Twitter bot detection using supervised machine learning

Twitter bot detection using supervised machine learning

A Ramalingaiah[1], S Hussaini[1], S Chaudhari[1]
[1]Department of CSE, MSRIT, Bangalore, Karnataka, India

E-mail: aparna@msrit.edu

Abstract. In the world of Internet and social media, there are about 3.8 billion active social media users and 4.5 billion people accessing the internet daily. Every year there is a 9% growth in the number of users and half of the internet traffic consists of mostly bots. Bots are mainly categorized into two categories: good and bad bots; good bots consist of web crawlers and chat bots whereas bad bots consist of malicious bots which make up 20% of the traffic, the reason they are not good is that they are used for nefarious purposes, they can mimic human behavior, they can impersonate legal traffic, attack IoT devices and exploit their performance. Among all these concerns, the primary concern is for social media users as they represent a large group of active users on the internet, they are more vulnerable to breach of data, change in opinion based on data. Detection of such bots is crucial to prevent further mishaps. We use supervised Machine learning techniques in this paper such as Decision tree, K nearest neighbors, Logistic regression, and Naïve Bayes to calculate their accuracies and compare it with our classifier which uses Bag of bots' word model to detect Twitter bots from a given training data set.

1. Introduction

Twitter bots are estimated to be 23 million for every 1.3 billion accounts, this approximates for 15% of the users. Different classes of users use Twitter for different purposes. For instance, business owners use to advertise about their products and get feedback, authors use to sell their books, Twitter communities consist of likeminded people who discuss their ideas online. Bot is another exciting feature of Twitter. Twitter bot is an automated version of a Twitter account i.e. it is a software which controls a Twitter account with the help of Twitter application interface [3]. Twitter is responsible in setting the rules of automation. These bots can perform several activities like follow, unfollow, tweet, re-tweet, like, or even directly message any accounts without human intervention. A study conducted by University of Southern California and Indiana University revealed that around 15% of the active Twitter accounts are bots [4]. These bots behave just as humans and thus it's difficult to spot them. These bots chat with unknown users or post irrelevant content like photographs, poetry at regular intervals. Popular bots that exist include @netflix bot that tweets about new shows available on Netflix, @HundredZeros which is a Twitterbot that posts URLs of the freely available eBooks on Amazon. and many more [5]. Twitter's huge active audience and interesting features have attracted unethical users like hackers, cyber criminals, cyber bullies for performing malicious activities. Many Twitter bots called political bots post malicious URLs and sensitive contents. Cyber criminals use these bots to circulate malicious and phishing URLs through automation and at regular intervals so as to extract sensitive information of varied people resulting in phishing scams and frauds. The number of malicious Twitter bots that distribute fake news, malicious URLs and distorted images have significantly increased in recent years. To analyze bot activity around the pandemic, CMU researchers have collected more than 200 million tweets discussing corona virus or COVID-19 since January 2020 and discovered that out of the top 50 influential retweets, 82% were bots. These malicious Twitter bots

try to reach as many people as possible by posting fake news, malicious URLs regularly. These bots assist hackers in breaching data. However, frequent complaint received from users is that Twitter's anti-spam actions are blocking legitimate Twitter users' accounts. Twitter recently admitted that a spam clean-up effort led to accidental suspension of accounts. In this research, Kaggle dataset of given Twitter bots with several features such as URL, id, description, listed count is used. After finding the respective accuracies, a bag of bots' algorithm is created to increase accuracy in detecting bots

2. Related Works

The most common techniques used for bot detection are Naïve Bayes, support vector machine (SVM), and neural networks for supervised machine learning. Newer work uses sentiment-based models that learn from users' content and user behavior to detect bots, they achieve an F1 score of 88.30%. Another method which was used as SVM, they make use of Levenshtein distance to detect bots for their classification approach, it detects similarities between the tweets but not the account they were created from, it gives an accuracy of around 95.77% for the given dataset [4]. Clustering algorithms with 126 features improved the performance by detecting 97.6% bots and also the more sophisticated bots crawling on the platform [4]. Network techniques are one of the most paramount techniques needed for bots to be detected, their followers, popularity in various trending issues, and their aid in criminal manipulation [3].

