Twitter Bot Account Detection Using Supervised Machine Learning

Febriora Nevia Pramitha
Cryptographic Engineering
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
febriora.nevia @student.poltekssn.ac.id

Raden Budiarto Hadiprakoso
Cryptographic Engineering
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
raden.budiarto@poltekssn.ac.id

Nurul Qomariasih
Cryptographic Engineering
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
nurul.qomariasih@poltekssn.ac.id

Girinoto
Cyber-Security Engineering
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
girinoto@poltekssn.ac.id

2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) | 978-1-6654-0151-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/ISRITI54043.2021.9702789

***Abstract*— Twitter is a primary social media platform gaining popularity among social networking websites at an alarming rate. Twitter's popularity and relatively open nature make it an excellent target for automated programs known as bots, which are computer programs that run automatically. In addition to spamming, bots can be used for various purposes, such as inducing conversations to change the topic of discussion, modifying user popularity, contaminating materials to spread misinformation, and conducting propaganda. This study's goal was to provide a fresh perspective on estimating the possibility of an account being identified as a bot by applying Machine Learning algorithms to a variety of scenarios. Both Random Forest and XGBoost algorithms are used in this application. The inquiry began with exploratory data analysis to determine the current status of the dataset. Next comes the process of model engineering, which involves the steps of requirement gathering and specification, feature selection and optimization, hyperparameter tweaking, and algorithm benchmarking. The findings of this investigation suggest that the XGBoost algorithm outperforms Random Forest, with an accuracy of 0.8908 for XGBoost and 0.8762 for Random Forest.**

Keywords*— bot detection, twitter bot, machine learning, random forest, xgboost

I. INTRODUCTION

Today, Twitter is one of the most widely used social media sites. According to Statista, Indonesia's Twitter user base will reach approximately 16.32 million in 2021 [1]. Twitter is a free service that is easily accessed on mobile devices. This reason is one factor that motivates individuals and businesses to engage with and share relevant material on the Twitter network. Twitter allows users to write and read 140-character tweets. Tweets, or short communications, are primarily public and can be read by other users [2]. Apart from being used by human people, Twitter is also home to many automated programs, colloquially referred to as Twitter bots. Twitter's popularity and open nature make it a desirable target for bots [3].

Twitter bots take advantage of Twitter's characteristics to conduct illegal activities such as black electoral campaigns [4], hate speech [5], and propaganda [6]. On the other hand, Twitter bots can be used to distribute unwanted advertisements or content. Twitter bots live with human users on the platform, concealing their automated nature through impersonation. As a result, identifying bots on Twitter is critical for preserving the integrity of tweet content. Detecting bots is crucial for identifying fraudulent users and safeguarding legitimate users against misinformation and evil intent.

According to [7], bots impersonating humans have been used to control debates, alter user popularity, pollute content, distribute misinformation, and carry out propaganda. Machine learning is capable of detecting this type of behavior [8]. Twitter elements such as tweet content, when users publish tweets, and account attributes can all be utilized to determine if an account is a bot or a human [9].

Numerous research efforts have been made to identify Twitter bots. Numerous bot detection datasets have been proposed during the last decade. Among these datasets is the pronbots [10] dataset, which contains Twitter bots and treats bot detection as outlier discovery. Meanwhile, most other datasets, such as Varol-icwsm [11] and Cresci-17 [12], include human users and bots that perform binary classification tasks. Cresci-17 is a frequently used dataset that has 2,764 human accounts and 7,049 bot accounts.

Bot detection is divided into two approaches: account-level detection and tweet-level detection. The dataset was collected via Twitter API scraping, as previous research' datasets were restricted by Twitter regulations and could only be used for account-level bot detection. However, account-level bot identification is more appropriate when employing feature-based Machine Learning in conjunction with classic Machine Learning. Machine Learning models are utilized because they can analyze large amounts of data using parameters [8]. This program was intended to provide a service for evaluating Twitter accounts. It takes input in a Twitter account's public profile, retrieves the account's most recent activity, and then calculates possible bot scores [6].

