Bot Detection Application on Twitter Using Machine Learning with Random Forest Classifier Algorithm

Aqilah Aini Zahra[1], Widyawan[2], Silmi Fauziati[3]

***Abstract*—A Twitter bot is a Twitter account programmed to automatically do social activities by sending tweets through a scheduling program. Some bots intend to disseminate useful information such as earthquake and weather information. However, not a few bots have a negative influence, such as broadcasting false news, spam, or become a follower to increase an account's popularity. It can change public sentiments about an issue, decrease user confidence, or even change the social order. Therefore, an application is needed to distinguish between a bot and non-bot accounts. Based on these problems, this paper develops bot detection systems using machine learning for multiclass classification. These classes include human classes, informative, spammers, and fake followers. The model training used guided methods based on labeled training data. First, a dataset of 2,333 accounts was pre-processed to obtain 28 feature sets for classification. This feature set came from analysis of user profiles, temporal analysis, and analysis of tweets with numeric values. Afterward, the data was partitioned, normalized with scaling, and a random forest classifier algorithm was implemented on the data. After that, the features were reselected into 17 feature sets to obtain the highest accuracy achieved by the model. In the evaluation stage, bot detection models generated an accuracy of 96.79%, 97% precision, 96% recall, and an f-1 score of 96%. Therefore, the detection model was classified as having high accuracy. The bot detection model that had been completed was then implemented on the website and deployed to the cloud. In the end, this machine learning-based web application could be accessed and used by the public to detect Twitter bots.**

***Keywords*—Bot Detection, Multiclass Classification, Machine Learning, Supervised Learning, Twitter.**

I. INTRODUCTION

One of the popular social media apps today is Twitter. It was launched in 2006 by Jack Dorsey [1]. Twitter is a micro-blog service in which the users can send short messages, which then called tweets. Each tweet is limited to 140-character only. However, the simplicity and capacity to send tweets as often as possible become the additional value from this application. According to statistics in 2018, there have been 326 million Twitter active users each month, with an average of 500 million tweets sent every day [2].

Bots or automated programs are growing in popularity alongside Twitter's popularity. They are created using the Twitter Application Programming Interface (API). A study by the United States Commission on Exchange and security in 2016 found at least 8.5% of active Twitter users were bots [3].

[1,2,3] *Department of Electrical and Information Engineering, Faculty of Engineering, Universitas Gadjah Mada, Jln. Grafika No.2, Kampus UGM Yogyakarta 55281 INDONESIA (tlp: 0274-552305; email: [1]aqilah.aini.z@mail.ugm.ac.id, [2]widyawan@ugm.ac.id, [3]silmi@ugm.ac.id)*

Bots have various objectives. It is undeniable that some bots have benefits in disseminating weather or earthquake information [4]. However, many malicious bots exist too. They harm by broadcasting malware links [5], disrupting other users, broadcasting terrorist propaganda, spam, news lies, making hoaxes, and doing political campaigns. A massive tweet volume is capable of polluting users' timeline, changing user perception, damaging user confidence, affecting the stock market, and even being able to undermine social order. Therefore, basic knowledge to distinguish between types of bot accounts and non-bot accounts is required.

Conventionally, identifying bot accounts and not bots can be carried out by observing the activity pattern on an account. For example, noticing that a particular account does more retweeting than creating original tweets, writes many tweets, but only has a few followers. Besides, the account also does not have a biography, a profile picture, and writes the same tweet content as another user at the same time. However, such cognitive approaches are rated inefficient and merely focus on precision.

Therefore, an approach to detecting bots with machine learning is created. The machine learning models are used because of their capacity to analyze extensive data based on parameters. According to a study, the random forest algorithm has the best accuracy, which is 95%, compared to Naïve Bayes Multinomial algorithm (70%), Naïve Bayes Gaussian algorithm (68%), and Logistic Regression algorithm (52%) [6]. Therefore, the random forest classifier algorithm is selected in this paper. The system classified the multiclass classification by classifying accounts into four different classes, which are human, informative, spammers, and fake followers. Then, a completed machine learning model was inserted into the web *application.* The website serves as an interface for end-users to use machine learning systems. In the end, the system underwent a deployment process to the cloud service.