Single class usage to train model is used compared to the binary class d to train a model. It is also called anomaly detection. Decorate classifier uses content, tweet account, and account usage features [1]. SVM is a non-probabilistic linear binary classifier. It plants all random points in a graph and the differentiator is used to divide the classes. The categories are segregated based on which side of the line they fall [2]. Neural network uses a Sigmoid function, where input, hidden and output nodes are used. The data sent through the input node is trained by the hidden nodes and the output node gives the results of fake and real profiles [2]. Random forest uses multiple decision trees and merges them. It is a combination of multiple learning models that increase the overall accuracy [2]. Network-centric technique is based on community detection and egocentric networks, which are mainly used in social interacting features, they collect the frequency of tweets rather than the origin of the tweet itself. [3]. K_f_reimp approach is a reimplementation of a reference classifier based on feature extraction which they have used. They have extracted the important features and used the AdaBoost module to create their classifier [4]. Statistical approach uses language-independent features which gives it an edge for the precision of analyzing the data. It avoids using the content parameters as much as possible [5]. Content-based approach uses language-dependent features without considering the content or the matter. Classification of tweets is done in two classes namely spam and non-spam users [5]. Hybrid based approach is a combination of statistical and content-based approach with more than 1000 parameters [5]. Supervised and unsupervised models are used, they collect data on the user and tweet objects which are used to classify as bots or nonbots [3]. Petri net model tracks the user's behavior and activities to generate a data set from different profiles. It keeps track of the user's content, the activities, its followers, and check if any malicious URL's have been sent or not. Features are described in the form of a state system or transitions [8]. Organized behavior model is a robust where user features include tracking of the user's status and temporal features where it checks the synchronicity of tweets in a particular time interval. It checks the propagation of bots in other user accounts to keep track of its proliferation [9]. Semi-supervised machine learning techniques are used. Most of the crucial features are extracted such several tweets, the response time, interval of tweets per second; anomaly detection is used to find out suspicious activities of users. A social graph is made using the set of anomalies as well as the crucial features with a set of conditions, which is used to detect bots [10]. The textual and image-based analysis is used. Image-based was used since most of the text was embedded in the images. The keywords would be extracted and reduced to their root forms, any repeated occurrence would be omitted and the input of these words would be given to the classifiers to get the output [11]. Unsupervised models to examine computer access, Markov models to study OS malware and natural language processing to extract/analyze messages from files is used [11].

Phishing detection uses a machine-learning classification approach based on random decision trees called PILFER. It has high accuracy and low error rate [11]. Table 1 gives the accuracies of these models.

Table 1. Accuracy of the applied techniques

Technique	Accuracy score
One class [1]	>95%
Decorate classifier [1]	F1 score of 0.88 or 88%
SVM [2]	70.3%
Neural networks [2]	90.1%
Random forest [2]	99.0%
Network-centric [3]	95.2%
K_f_reimp [4]	>98.0%
Statistical [5]	97.55% for Deep forest
Content-based [5]	92.83% and 94.25% for supervised Word2vec
Hybrid [5]	> 95% for synthetic minority overlapping
Machine learning [3]	97.6% for clustering with 126 features
Petri net model [8]	>97% for the random forest
Organized behaviour [9]	>97% for the random forest
SocialBotHunter [10]	99.0%
Spam detection [11]	SVM gives best performance
Malware detection [11]	Same as spam detection
Phishing detection [11]	Naïve Bayes classifier with clustering gives good results.

3. Twitter Bot Detection Method

We are using a dataset from Kaggle [15] as shown in Figure 1. This dataset contains various attributes like URL, description, friends, several followers, a screen name (used to communicate online), location, id, verified (if the user is authenticated), favorite (used for liked tweets), listed count. The data set is trained to identify bots. On this dataset any imbalance of data is removed, features are extracted. For feature independence, the Spearman coefficient is applied. The resulting coefficients; few of them are used in feature extraction and the rest for feature engineering. For feature engineering, a bag of bots' words is fed and applied to the new features. Using these features Decision tree (DT), Logistic regression (LR), K nearest neighbors (KNN), and Naïve Bayes (NB) algorithms are implemented. The algorithm with the highest accuracy is calculated and tested for real-time data.

