benford's Law Distribution

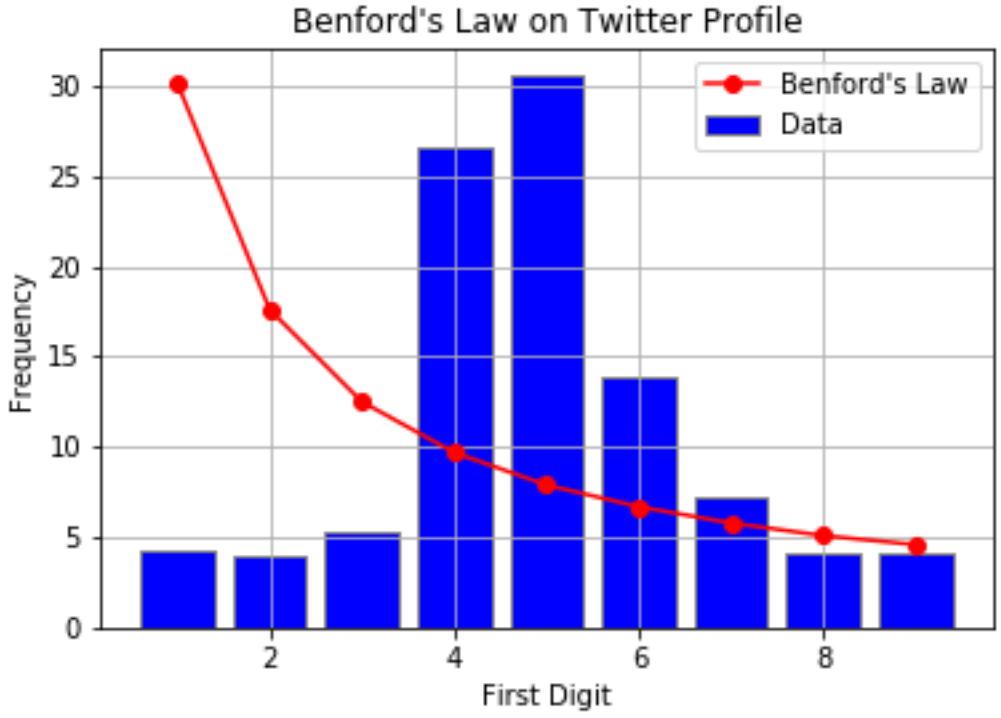

Fig.22 EDA Data Sample that doesn't follows Benford's Law Distribution

The datasets from [11], [12], [13] were only used to collect the Twitter @handle and the 'bot' or 'human' label provided by the original authors. Afterwards the first-degree egocentric network data i.e., the following_counts of each friend for that @handle was collected with the help of the Twitter API manually by the author of this project. Once all the first-degree egocentric data from each of the four datasets was available a new combined dataset was created. This combined dataset only contained the FSLD frequencies of each data sample and a label of 0 for human and 1 for bot profile.

	1	2	3	4	5	6	7	8	9	Bot
0	0.00	100.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1
1	31.82	19.48	7.79	11.04	7.14	7.14	4.55	3.90	7.14	0
2	2.11	96.43	0.70	0.13	0.13	0.06	0.13	0.13	0.19	1
3	40.00	16.36	11.82	5.45	4.55	9.09	6.36	3.64	2.73	0
4	18.57	5.02	24.19	2.05	1.15	1.51	11.23	18.81	17.45	1

Fig.23 Final Dataset with FSLD Frequencies and Bot Label

3.3.2 Training and Testing Classifiers

Once the preprocessing was done and the combined dataset with labels was available Jupyter Notebook was used to read the csv data file. The data was split into train and test sets with a 75:25 split. Synthetic Minority Oversampling Technique (SMOTE) was used to treat the imbalance between our bot and human samples. We trained and tested six supervised machine learning classifiers namely: Logistic Regression, Naïve Bayes, Support Vector Machine, Random Forest, AdaBoost, and MLP. Random Forest and AdaBoost models gave high accuracy scores, but the best model was the neural network model MLP. Next chapter discusses the results.