II. THEORETICAL FRAMEWORK

A. Classification Type

Classification is a process of assigning a category or label that has been defined as data that does not yet have a category. In general, there are three types of the data classification process, namely binary, multiclass, and multilabel classification [7].

- Binary classification is a process of classifying each element in a group into two groups or categories.
- Multiclass classification is a classification process involving more than two classes. However, the multiclass classification creates an assumption that each given sample is categorized into a single label (mutually exclusive).

- A cultilabel classification is a classification process that puts samples into a set of targets. This classification predicts the properties of data that are not mutually exclusive. Examples of this classification are found in document classification.

B. Machine Learning

Machine learning is a technique that enables the system to learn from data compared to using direct programming so that it can deliver relevant results [8].

C. Random Forest Algorithm

Random forest was first introduced by Leo Breiman [9]. The random forest classifier is the development of the decision tree. It consists of a combination of many decision trees, with each tree relying on independent random vector values with an equivalent distribution of each tree [9].

D. Twitter Social Media

Twitter is a micro-blogging social network that allows its users to send and read short messages up to 140 words, which are then called tweets [10]. Jack Dorsey founded this social media in 2006. Unlike social media such as Facebook or MySpace, on Twitter, the relationship between to follow an account and the followers are not reciprocal. It means that an account can follow other accounts without automatically be followed by the account it follows.

E. Bots and Twitter Bot Types

In general, bot means an application that performs tasks automatically. In social media, bot domain is a social media accounts programmed to perform social media activities automatically, so they look like real humans. According to research from the University of Southern California, at least 9% to 15% of active Twitter users are bots [11]. Until 2017, there were 319 million active users each month. It means there are almost 48 million bot accounts spread on the Twitter social network.

Factors that influence bot growth include Twitter API support, bot development cycles that can be created quickly, Twitter public platforms, and the flexibility to create as many accounts as possible. According to the Digital Forensic Research (DFR) of the Atlantic Council Lab, there are several features indicating that an account is a bot, including amplification, anonymity, activity, similarity, and description of "bot" in the account [12]. Whereas the Twitter bot types based on account activity are as follows.

- Informative, i.e., a bot that functions to disseminate information to users. For example, bots that publish facts, earthquake information, and write poetry content as well as humor content [13].
- Spammers, i.e., bots that work to broadcast spam content [14].
- Fake Followers, i.e., bots that act as shadow followers for an account. The purpose of using fake followers is to create an image that an account seems to have prominent popularity [14].

F. Flask

Flask is a Python micro-framework that provides the basic functionality of a web framework [15]. It allows plug-ins to be added to add functionality and features. It is named a microframework because it has a very simple core functionality, yet it can be expanded by adding plug-ins.

G. Heroku

Heroku is a cloud computation-based application that is useful for doing deployment and management services. As a Platform as a Service (PaaS), Heroku provides a service that allows running scripts directly without requiring complex configuration, so developers can focus on application code development without the need to think about architecture and servers.

H. Waterfall Method

This method emphasizes the planning and scheduling process before starting the system development. This method is best used if the product definition is clear, the project is short-lived, technology is known, and resources are available. The advantages of this method are organized documentation, proper to be used for known needs, and easily understood. The weakness of this method is the need for appropriate management, and small mistakes will be a big problem if not noticed from the beginning of development, high risk, and not a good model for intricate work.

I. Classification Evaluation

A metric evaluation is a set of metrics used to measure a classifier's performance. Different metrics measure different classifier characteristics. Evaluation metrics consist of three types, namely, threshold, opportunity, and ranking metrics

III. METHODOLOGY

The reference method employed in this paper is the waterfall method. This method was selected because it was easy to understand. Moreover, the system requirements were precise, the technology was known, and the documentation was organized. Whereas the research process flow was carried out based on the flowchart like in Fig. 1.

In the initial stage, there was an identification of problems occurring at present. Then, a literary study was carried out to collect references related to bot detection systems. After that, an observation was made on the existing bot detection system, namely Botometer, Bot or not, and I Spot a Bot. The next step was to create a system design, then downloaded data through public repositories and did the crawling to the Twitter API to download supporting data.