The data is parsed from training data CSV file containing parameters which are required to detect the bots from the bots, bots are represented by 1 and non-bots as 0. The training data has 2792 entries and test data has 576 entries. The status column has all special characters and is in JSON format, the id is randomly generated in no particular order, and here screen name and description are crucial in determining the bots. The bot column has binary values, which are 0 and 1; in further evaluation, this column is used for cross-checking the accuracy of the model. This process is shown in the Figure 2.

	A	B	C	D	E
1	id	id_str	screen_na	location	descr
2	8.16E+17	"81574578	"HoustonP	"Houston,	"Rare
3	4.84E+09	4.84E+09	kernyeahx	Templevill	From
4	4.3E+09	4.3E+09	mattlieberisbot		Inspir
5	3.06E+09	3.06E+09	sc_papers		
6	2.96E+09	2.96E+09	lucarivera1	Dublin, Un	Inspir
7	8.41E+17	8.41E+17	dantheimp	Austin, TX	Just a
8	2.48E+09	2.48E+09	_all_of_us	in a machi	bot b
9	3.33E+09	3.33E+09	KatamariItems		[Bot
10	3E+09	3E+09	AutophagyPapers		Twitt
11	3.27E+09	3.27E+09	HSC_papers		

Figure 1. Sample of training data.

From the given data set, we find out all the null values for the given features. Figure 3 shows that location, description, url, status, and has_extended_profile have the highest number of missing values. The behavior of non-bots and bots have a distinctive feature wherein users who turn out to be bots have more followers on their profile as compared to friends and non-bots have equitable distribution of friends and followers as shown Figure 4.

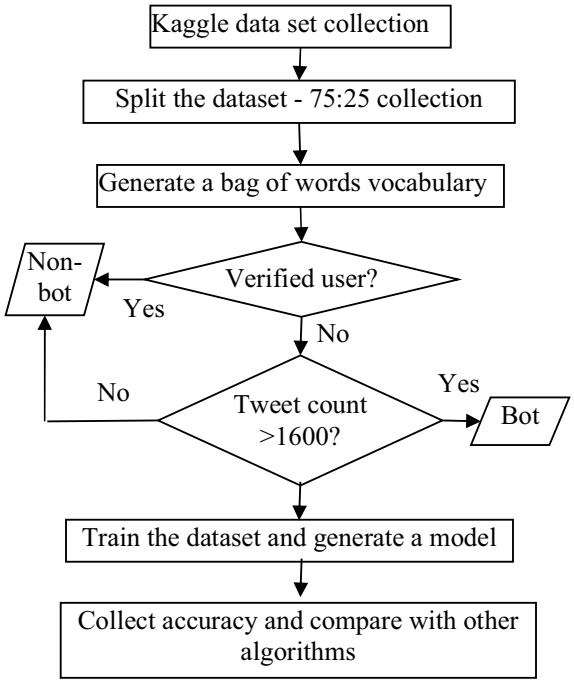

Figure 2. Bag of words flowchart.

The listed count is checked, if the count is more than 16000, it is more likely to be a bot and if the number of followers is more than 10000 and less than 200 then also chances of being a bot. Extracting the conditions from the features screen name, location, and verified, additional features are created to categorize the bots separately; the features are listed respectively as screen_name_binary, location_binary, and verified_binary. The acceptable output for these new features is True and False

only.