This study presents a Machine Learning technique based on the benchmarking algorithms Random Forest (RF) and Extreme Gradient Boost (XGBoost) to obtain the optimal model for bot account detection. The algorithm was chosen due to its popularity as measured by the number of competitions won on Kaggle [13]. Additionally, the objective of this study is to identify characteristics that differentiate real Twitter accounts from bot accounts. Additionally, the discussion section of this study is structured as follows: in section 2, we evaluate related research, and in section 3, we determine the research approach used. We execute a data analysis test in Section 4. Finally, in Section 5, we will conclude the work and recommend further research.

Authorized licensed use limited to: Bursa Uludag Universitesi. Downloaded on January 30,2023 at 01:29:28 UTC from IEEE Xplore. Restrictions apply.

II. Related Works

Spearman correlation is used in the study [14] to conduct the appropriate feature extraction. The current work constructs a model using four algorithms: Decision Tree, Multinomial Naive Bayes, Random Forest, and Bag of Words. The most effective learning models are applied to pre-processed data in real-time. The result is either 0 or 1, with 1 indicating the presence of a bot and 0 indicating the absence of a bot.

Ref. [15] provides a system for detecting bots using real-time Twitter processing, which enables efficient analysis that is scalable. This study discovered that picking training data subsets carefully led to higher model accuracy and generalization than training on all available data.

Paper [16] compared the tree-boost method XGBoost to the more classic MART algorithm and made a case for why XGBoost wins so many Kaggle competitions. Furthermore, this paper discusses the strategies or procedures proposed and their impact on the model. This paper suggests that adaptive tree improvement can be used to determine the model's local environment.

Neural networks are increasingly being used for Twitter bot detection due to the introduction of deep learning. Stanton et al. [17] utilized the Generative Adversarial Network (GAN) for Twitter bot detection. The GAN algorithm is used to detect anomalies that occur in a Twitter bot account. Wei et al. [18] identified bots using the Recurrent Neural Network (RNN) and tweet semantics. Kudugunta et al. [19] proposed a Long Short-Term Memory (LSTM)-based bot detecting based on user attributes and tweet semantics. However, the deep learning model's drawback is black-boxed, with no explanation for the resultant output. This study applies machine learning techniques to determine the relevance of features that identify humans from bot Twitter accounts.

III. Method

This study began with the collection of research reference data, in this case through the examination of published literature. Kaggle data set obtained. This dataset only contains human or bot ids and labels; scraping is performed using the Twitter API and stored in a CSV file to obtain profile feature information from an id or account. Previously, while scraping data via the Twitter API, it was required to submit an API key request to Twitter in the form of a consumer key, consumer secret, access token, and access token secret, which served as authentication criteria for scraping Twitter data.

Access Token and Token Secret are one-of-a-kind values, similar to personal identification numbers (pins), that must be kept private and only known to the Twitter account owner. Due to Twitter's restrictions, the profile of an account collected using Twitter API scraping is limited. Scraping is carried out using Python and the Beautiful Soup library.

After obtaining a dataset comprising various attributes associated with user accounts, data pre-processing will occur. At this step, traits that are deemed irrelevant to classification will be eliminated. Then comes Exploratory Data Analysis (EDA), the preliminary examination of the data to establish the most appropriate analysis model.

Predictive modeling is then performed utilizing a variety of machine learning techniques, including Random Forest and XGBoost. The engineering of features is carried out throughout the modeling process. In this modeling, SMOTE is used to overcome data imbalances. This process is required to ensure that the categorization findings are not skewed, given the highly imbalanced human and bot accounts sample size. The following step in optimizing the model's algorithmic usage is hyperparameter tuning. The best algorithm can be utilized to assess the feature relevance of the features in the used dataset. Then, using metric evaluation, benchmark the model's accuracy results to determine the optimal model for detecting bot accounts on Twitter.

We will determine which algorithm is the best accurate at recognizing a Twitter account as a bot or not based on the benchmarking results. Accuracy, recall, precision, and F1 score are the metrics used in the benchmarking model test. The formulas in Equations 1-4 are used to calculate the accuracy, precision, recall, and F1 scores.

$$\text{Accuracy} = \text{TP} + \text{TN}/\text{TP} + \text{FP} + \text{FN} + \text{TN} \quad (1)$$

$$\text{Precision} = \text{TP}/\text{TP} + \text{FP} \quad (2)$$

$$\text{Recall} = \text{TP}/\text{TP} + \text{FN} \quad (3)$$

$$\text{F1 Score} = 2 * (\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision}) \quad (4)$$

TP (*True Positive*), TN (*True Negative*), FP (*False Positive*), FN (*False Negative*)

IV. Result and Dicussion

The first step in analyzing the data is to study the characteristics of the data. At this stage, the dataset is imported which is used for data analysis, where the dataset is in the form of a comma separated values (csv) file with the extension. A brief summary of the data frame type or data type of each feature can be found using the pandas dataframe.info () syntax, the result is that there are 15 features as follows:

1. id, is an integer representation of the unique identifier for the user. id of type int.