The downloaded data was then a research dataset. Furthermore, there was a data pre-processing in the form of data cleansing and feature engineering. A feature engineering was carried out by extracting the basic parameters of the dataset and creating the derived parameters through a calculation process. This feature set then became the three classes of analysis, namely temporal analysis, tweet content analysis, and user profile analysis. Then, the data were aggregated according

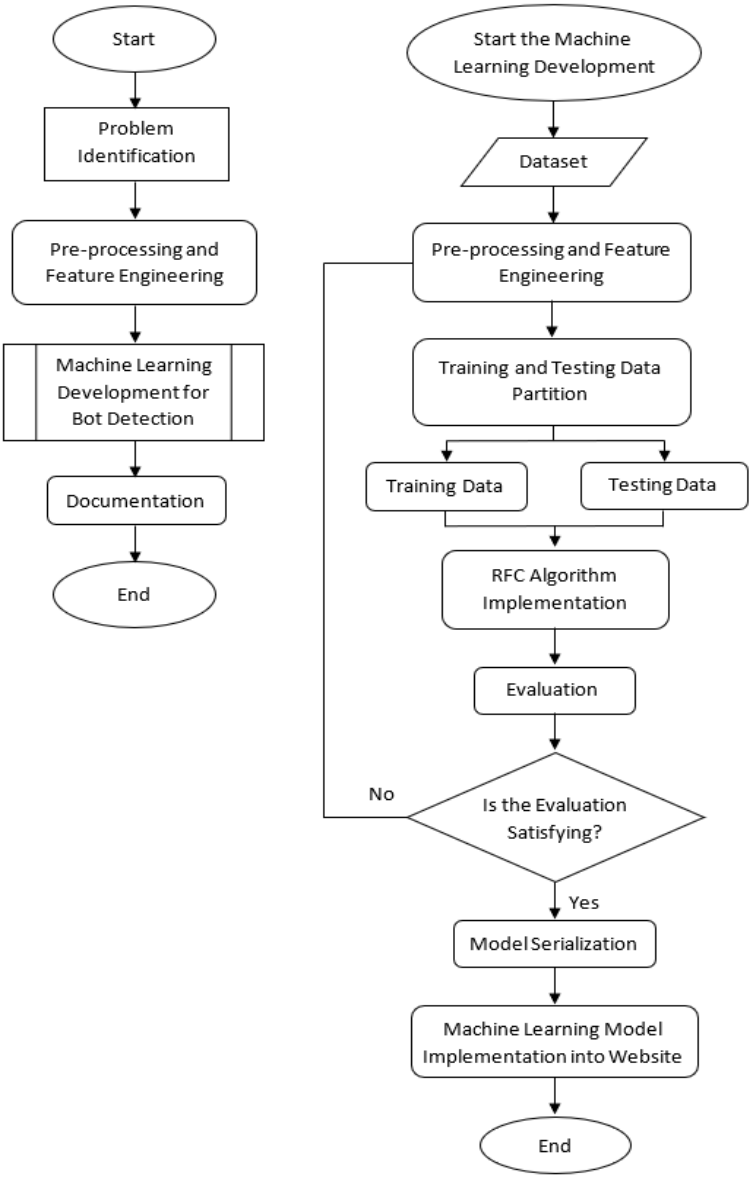

Fig. 1 Flowchart of research implementation.

TABLE I
PARTITION AND ACCURACY RATIO

Partition Ratio	Accuracy
50:50	95.12%
60:40	96.36%
70:30	96.57%
80:20	96.79%
90:10	93.16%

to the average value of each username. A scaling was applied to these data aggregation to change the range of data values to between 0-100. The next step was to do data partition, which divided data into training data and test data. Throughout the implementation process, it was recognized that the 80:20 partition produced the best accuracy. Table I shows partition and accuracy ratio.