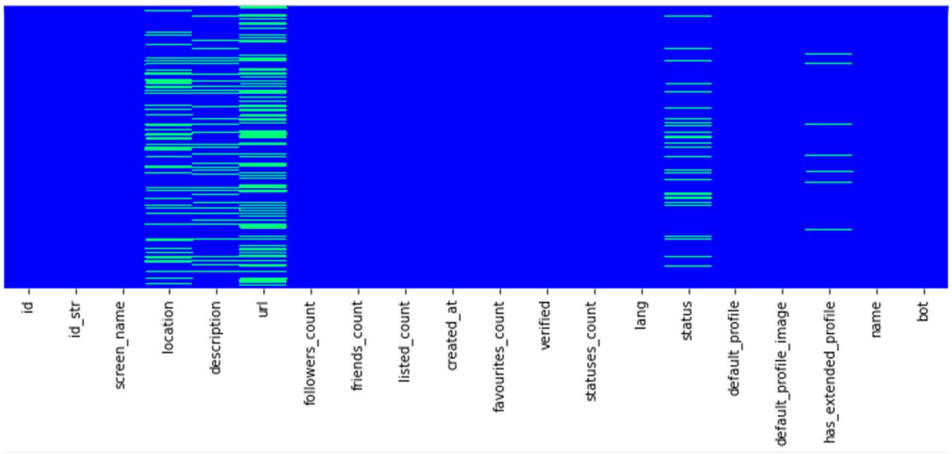

Figure 3. Heat map of missing values.

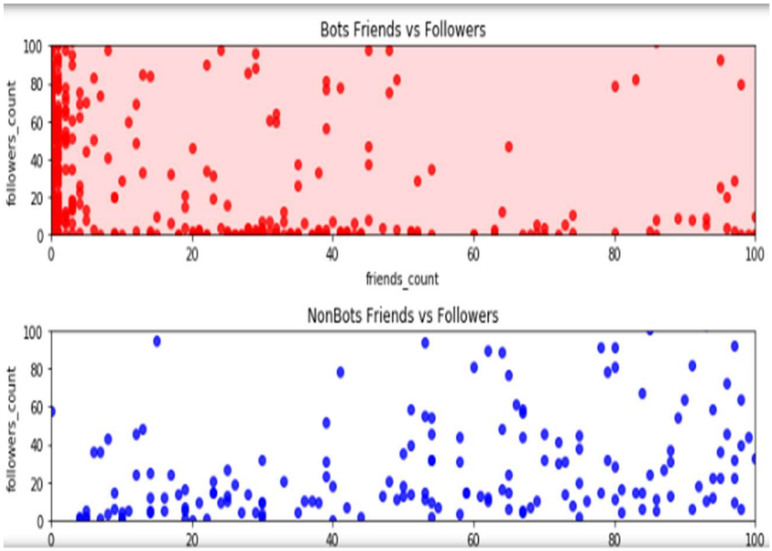

Figure 4. Comparison of Followers in bots vs non-bots.

A Spearman correlation confusion matrix is formed for all the features to check the closest values for the features nearing True positives. Those features are taken to perform feature engineering and the rest are left out. There is no correlation between the id, statuses_count, default_profile, default_profile_image, and the target variable. There is a strong correlation between verified, listed_count, friends_count, followers_count, and target variable. We cannot perform a correlation for categorical attributes. So, we will take screen_name, name, description, status into feature engineering, while use verified, listed_count for feature extraction.

Following are the definitions for the confusion matrix. (1) True positive = All the diagonal values across the matrix (2) False negatives = Sum of rows excluding the True positive (3) False positives = Sum of columns excluding True positive (4) True negative = Sum of all rows and columns excluding

the class's row and column.

Feature engineering and extraction is as follows. (1) A bag of bots' words is taken which consists of words commonly used by bots. (2) A new column is made known as status_binary from the status column. (3) It contains a bag of bots' words needed for testing the modified dataset (4) Feature extraction is performed on a listed count greater than 16000 as it is more likely to be a bot. (5) Training data are given for the algorithms with the following features to be trained: 'bot', 'friends_count', 'followers_count', 'statuses_count', 'description_binary', 'name_binary', 'screen_name_binary', 'status_binary', 'verified', 'listed_count_binary' (6) Various algorithms are trained to check the accuracy for the detection of bots as shown in Table 2. Where Tr indicates training accuracy and Te indicates testing accuracy.