2. u_name, or username is a name that identifies an account where its function is very important to distinguish one user from another.

3. screen_name, is the display name or alias on an account, screen_names are unique but can change, screen_name is an object type.

4. verified, to find out whether an account is verified or not, verified is a boolean type.

5. geo_enabled, to attach user geographic data, geo_enabled is a boolean type.

6. default_profile, if "True", indicates that the user has not changed the theme or background of their user profile. default_profile of type boolean.

7. default_profile_image, indicates that the user has not uploaded their own profile picture and the default image is used instead. default_profile_image of type boolean.

8. favorites_count, is the number of tweets that the user liked during the active period of the account. favorites_count of type int.

9. followers_count, the number of followers currently on an account. followers_count of type int.

10. friends_count, is the number of users following an account (followers or "following"). friends_count of type int.

11. statuses_count, is the number of tweets (including retweets) issued by the user. statuses_count of type int.

Authorized licensed use limited to: Bursa Uludag Universitesi. Downloaded on January 30,2023 at 01:29:28 UTC from IEEE Xplore. Restrictions apply.

12. network, a feature that is processed from the friends_count and followers_count features. network type float.

13. average_tweets_per_day, is the average tweet issued per day, average_tweets_per_day type int.

14. created_at, is the time of creation of an account, created_at type object.

15. account_type, is the type of an account, whether bot or non-bot (human), account_type type object.

In the following stage, the data will be explored, analyzing the relationship or pattern on the data. This is primarily intended to find out what features most distinguish an account from being a human or a bot.

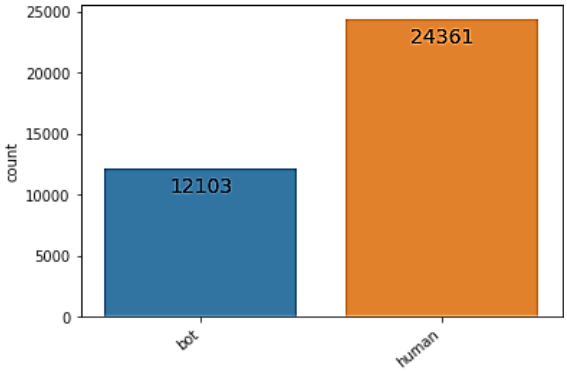

Fig. 1. Number of bot and human accounts

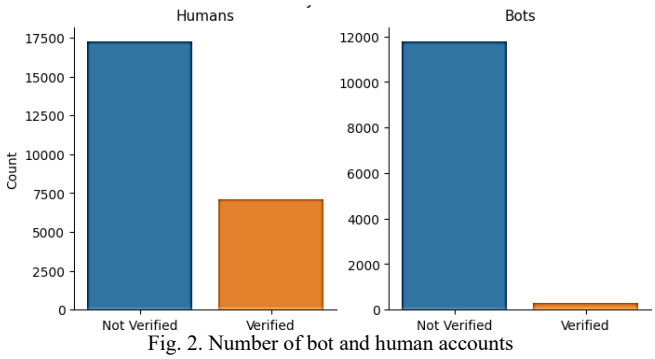

Fig. 2. Number of bot and human accounts

The data comparison chart in Fig. 1 is used to determine the number of inputs required for each account type. Bot accounts numbered 12103, whereas non-bot (human) accounts numbered 24631. These findings show an imbalanced distribution of classes. Figure 1 depicts a graphical representation of the account type and validated fields. There were 313 verified bot accounts and 11789 unverified bot accounts. The overall number of verified non-bot (human) accounts was 7086, whereas the total number of unverified accounts was 17275. Fig. 2 illustrates the number of account types by verification status.