After the data was partitioned with a ratio of 80:20, training data was implemented on the random forest classifier algorithm implementation. The obtained results were then tested for accuracy based on the threshold feature importance. This process was employed to re-select features to achieve the highest accuracy based on existing parameters. After that, the algorithm implementation was repeated with the selected parameters. The accuracy results were evaluated, applying the multiclass confusion matrix using (1) to (4).

$$mean\ of\ accuration = \frac{\sum_{i=1}^{k} \frac{tp_i + tn_i}{tp_i + tn_i + fp_i + fn_i}}{k} \quad (1)$$

$$precision_{\mu} = \frac{\sum_{i=1}^{k} tp_i}{\sum_{i=1}^{k} (tp_i + fp_i)} \quad (2)$$

$$recall = \frac{\sum_{i=1}^{k} tp_i}{\sum_{i=1}^{k} (tp_i + fn_i)} \quad (3)$$

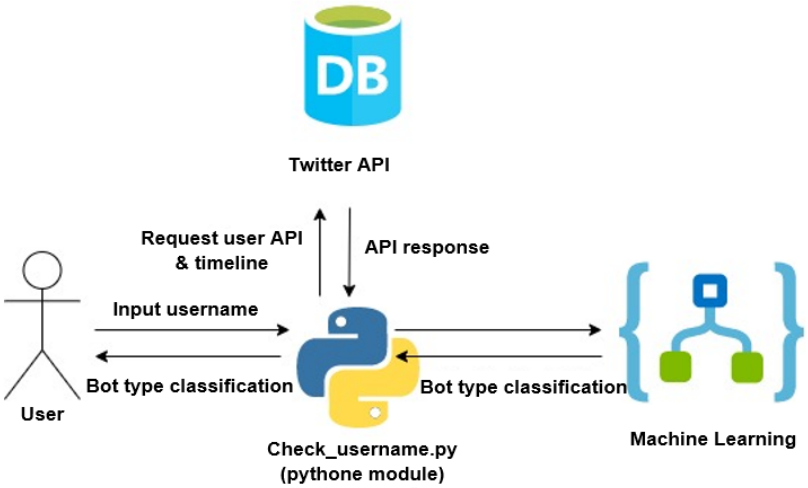

Fig. 2 Bot detection system scheme.

TABLE II
DATASET DETAILS

No.	Type of Data	Number of Accounts	Number of Tweets	Percentage
1	Username data of spammer account	859	42,950	36.82%
2	Username data of fake follower accounts	334	16,700	14.32%
3	Username data of informative accounts	293	14,650	12.56%
4	Username data of the original account	847	42,350	36.31%
Total		**2,333**	**116,650**	**100%**

$$F1 - score_{\mu} = \frac{2 \times precision_{\mu} \times recall_{\mu}}{precision_{\mu} + recall_{\mu}} \quad (4)$$

where,

tp_i = true positive for $C_{i\ class}$
tn_i = true negative for $C_{i\ class}$
fp_i = false positive for $C_{i\ class}$
fn_i = false negative for $C_{i\ class}$
k = total amount.

The constructed machine learning model was stored in the form of pickle *(.pkl)*. This process is called serialization. It was carried out because the bot detection model would be implemented on the website.

The next process was to utilize the Flask framework to create a website with a bot detection model as a back-end. After that, the next step was creating an interface as a front-end using HTML, CSS, and Javascript markup languages. Finally, the system that had been constructed was deployed to the cloud services using the Heroku platform. The bot detection system scheme was made according to Fig. 2.

TABLE III
UTILIZED PARAMETERS

Types of Analysis	Parameter's Name	Type
Profile Analysis	friends_count	Basic
	Verified	Basic
	followers_count	Basic
	URL	Basic
	statusses_count	Basic
	default_profile_image	Basic
	default_profile	Basic
	favorites_count	Basic
	listed_count	Basic
	Location	Basic
	contains_bot_name	Derivative
	length_of_bio	Derivative
	age_in_days	Derivative
	ratio_favorites_per_age	Derivative
	ratio_statusses_per_age	Derivative
	ratio_friends per_followers	Derivative
	Reputation	Derivative
Tweet Content Analysis	tweet_retweet_status	Derivative
	Tweet_retweet_count	Derivative
	Tweet_favorite_count	Derivative
	Tweet_hashtag_count	Derivative
	Tweet_urls_count	Derivative
	Tweet_user_mentions_count	Derivative
	Avg_word	Derivative
	Word_count	Derivative
	Char_count	Derivative
	Lexical	Derivative
Temporal Analysis	Entropy	Derivative