4. Results

In this chapter, we discuss the results of the experiments performed in our project. The training results for each ML model are discussed in detail with performance measure Confusion Matrix, Accuracy, Precision, Recall, F-Measure, and AUC-ROC curve.

4.1 Naïve Bayes

The first phase of experiments was the training and testing of Naïve Bayes classifier. This was a naïve approach as the model considers all the features independently with no correlation. With all the 9 features evaluated independently the Naïve bayes model has good performance. Figure 24 shows the AUC-ROC curve and Confusion Matrix. Table 3 shows the overall performance of the first phase of experiments.

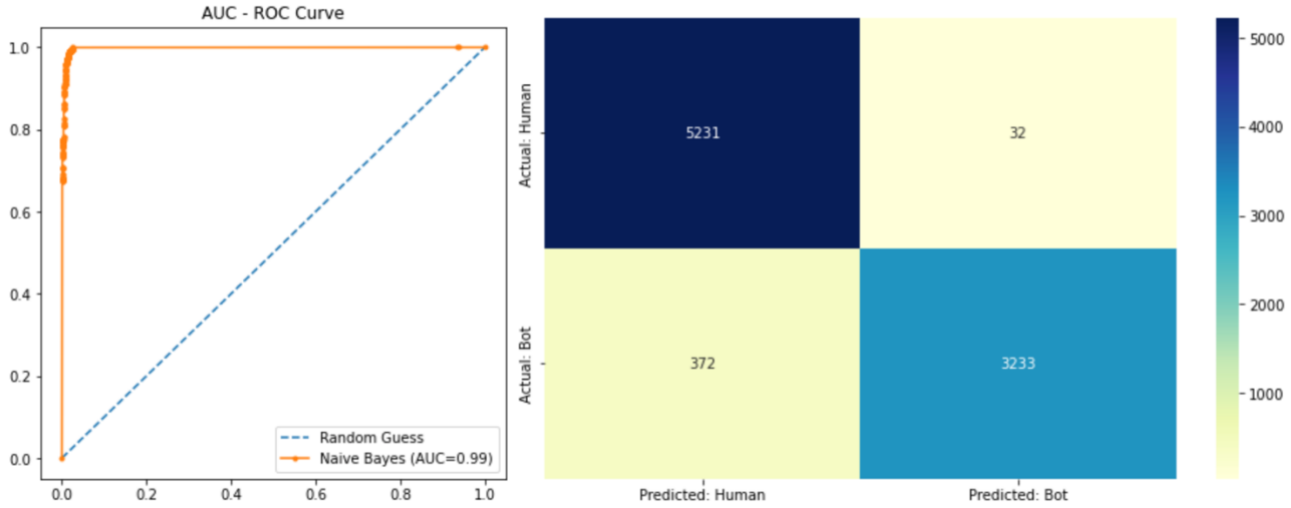

Fig.24 AUC-ROC Curve and Confusion Matrix of Naïve Bayes

Table 3: Naive Bayes Performance

Model	Accuracy	Precision	Recall	F-Measure
Naive Bayes	95.44 %	99.02 %	89.68 %	94.12%

4.2 Logistic Regression

In the second phase of experiments, we trained the logistic regression model which was very quick at training and easier to implement. Since we have only 9 features and a binary classification problem logistic regression model has high performance. Figure 25 shows the AUC-ROC curve and Confusion Matrix. Table 4 shows the overall performance of the second phase of experiments.