The graph in Fig. 3 depicts the dataset's distribution by number of followers, number of friends, network status, and number of tweets per day. It is clear from Fig. 3 that. Human accounts have a higher curve than bot accounts, indicating that human accounts have a greater potential to have more followers. Additionally, the curve for human accounts is steeper than the curve for bot accounts, indicating that human accounts have a greater proclivity to amass followers than bot accounts.

The distribution graph in Fig. 3 is based on the number of followers and following. Based on the first three graphs, it is clear that human accounts have a higher curve than bot accounts, indicating that human accounts tend to have more followers. Additionally, data indicates that human accounts tend to have a larger following than bot accounts. The second graph depicts a comparison curve for an account's total number of following. Human accounts have a steeper curve than bot accounts, indicating that they have a greater number of following.

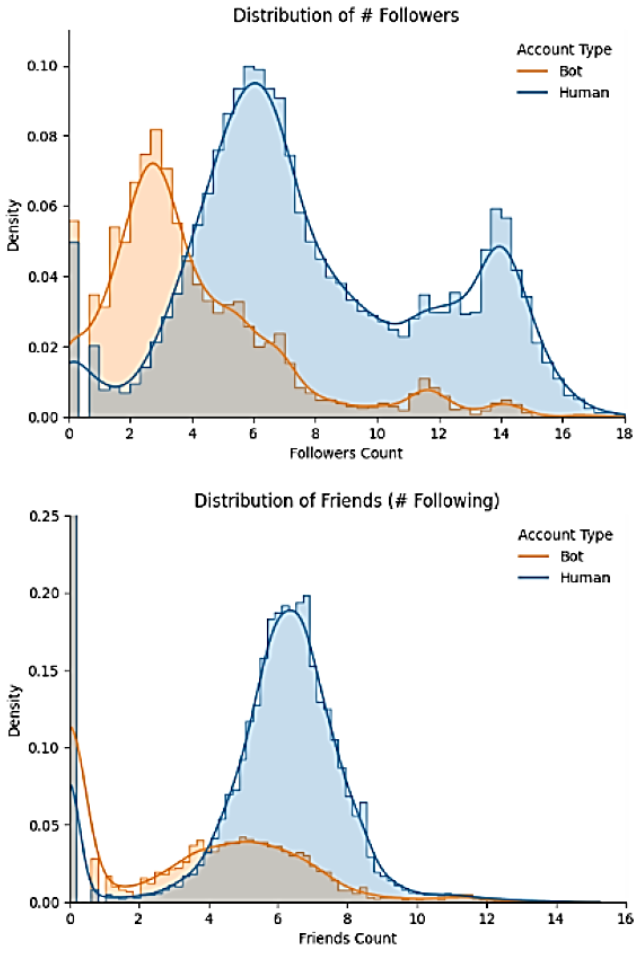

Fig. 3. Distribution of data

The next process is data pre-processing. At this stage, converting the data to a more usable form, converting the Boolean to binary 1/0, and converting the datetime to the account creation time stamp. Some of the pre-processing features include 'account_type', 'default_profile', 'default_profile_image', 'geo_enabled', 'verified', 'created_at'.

Based on the previous data exploration stage, it is known that the data used is not balanced (imbalanced data), so to overcome this, over-sampling is carried out using SMOTE. The benchmarking results obtained are shown in table 1.

TABLE I. NON SMOTE AND SMOTE RESULTS COMPARISON

NON-SMOTE			SMOTE		
Train Score			Train Score		
	Random Forest	XG Boost		Random Forest	XG Boost
Accuracy	0.87431	0.87302	Accuracy	0.88396	0.88786
Precision	0.85329	0.81151	Precision	0.88995	0.89542

Authorized licensed use limited to: Bursa Uludag Universitesi. Downloaded on January 30,2023 at 01:29:28 UTC from IEEE Xplore. Restrictions apply.