IV. RESULTS AND DISCUSSIONS

A. System Development

1) Data Collection: The initial process in the system development was by collecting data. The labeled username data were downloaded to a public repository. Then, a crawling

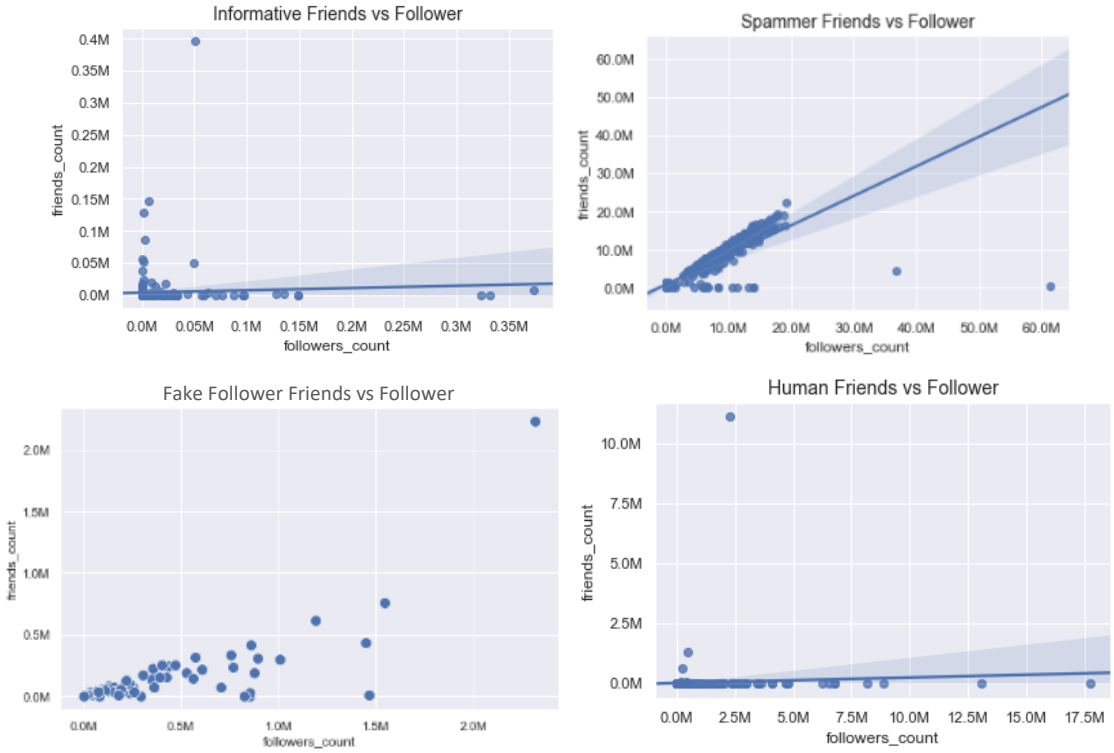

Fig. 3 A scatter plot on a comparison of the number of accounts following and followers in the dataset.

process was carried out to obtain additional data such as metadata, tweets, and other attributes using the Tweepy library. The data collection process employed the Jupyter Notebook as an Integrated Development Environment (IDE). The dataset details are described in Table II.

2) Pre-processing: The pre-processing process included a data cleansing process, i.e., clearing data by removing or replacing values of the empty, invalid, or Not a Number (NaN) values. The next step was to do deletion on the duplicate data.

The following process was defining the utilized parameters as a set analysis feature. Processing was carried out to adjust the information structure so that it could be processed with the random forest classifier algorithm. This algorithm is only capable of processing numerical values. The utilized parameters are listed in Table III.

After the parameter analysis was carried out, the significant differences between each category were recognized according to the following and follower parameter through a graph in Fig. 3. After the pre-processing, a data scaling process was carried out to change the value range to 100. This aimed to equalize the values between parameters.

3) Data Partitioning: The data partitioning process aimed to separate the dataset into training data and test data with a ratio of 80:20. This partitioning process was carried out using a train_test_split() method in the sklearn library.