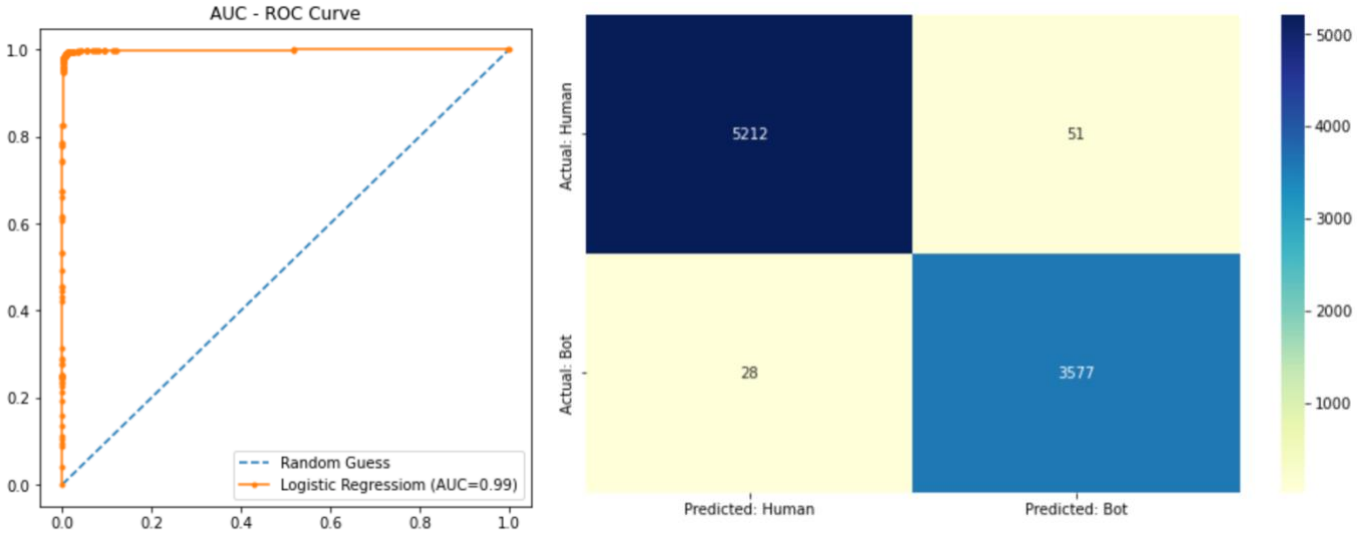

Fig.25 AUC-ROC Curve and Confusion Matrix of Logistic Regression

Table 4: Logistic Regression Performance

Model	Accuracy	Precision	Recall	F-Measure
Logistic Regression	99.11 %	98.59 %	99.22 %	98.91

4.3 SVM

In the third phase of experiments, we trained and tested a Support Vector Classifier model on our dataset. The SVC took more time to train that the previous experiments, but it is well suited for our dataset as we have a binary classification problem at hand. Figure 26 shows the AUC-ROC curve and Confusion Matrix. Table 5 shows the overall performance of the third phase of experiments.