Another metric that we have considered to evaluate our consistency is receiver operating characteristics (ROC) and area under the curve (AUC). ROC plots True positive rate (TPR) against false positive rate (FPR) also known as the sensitivity (y-axis) to specificity (x-axis) scale. AUC is an efficient sorting-based algorithm that tells us how good the curve is, it is the evaluation of the classifier over various threshold values, the more top-left the curve is, the higher will be the area and we get a higher AUC score. Table 3 shows AUC scores in percentage for various train-test set combinations.

Table 2. Accuracies for various classifier

Accuracy		**DT**	**KNN**	**LR**	**NB**
90%-10%	Tr	88.17	86.46	66.4	40.30
	Te	87.1	86.42	65	43.57
80%-20%	Tr	88.5	86.1	68.3	40.68
	Te	87.2	82.6	67.4	39.7
70%-30%	Tr	88.2	86	69	40
	Te	88.4	82	67	38
60%-40%	Tr	86.7	86.20	69.65	40.56
	Te	86.6	82.36	67.71	38.04

Table 3. AUC score for various classifier

Accuracy		**DT**	**KNN**	**LR**	**NB**
90%-10%	Tr	95.7	94.9	72.5	62.3
	Te	95.5	91.9	67.8	61.4
80%-20%	Tr	95.8	94.7	73.4	62.4
	Te	94.3	88.8	73.3	60.5
70%-30%	Tr	95.70	94.5	71.98	63.6
	Te	94.9	88.2	70.5	61.1
60%-40%	Tr	94.9	94.4	73.5	61.4
	Te	93.0	88.3	69.4	61.6

Proposed Classifier: The proposed classifier is based on Bag of Words (BoW) model which is used to extract features from text in the areas of Natural Language Processing or NLP, Computer Vision and Information Retrieval (IR), as shown in Figure 5. This model maintains a bag (multiset) of words and keeps track of multiplicity of words by counting the frequency of words in a document. Document definition could be a single sentence or all of Wikipedia and is mainly based on users' needs. The BoW model generates a frequency vector as its output.

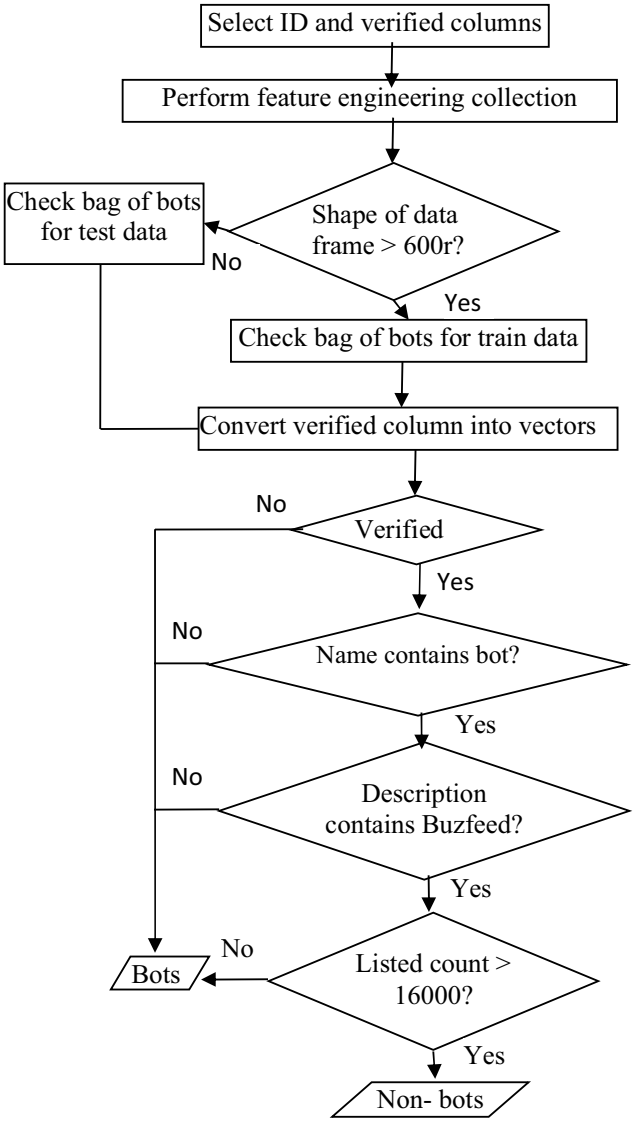

Figure 5. Proposed Classifier.