4) Implementation of the Random Forest Classifier Algorithm: The algorithm implementation was carried out by implementing the algorithm in the training data and testing it with the test data. This algorithm can be accessed in the scikit-learn using Python version 3.7. This algorithm implementation also included weighing in each class due to uneven data distribution.

TABLE IV
MULTICLASS CONFUSION MATRIX

		Prediction				Total
		F	H	I	S	
Actual	F	61	5	1	1	88
	H	1	155	1	2	261
	I	1	1	56	0	78
	S	0	2	0	180	273
		Total Amount				700

Note: F= Fake Follower, H= Human, I= Informative, S = Spammer

5) Feature Importance: Based on the measurement of feature importance scores, the feature with the highest weight is the "favorites", while the lowest is "contains_bot_name". When visualized, the score will look like Fig. 4. By performing an accuracy test based on the threshold feature importance in Fig. 4, a graph can be drawn as shown in Fig. 5. It shows that the accuracy is relatively stable until the threshold reaches 0.01. Therefore, to obtain the highest accuracy, the feature was selected by taking 20 feature sets with the most significant weight to form the final detection model.

6) Evaluation of Bot Detection Models: The evaluation of bot detection models employed a multiclass confusion matrix, as shown in Table IV. The generated accuracy value is 96.79%, while the classification process creates values of precision, recall, and f-1 corresponds to Table V.

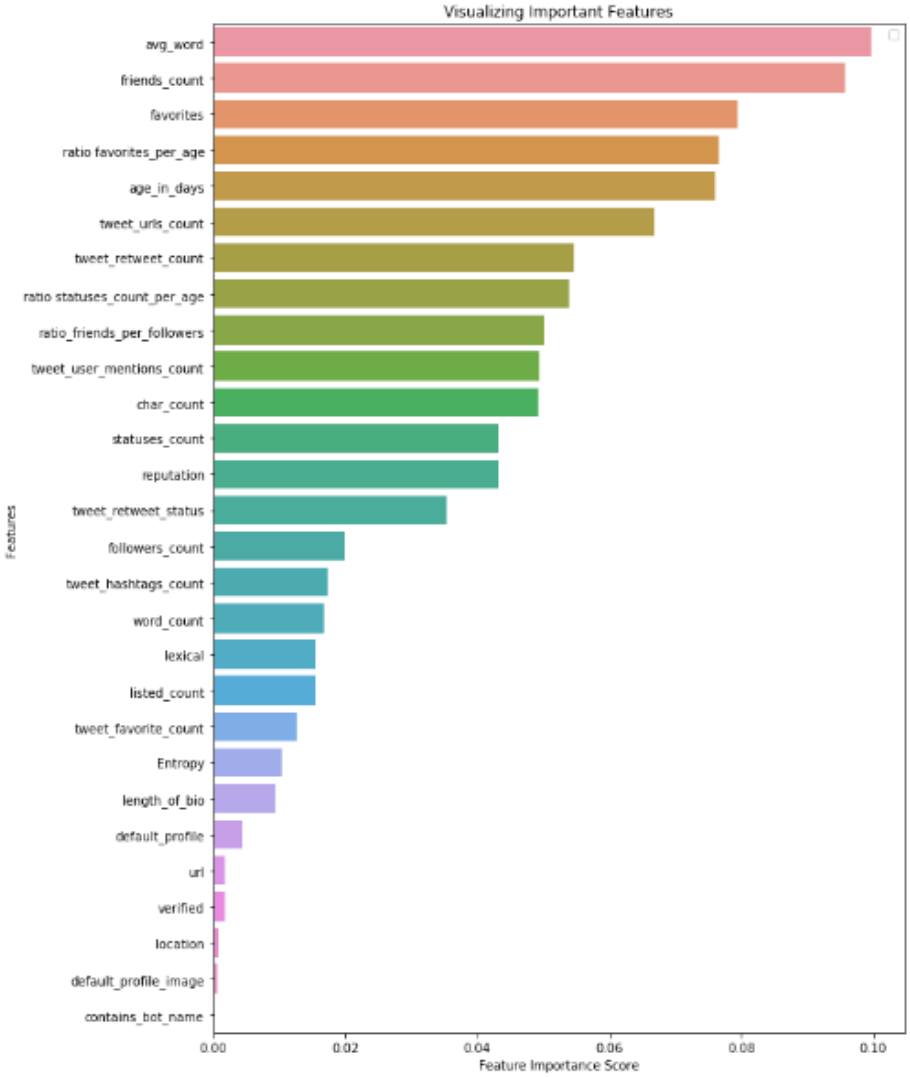

Fig. 4 Feature importance result of the test parameter.