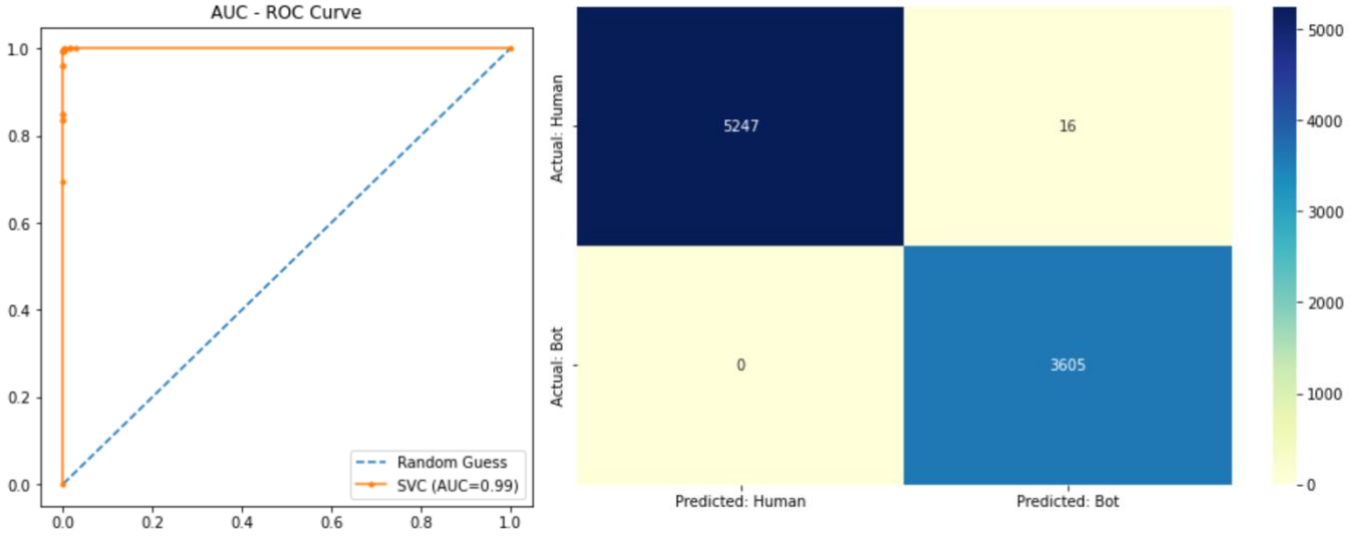

Fig.26 AUC-ROC Curve and Confusion Matrix of SVC

Table 5: SVC Performance

Model	Accuracy	Precision	Recall	F-Measure
SVC	99.82 %	99.56 %	100 %	99.78 %

4.4 Random Forest

In the fourth phase of experiments, we trained and tested a Random Forest Classifier which uses multiple decision trees to gain high accuracy. The model trained faster that SVM and has better overall performance compared to all the previous phases. In supervised machine learning approaches Random Forest model is expected to deliver very high accuracy. Figure 27 shows the AUC-ROC curve and Confusion Matrix. Table 6 shows the overall performance of the fourth phase of experiments.

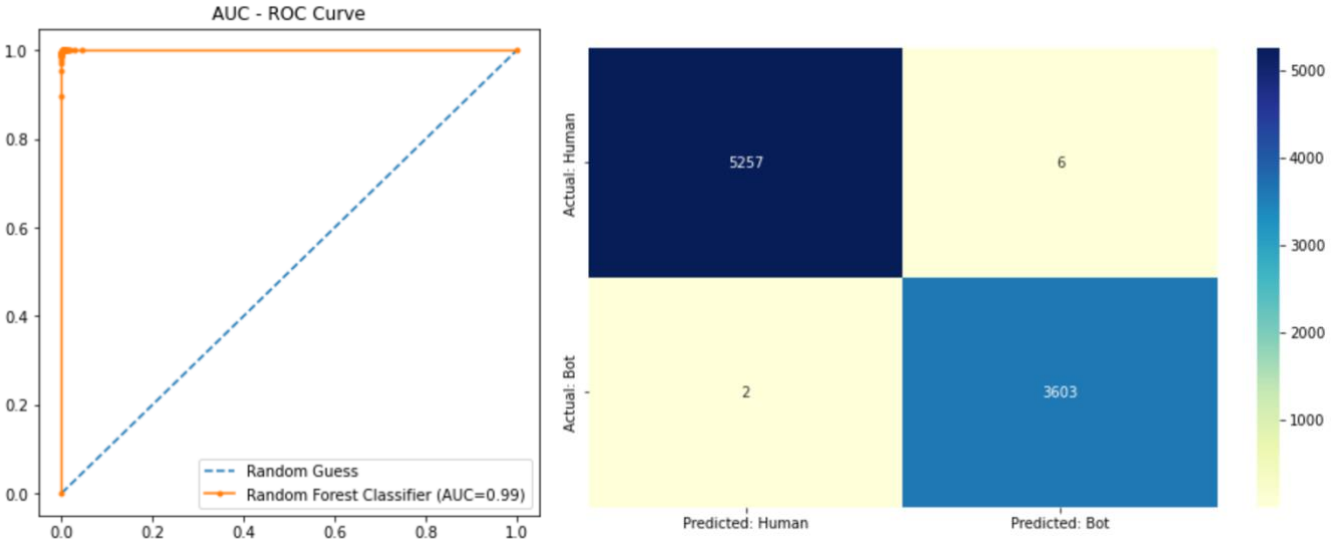

Fig.27 AUC-ROC Curve and Confusion Matrix of Random Forest

Table 6: Random Forest Performance

Model	Accuracy	Precision	Recall	F-Measure
Random Forest	99.91 %	99.83 %	99.95 %	99.89 %

4.5 AdaBoost

The fifth phase of experiments was to push the accuracy score as high as possible, so we trained and tested an Adaptive Boosting Model. This is another ensemble learning technique like Random Forest with high accuracy. It trains multiple weak models and aggregates them with weights to form a strong classifier, but the model tends to overfit. Figure 28 shows the AUC-ROC curve and Confusion Matrix. Table 7 shows the overall performance of the fifth phase of experiments.