This model comprises of few steps in which defining the vocabulary (wherein vocabulary is a vector of all the distinct words that were extracted from input documents) happens to be the first step. In the next step, this vocabulary vector is used to convert sentences into a frequency vector. Finally, a numerical vector is generated which can then be utilized as inputs in different machine learning algorithms to classify chatbots and to classify documents into topics, and in many textual-based machine learning applications. This BoW model is popularly used in several different scenarios because of its effectiveness and simplicity. For example, consider the three sentences as follows. (1) My name is Bob (2) My name is not Alice (3) I live nearby.

To make sense of this, we make the vocabulary of words 'My', 'name', 'is', 'Bob', 'not', 'Alice', 'I', 'live', 'nearby'. Now we take each of these words and mark their occurrences with 1's and 0's such that we get 3 vectors as shown in Table 4. Sentence1 vector [1, 1, 1, 1, 0, 0, 0, 0, 0], Sentence2 vector [1, 1, 1, 0, 1, 1, 0, 0, 0] and Sentence3 vector [0,0,0,0,0,0,1,1,1].

This is the principal idea behind Bag of words model we use for our classifier. As seen in the Figure 6 and Figure 7, our classifier performs the highest using bag of words model, with stringent conditions on the test data.

Table 4. Occurrences of each word

Word	Sentence1	Sentence2	Sentence3
My	1	1	0
name	1	1	0
is	1	1	0
Bob	1	0	0
not	0	1	0
Alice	0	1	0
I	0	0	1
live	0	0	1
nearby	0	0	1
Occurrences	4	5	3

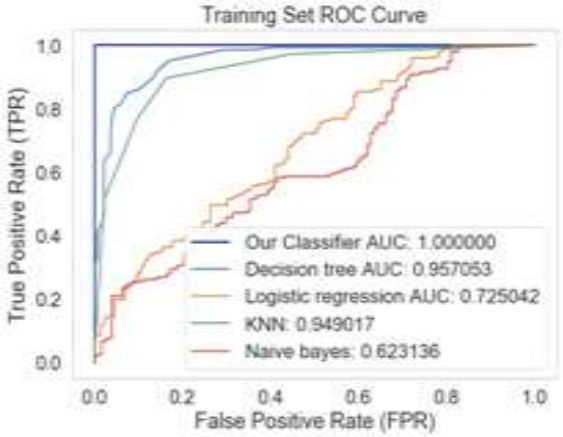

Figure 6. Receiver characteristics of all classifiers during training

4. Result and Discussion

Among all the train-test data sets 90%-10% gives us the highest accuracy and AUC scores compared to others. 60%-40% gives us poor accuracy and AUC scores, 70%-30% average scores, and 80%-20% optimal scores. For our project, we considered 80%-20% as our sample for training as the accuracy and AUC scores compared to 90%-10% are close enough and almost negligible, it is also a fair split and gives us a low variance for random samples.

Our dataset is moderately large, it gives a perfect fit for the ratio of the number of samples and is computationally less intensive when it comes to training the data.

We use accuracy as a measuring parameter to find out the best method as accuracy takes true positives and true negatives into account which is enough to classify bots and nonbots. For 600 samples of data, it'll give accuracy above 85%, hence it can be used in small to moderate samples to detect bots.