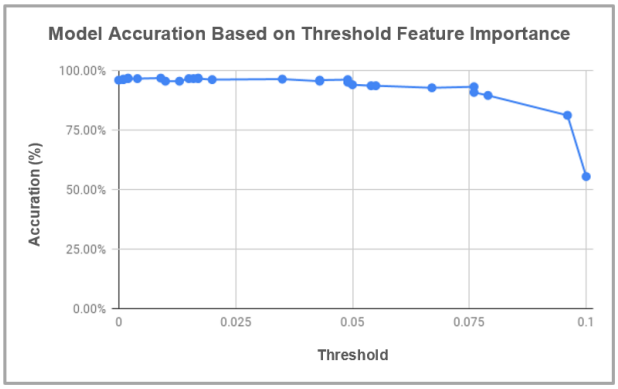

Fig. 5 Accuracy graphs based on threshold feature importance.

7) Model Serialization: The model serialization generated a pickle file that can be deserialized, and this adjusted to the platform of implementation's destination. In this paper, the file would be implemented on a website with the Flask framework; therefore, the primary programming language was Python.

TABLE V
SUMMARY OF CLASSIFICATION EVALUATION

	Precision	Recall	f1	Support
Fake follower	0.97	0.90	0.93	68
Human	0.95	0.97	0.96	159
Informative	0.97	0.97	0.97	58
Spammer	0.98	0.99	0.99	182
Average	0.97	0.96	0.96	

8) Website Implementation: The website implementation was carried out with a Flask micro framework using the Python programming language. Fig. 6 shows the display of the web application's main page. Through the main page, users could input their Twitter account username without using an '@'. After that, the website server would receive the input and process it through machine learning.

After the process was complete, the system produced an output in the form of a profile and prediction category class results, such as in Fig. 7. The system was then deployed into

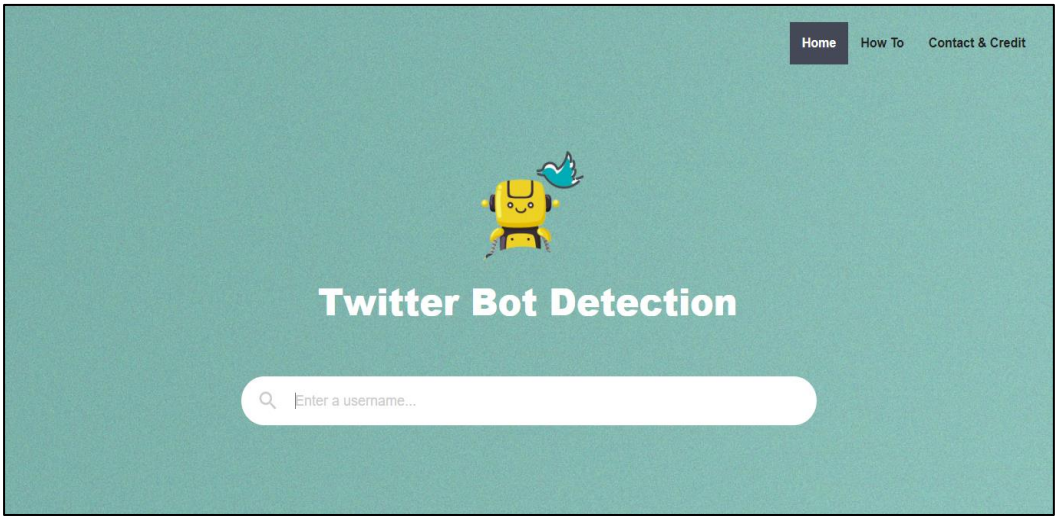

Fig. 6 The Twitter Bot Detection main page.