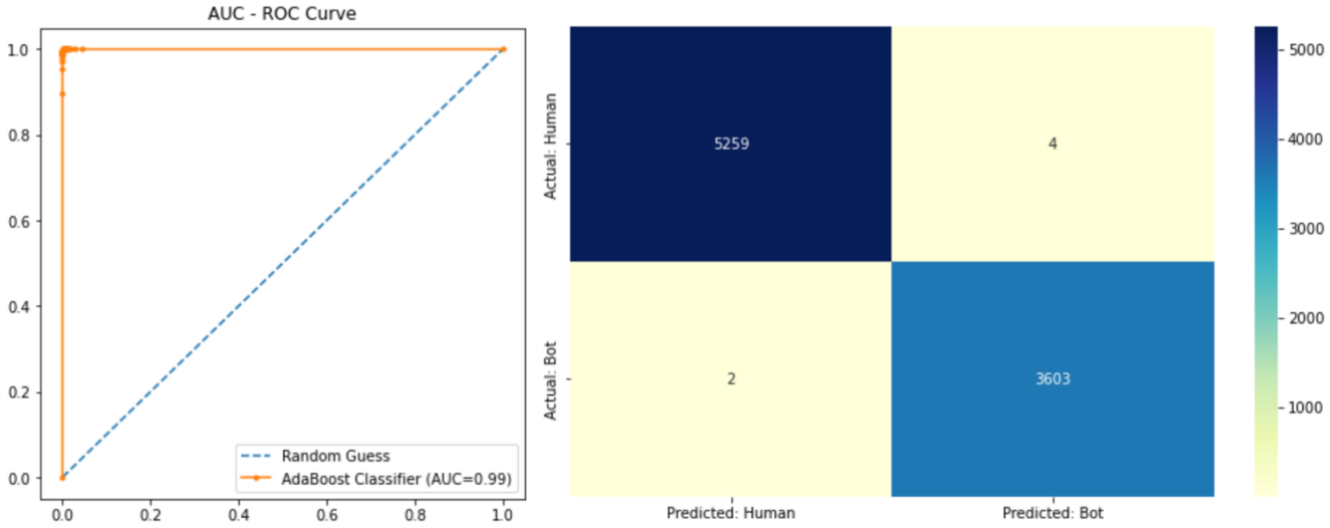

Fig.28 AUC-ROC Curve and Confusion Matrix of AdaBoost Classifier

Table 7: AdaBoost Classifier Performance

Model	Accuracy	Precision	Recall	F-Measure
AdaBoost Classifier	99.93 %	99.89 %	99.95 %	99.92 %

4.6 MLP

The sixth and final phase of experiments was completed by training and testing a feedforward neural network model called Multi-Layer Perceptron. The MLP classifier have and Input and Output layer just like the Logistic Regression model, but it also has hidden layers with neurons to achieve the best possible results. The MLP model ended up giving the highest accuracy (near perfect) and overall performance. This is the final model that was selected as our final classifier. Figure 29 shows the AUC-ROC curve and Confusion Matrix. Table 8 shows the overall performance of the sixth phase of experiments.