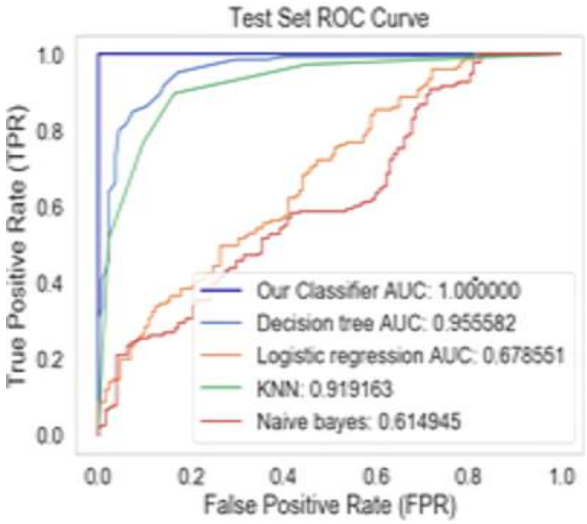

Figure 7. Receiver characteristics of all classifiers during testing

Decision tree is taken because it is one of the intelligent and readable splitting classifiers, as it can take multiple dimensions into account, it does not require any domain knowledge prerequisite to it. With multi-branching, it reduces noise and improves the overall accuracy as shown in Table 5. Target attribute or the root node yields good results as attributes Information gain ratio and Gini index are taken into account whereas KNN does not assume the functional form, this makes it easy to classify based on the nearest distance, it also performs highly well as it is a lazy learner and not much computation is required. For small samples, it may work well as it scores well above 82%.

Table 5. Accuracy and AUC scores (80-20)

Technique	AUC train score	AUC test score	Training accuracy	Test accuracy
DT	95.8%	94.3%	88.5%	87.2%
KNN	94.7%	88.8%	86.1%	82.6%
LR	73.4%	73.3%	68.3%	67.4%
NB	62.4%	60.5%	40.68%	39.7%

Logistic regression and Naïve Bayes perform poorly as logistic regression cannot handle a large number of features and are more prone to overfitting whereas Naïve Bayes doesn't take into account the grammatical nuances or special characters when using to find the probability, improper training of a particular class makes it difficult to predict the result. In any category of train-test data split Naïve Bayes should be avoided as overfitting can be very persistent which yields inaccurate and poor results.

In our classifier we have used the concepts of Bag of words, which have been implemented that gives us the highest accuracy and AUC score as the words in the Status column have been converted into vector format and parsed to a different file with id and bots, Buzzfeed words and listed count functions helped to enhance the accuracy.

As we can see from Table 6, our classifier performs the best as it uses a bag of words model, which only searches for particular words in users, screen names, and tweets. With such a simple algorithm, it efficiently detected those words, vectorize them, and easily classify them as bots or non-bots as two binary values. The training and the test accuracies as exclusively different i.e., the features are trained only on the training data which is a separate file and experimented on test data which is a separate test file. The test data yields accuracies >99% with stringent conditions hence it is considered to be a good classifier for our given dataset. Even though there are other models which perform better than bag of bots, such as Neural networks, Petri model and Statistical approach, they require high computational requirements which aren't cost effective and easy to implement, Bag of words model can be

implemented on any real time data set and yield similar results making it an effective supervised machine learning algorithm. Support Vector Machine is also an efficient classification and regression algorithm with higher speed and performance but it works well only with limited amount of data and bag of bot's word model performs slightly better when it comes to text classification.

Table 6. Comparison of accuracies and AUC

Technique	AUC train	AUC test	Training accuracy	Test accuracy
DT	95.8%	94.3%	88.5%	87.2%
KNN	94.7%	88.8%	86.1%	82.6%
LR	73.4%	73.3%	68.3%	67.4%
NB	62.4%	60.5%	40.68%	39.7%
Own classifier	100%	100%	96%	99.2%

Even though it is simple and easy to implement the algorithm, the few constraints that were faced was that there was no correlation between the words, for are larger data set which contains more than 10,000 entries, it leads to a high dimensionality feature of vectors, since correlation of words was less and there were highly sparse vectors to be found which hindered the output. The principle of the model can be applied to small and medium real time data sets and more importantly since it can be used with small data sets, the results give a good idea as how to approach larger datasets since they follow the same principles and constructs of natural language processing whichever model may be implemented.