Fig. 7 Image of the prediction result.

the Heroku cloud service. Through this platform, the web application can be accessed at http://botclassifier.herokuapp.com.

V. Conclusions

Based on the process and obtained results in the system development stage as well as the machine learning evaluation, several conclusions can be formulated as follows. The development of a bot detection system using a machine learning with the random forest classifier has been successfully developed. The features of this system include the classification of four class types, namely non-bot (human) class, fake followers bot, spammers bot, and informative bot, and this can be accessed via the web. The training data partition and test data of 80:20 produce the highest accuracy (96.795) compared to the test data partition of 50:50 (95.12%), 60:40 (96.36%), 70:30 (96.57%), and 90:10 (93.16%). The feature selection process based on the feature importance score shows that only 17 features with a threshold of 0.017 contribute to increasing the high accuracy. The multiclass classification process using the random forest classifier based on the 17 feature sets produces an accuracy of 96.79%, 97% precision, 96% recall, and an f-1 score of 96%. Thus, the detection model is classified as having high accuracy.

References

[1] (2014) "Jack Dorsey Biography," [Online], https://www.biography.com/business-figure/jack-dorsey, access date: 3-Sep-2019.

[2] (2018) "Twitter by the Numbers (2018): Stats, Demographics & Fun Facts," [Online], https://www.omnicoreagency.com/twitter-statistics/, access date: 13-Jun-2019.

[3] V.S. Subrahmanian, A. Azaria, S. Durst, V. Kagan, A. Galstyan, *et al.*, "The DARPA Twitter Bot Challenge," *Computer (Long. Beach. Calif.)*, Vol. 49, No. 6, pp. 38–46, 2016.

[4] (2019) "Nain Weather Bot (@NainWxPxBot) | Twitter." [Online], https://twitter.com/nainwxpxbot, access date: 3-Sep-2019.

[5] K. Zetter (2009) "Trick or Tweet? Malware Abundant in Twitter URLs," [Online], https://www.wired.com/2009/10/twitter-malware/, access date: 3-Sep-2019.

[6] M. Haidermota, "Classifying Twitter User as a Bot or Not and Comparing Different Classification Algorithms.," *Int. J. Adv. Res. Comput. Sci.*, Vol. 9, No. 3, pp. 29–33, 2018.

[7] M. Hossin and Sulaiman, "A Review on Evaluation Metrics for Data Classification Evaluations," *Int. J. Data Min. Knowl. Manag. Process*, Vol. 5, No. 2, pp. 1-11, 2015.

[8] J. Hurwitz and D. Kirsch, *Machine Learning For Dummies*, Hoboken, USA: John Wiley & Sons, Inc., 2018.

[9] L. Breiman, "Random Forests," *Mach. Learn.*, Vol. 45, pp. 5–32, 2001.

[10] H. Kwak, C. Lee, H. Park, and S. Moon, "What is Twitter, a social network or a news media?," *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, 2010, pp. 591–600.

[11] M. Newberg (2017) "Nearly 48 million Twitter accounts could be bots, says study," [Online], https://www.cnbc.com/2017/03/10/nearly-48-million-twitter-accounts-could-be-bots-says-study.html, access date: 27-Aug-2019.

[12] (2017) "#BotSpot: Twelve Ways to Spot a Bot," *Medium*, [Online], https://medium.com/dfrlab/botspot-twelve-ways-to-spot-a-bot-aedc7d9c110c, access date: 27-Aug-2019.

[13] S. Schreder (2018) "10 Twitter bots that actually make the internet a better place - Internet Citizen." [Online], https://blog.mozilla.org/internetcitizen/2018/01/19/10-twitter-bots-actually-make-internet-better-place/, access date: 19-Dec-2019.

[14] A. Khalil, H. Hajjdiab, and N. Al-Qirim, "Detecting Fake Followers in Twitter: A Machine Learning Approach," *Int. J. Mach. Learn. Comput.*, Vol. 7, No. 6, pp. 198–202, 2018.

[15] F.A. Aslam and H.N.M.J.M.M.M.M.A. Gulamgaus, "Efficient Way Of Web Development Using Python And Flask," *Int. J. Adv. Res. Comput. Sci.*, Vol. 6, No. 2, pp. 54–57, 2015.