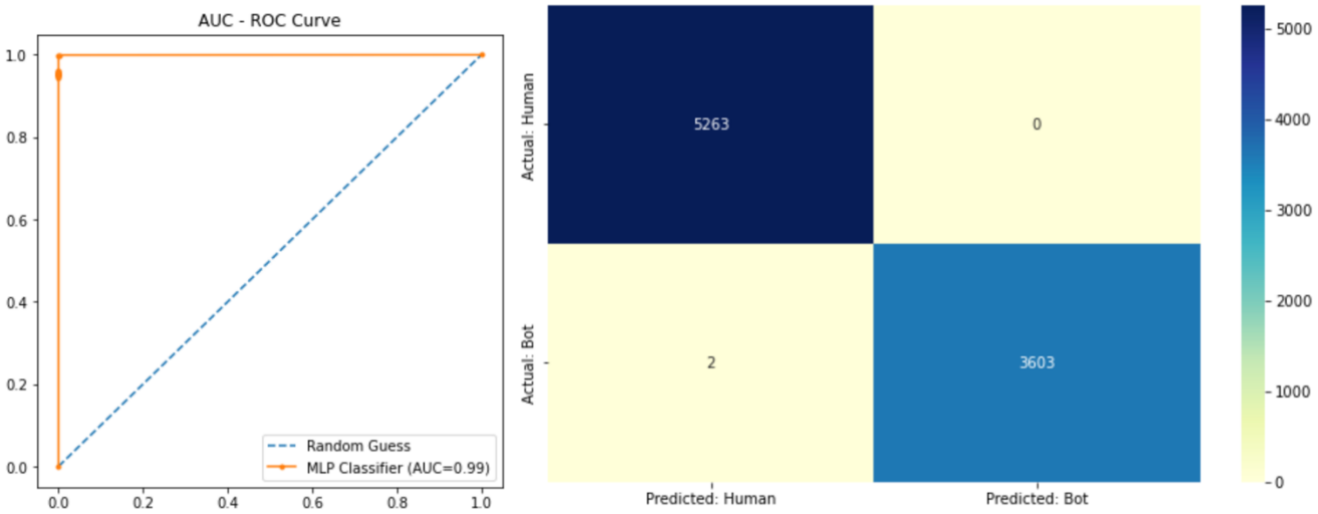

Fig.29 AUC-ROC Curve and Confusion Matrix of MLP

Table 8: MLP Performance

Model	Accuracy	Precision	Recall	F-Measure
MLP	99.98 %	100 %	99.95 %	99.97 %

4.7 Latency Analysis of ML Algorithms

In this section, we are going to study the latency in training our supervised machine learning models on our ground truth dataset. When the size of the dataset increases the training time of our models tends to increase significantly. We are going to train each of our models on our training dataset of around 25k samples. The latency analysis will be based on the training time taken by each model in milliseconds. The Table 9 below shows the latency analysis results.

Table 9: Latency Analysis on Training Dataset

MODEL	**Latency (ms)**
Naïve Bayes	7.26 ms
Logistic Regression	51.26 ms
Support Vector Machine	2935.64 ms
Random Forest Classifier	1862.07 ms
AdaBoost Classifier	1526.95 ms
Multi-Layer Perceptron	3308.19 ms

With this data we will be able to understand the training times of the algorithms as we scale our datasets for better prediction accuracy. The optimum algorithm will be chosen based on time and speed constraints for future scaling.

4.8 Statistical Tests Majority Vote

Once the neural network MLP model was trained and selected for its high-performance accuracy, we were ready to test our model on new accounts and then verify our model's classification based on a majority vote of our three statistical tests. We took four random samples with two bots and two humans and tested our MLP classifier prediction with our statistical tests for majority vote. Here we are testing the goodness of fit of the FSLDs with the Benford's Law distribution. The hypothesis for tests was formulated as:

Null hypothesis (H_0) = Account FSLDs follow Benford's Law

Alternative hypothesis (H_1) = Account FSLDs violate Benford's Law

If p-value < 0.05 then reject H_0 (Nonconformity), else accept H_0 (Conformity)

Table 10: Statistical Tests Majority Vote

	Bot1	**Bot2**	**Human1**	**Human2**
Pearson Chi-Squared Test	Nonconformity	Nonconformity	Conformity	Conformity
Kolmogorov-Smirnov Test	Nonconformity	Conformity	Conformity	Conformity
Pearson Correlation Coefficient Test	Nonconformity	Nonconformity	Conformity	Conformity
Majority Vote	**Bot**	**Bot**	**Human**	**Human**

The majority vote of our statistical tests validates the prediction results of our MLP classifier and hence we have proved that the MLP classifier trained on our ground truth dataset can be used to detect social bots on Twitter.

5. Conclusion and Future Works

This chapter will discuss in detail the conclusion of our project and any future scope for improvements or advancements. Since social media bot detection is becoming an increasing problem more research and advanced techniques can extend this project.