5. Conclusion and Future Scope

The bag of words model represented in this paper is effective as it can be used for text-based as well as in image classification. We see the performance of decision tree which gives the highest accuracy due to its splitting of sub trees which gives a greater result, K nearest neighbors gives an optimal accuracy for both training and test set for each train-test split hence giving a balance. Logistic regression and Naïve Bayes perform very poorly in this regard as Logistic regression is prone to overfitting since all the independent variables need to be identified, it has a high reliance on proper presentation of data and Naïve Bayes fails due to poor assumption of independent predictor features, it assumes all attributes are mutually independent, if a variable is not observed in the training set but observed in the test set, the model will not be able to make predictability as the model is assigned zero probability. Natural language processing in text mining or text classification can be used better to analyze the parts of a sentence, it is used in the scientific community for condensing large articles into a summarized form, in health care industry it is used to make divisions of patients according to their ailments and lot more so that it's easy to access a patient's health care records.

References

[1] Jorge R, Javier M, Raúl M, Octavio L and Armando L, 2020, A one-class classification approach for bot detection on twitter, Computers and Security, 91, pp. 1-14.
[2] Rodríguez-Ruiz J, Mata-Sánchez J, Monroy R, Loyola-González O, and López-Cuevas A, 2020, A one-class classification approach for bot detection on twitter, Computers and Security, 91, pp. 1-14.
[3] Rahman M, Likhon A, Rahman A and Choudhury M, 2019, Detection of fake identities on twitter using supervised machine learning, PhD dissertation, Brac University.
[4] Beskow D and Carley K, 2018, Bot conversations are different: leveraging network metrics for bot detection in twitter, Proc. Int. Conf. On Advances in Social Networks Analysis and Mining (Barcelona, Spain), pp. 825-832.
[5] Knauth J, 2019, Language-agnostic twitter-bot detection, Proc. Int. Conf. on Recent Advances in Natural Language Processing (Varna, Bulgaria), pp. 550-558.
[6] Daouadi K, Rebaï R, and Amous I, 2020, Real-time bot detection from twitter using the twitterbot+ framework, Journal of Universal Computer Science, 26, pp. 496-507 [4].
[7] Dabiri S, and Heaslip K, 2019, Developing a twitter-based traffic event detection model using deep learning architectures, Expert Systems with Applications, 118, pp. 425-439.
[8] Gabryel M, Damaševičius R, and Przybyszewski K, 2018, Application of the bag-of-words algorithm in classification

the quality of sales leads, Proc. Int. Conf. on Artificial Intelligence and Soft Computing, Springer (Cham), pp. 615-622.
[9] Sahoo S, and Gupta B, 2019, Hybrid approach for detection of malicious profiles in twitter, Computers and Electrical Engineering, 76, pp. 65-81.
[10] Beğenilmiş E, and Uskudarli S, 2018, Organized behavior classification of tweet sets using supervised learning methods, Proc. Int. Conf. on Web Intelligence, Mining and Semantics (Novi Sad Serbia), pp. 1-9.
[11] Ali D, Abadi M, and Dadfarnia M, 2018, Socialbothunter: botnet detection in twitter-like social networking services using semi-supervised collective classification, Proc. Int. Conf. on Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, and Big Data Intelligence and Computing and Cyber Science and Technology Congress (Athens, Greece), pp. 496-503.
[12] Torres J, Comesaña C, and García-Nieto P, 2019, Machine learning techniques applied to cybersecurity, International Journal of Machine Learning and Cybernetics, 10, pp.2823-2836 [10].
[13] Satija T, and Kar N, 2019, Detecting malicious twitter bots using machine learning, Proc. Int. Conf. on Computational Intelligence, Security and Internet of Things, Springer (Singapore), pp. 182-194.
[14] Kenta M, Takashi I, and Masayuki G, 2011, A proposal of extended cosine measure for distance metric learning in text classification, Proc. Int. Conf. on Systems, Man, and Cybernetics, Anchorage (AK, USA), pp. 1741-1746.
[15] Kaggle, accessed on May 23rd 2020, https://www.kaggle.com/charvijain27/detecting-twitter-bot-data.