5.1 Conclusion

The proposed technique in this project, which given a Twitter @handle will collect the following_counts of all the friends of that profile and extract the FSLD frequencies to feed our neural network classifier works best for detecting malicious social bots against humans. This is due to the strategic selection of our databases used as ground truth for training our model. The main goal of the project was to create a ground truth dataset from scratch and then training a neural network model on the dataset while also validating the results with majority vote of statistical tests has been achieved. The project enables us to identify if a Twitter user is a malicious social bot or human with very little efforts. The overall project technique is novel and has never been implemented in this fashion.

Any supervised machine learning technique used for bot detection will only be as good as the ground truth data that was used to train it. As the social bots keep changing their patterns and techniques rapidly, even the most sophisticate bot detection algorithms will fail as their training rules become outdated. Benford's Law is an unavoidable naturally occurring phenomenon present in the world and it is prevalent on Twitter [18]. Since, the malicious bots break away from the natural pattern by synthetically following other social bots and malicious accounts they tend to unknowingly violate the Benford's Law. Hence, our project will be able to identify malicious bots or suspicious accounts even if the bot behavior patterns keep evolving.

There are certain limitations faced by our project due to the use of anonymizedTwitter dataset [3] and the way Benford's Law works. To analyze any account on Twitter we need the account to be following at least 100 other accounts. First reason for this is, it was the technique used in data collection by the authors of anonymizedTwitter dataset, this makes our classifier bad at detection if the account has less friends. Second reason for this is, Benford's Law requires orders of magnitude and certain number of samples to work with the FSLD frequency distribution.

5.2 Future Works

The project can be extended to create a web browser extension where the users will be able to classify Twitter accounts in real-time without gathering all the first-degree egocentric data and feeding it to our model. The browser extension will be able to send pop-up messages to user to warn about any suspicious profile that is encountered during regular activity.

Another extension to this project would be, once our classifier flags a Twitter account, we could employ other techniques to identify if the suspicious account is part of a bigger botnet.

This same research technique can be applied on Facebook datasets to see if we can successfully classify bots on Facebook with the help of Benford's Law.

LIST OF REFERENCES

[1] S. Newcomb, "Note on the Frequency of Use of the Different Digits in Natural Numbers," *American Journal of Mathematics*, vol. 4, no. 1, pp. 39-40, 1881.

[2] F. Benford, "The law of anomalous numbers," *Proceedings of the American Philosophical Society*, pp. 551-572, 1938.

[3] J. Golbeck, "Benford's Law can detect malicious social bots," *First Monday*, Aug 2019. [Online] Available: https://firstmonday.org/ojs/index.php/fm/article/view/10163/8063

[4] J. Golbeck, "Benford's Law applies to online social networks," *Plos One*, vol. 10, no. 8, pp. 1-11, Aug 2015.

[5] M. J. Nigrini, "Benford's Law: Applications for forensic accounting, auditing, and fraud detection," *John Wiley & Sons*, vol. 586, Apr 2012.

[6] C. Durtschi, W. Hillison, and C. Pacini, "The Effective Use of Benford's Law to Assist in Detecting Fraud in Accounting Data," *Journal of Forensic Accounting*, vol. 99, no. 99, pp. 17-34, Jun 2004.

[7] M. Kolomeets, D. Levshun, S. Soloviev, et al. "Social networks bot detection using Benford's law," *in 13th International Conference on Security of Information and Networks*, pp. 1-8, Nov 2020.

[8] M. Kolomeets, O. Tushkanova, D. Levshun, et al. "Camouflaged bot detection using the friend list," *29th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, pp. 253-259, IEEE, Mar 2021.

[9] E. Ortiz-Ospina, "The rise of social media," *Our World in Data*, 2019. [Online] Available: https://ourworldindata.org/rise-of-social-media

[10] L. Madahali, M. Hall, "Application of the Benford's law to Social bots and Information Operations activities," *2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, pp. 1-8, IEEE, Jun 2020.

[11] K. C. Yang, O. Varol, C. A. Davis, et al. "Arming the public with artificial intelligence to counter social bots," *Human Behavior and Emerging Technologies*, vol. 1, no. 1, pp. 48-61, Jan 2019.

[12] S. Cresci, R. Di Pietro, M. Petrocchi, et al. "The paradigm-shift of social spambots," *Proceedings of the 26th International Conference on World Wide Web Companion*, pp. 963-972, Apr 2017.

[13] Z. Gilani, R. Farahbakhsh, G. Tyson, et al. "Of Bots and humans (on Twitter)," *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 349-354, Jul 2017.

[14] C. A. Davis, O. Varol, E. Ferrara, et al. "BotOrNot: A System to Evaluate Social Bots," *Proceeding of the 25th International Conference Companion on World Wide Web*, pp. 273-274, Apr 2016.

[15] S. Cresci, R. Di Pietro, M. Petrocchi, et al. "Fame for sale: Efficient detection of fake Twitter followers," *Decision Support Systems*, vol. 80, pp. 56-71, Dec 2015.

[16] Z. Chu, S. Gianvecchio, H. Wang, et al. "Who is Tweeting on Twitter: Human, Bot, or Cyborg?," *Proceedings of the 26th Annual Computer Security Applications Conference*, pp. 21-30, Dec 2010.

[17] O. Loyola-Gonzalez, R. Monroy, J. Rodriguez, et al. "Contrast Pattern-Based Classification for Bot Detection on Twitter," *IEEE Access*, vol. 7, pp. 45800-45817, Apr 2019.

[18] I. Mbona, J. H. Eloff, "Feature selection using Benford's law to support detection of malicious social media bots," *Information Sciences*, vol. 582, pp. 369-381, Jan 2022.

[19] A. Pant, "Introduction to Logistic Regression," Jan 2019. [Online] Available: https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148

[20] S. Yildirim, "Naïve Bayes Classifier - Explained," Feb 2020. [Online] Available: https://towardsdatascience.com/naive-bayes-classifier-explained-50f9723571ed

[21] A. Yadav, "Support Vector Machines (SVM)," Oct 2018. [Online] Available: https://towardsdatascience.com/support-vector-machines-svm-c9ef22815589

[22] T. Yiu, "Understanding Random Forest," Jun 2019. [Online] Available: https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[23] M. Fabien, "Boosting and AdaBoost clearly explained," Feb 2019. [Online] Available: https://towardsdatascience.com/boosting-and-adaboost-clearly-explained-856e21152d3e

[24] S. Loukas, "Multilayer Perceptron Classifier (MLP)," Nov 2021. [Online] Available: https://towardsdatascience.com/classifying-handwritten-digits-using-a-multilayer-perceptron-classifier-mlp-bc8453655880

[25] S. Narkhede, "Understanding Confusion Matrix," May 2018. [Online] Available: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

[26] S. Narkhede, "Understanding AUC-ROC Curve," Jun 2018. [Online] Available: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

[27] M. J. Slakter, "A comparison of the Pearson chi-square and Kolmogorov-Smirnov tests with respect to validity," Journal of the American Statistical Association, vol. 60, no. 311, pp. 854-858, Sep 1965.

[28] F. J. Massey Jr, "Kolmogorov-Smirnov test for goodness of fit," *Journal of the American Statistical Association*, vol. 46, no. 253, pp. 68-78, Mar 1951.

[29] J. Benety, J. Chen, Y. Huang, et al. "Pearson correlation coefficient," *In Noise reduction in speech processing*, pp. 1-4, 2009.

[30] M. J. Nigrini, "A Taxpayer Compliance Application of Benford's Law," *The Journal of the American Taxation Association*, vol. 18, no. 1, p. 72, 1996.

[31] L. D. Samper-Escalante, O. Loyola-Gonzalez, R. Monroy, et al. "Bot Datasets on Twitter: Analysis and Challenges," *Applied Sciences*, vol. 11, no. 9, p. 4150, Jan 2021.

[32] S. Oates, J. Gray, "#Kremlin: Using Hashtags to Analyze Russian Disinformation Strategy and Dissemination on Twitter," *Available at SSRN 3445180*, Aug 2019.

[33] B. F. Roukema, "A first-digit anomaly in the 2009 Iranian presidential election" *Journal of Applied Sciences*, vol. 41, no. 1, pp. 164-199, Jan 2014.

[34] S. Cresci, R. Di Pietro, M. Petrocchi, et al. "DNA-inspired online behavioral modeling and its application to spambot detection," *IEEE Intelligent Systems*, vol. 31, no. 5, pp. 58-64, Mar 2016.