Recall	0.75025	0.80424	Recall	0.87603	0.87810
F1 Score	0.79843	0.80778	F1 Score	0.88293	0.88666
ROC AUC	0.93001	0.93417	ROC AUC	0.95375	0.95712
Test Score			**Test Score**		
	Random Forest	**XG Boost**		**Random Forest**	**XG Boost**
Accuracy	0.8784	0.8613	Accuracy	0.8808	0.8870
Precision	0.8530	0.8087	Precision	0.8868	0.8944
Recall	0.7473	0.7399	Recall	0.8738	0.8785
F1 Score	0.7966	0.7728	F1 Score	0.8803	0.8864

Table I shows that XGBoost and Random Forest yield relatively similar values. Before SMOTE was implemented, Random Forest was superior to XGBoost, but after SMOTE was implemented, XGBoost was superior to Random Forest. The use of SMOTE results in performance improvements such as increased accuracy. The following test is an assessment of feature importance to see what features distinguish the most between bot accounts and humans.

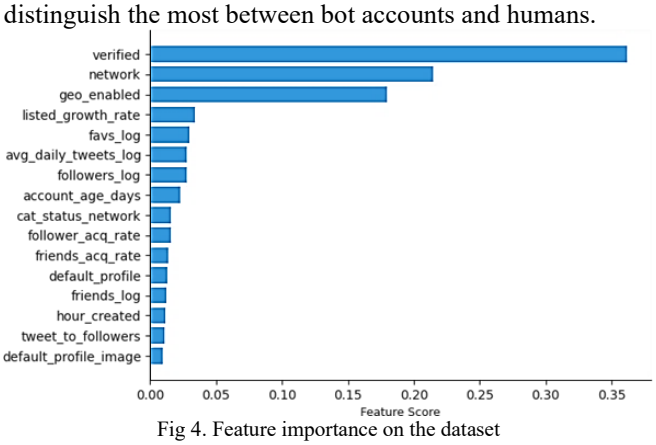

Fig 4. Feature importance on the dataset

Based on Fig. 4, it can be seen that the plots that have a level of importance in the XGBoost algorithm in the top 3 ranks are 'verified', 'network', 'geo_enabled'. Fig. 4 also shows that the feature score no longer shows a significant number apart from these three attributes. The lowest importance is default_profile_image, which means it does not differentiate much between bot or human accounts.

To optimize model performance, we perform model hyperparameter tuning. Hyperparameter tuning is the process of finding the hyperparameter value. This study uses the grid-search library to find the best hyperparameters to produce the model with the best performance. In Random Forest, the tuning hyperparameters are 'bootstrap', 'max_depth', 'max_features', 'min_samples_leaf', 'min_samples_split' and 'n_estimators', while in XGBoost, the hyperparameters that are tuned are 'max_depth', 'n_estimators', 'learning_rate'. The results obtained after performing hyperparameter tuning are described in table 2.

TABLE II. HYPER-PARAMETER TUNING RESULT

Train Score		
	Random Forest	**XG Boost**
Accuracy	0.87762	0.89230
Precision	0.88331	0.89832
Recall	0.86999	0.88456
F1 Score	0.87659	0.89137
ROC AUC	0.95093	0.95961

Test Score		
	Random Forest	**XG Boost**
Accuracy	0.8762	0.8908
Precision	0.8810	0.8942
Recall	0.8709	0.8873
F1 Score	0.8759	0.8907
ROC AUC	0.9621	0.9596

Table II is the final result of benchmarking using SMOTE with hyperparameter tuning. Based on Table 4. it can be seen that the values generated by the two algorithms have changed. XGBoost still produces superior scores than Random Forest (RF) with an accuracy score of 89.08% while the accuracy of RF is 87.62%. These results show the XGBoost algorithm has a significant advantage with 1.46%.

XGBoost is a viable solution for unbalanced data sets, random forests cannot be trusted in certain situations. In applications such as fraud detection, bot detection class is likely to be unbalanced, with a disproportionately large number of legitimate transactions compared to inauthentic transactions. The reason is that when XGBoost fails to predict anomalies the first time, it increases its preferences and weights in the next iteration, increasing its ability to predict low-participation classes.

V. CONCLUSION

The following conclusions are drawn from the tests conducted: The XGBoost method outperforms Random Forest (RF) and Support Vector Machine (SVM), with ultimate accuracy values of 0.8908 for the XGBoost algorithm, 0.8762 for Random Forest, and 0.8689 for SVM. The primary distinguishing elements between human and bot accounts are verified, network, and geo-enable. Where verified is a feature that allows to determine whether or not an account is verified. Human accounts are frequently validated. The term "network" refers to the overall number of followers and followers, whereas "geo-enable" refers to the user's geographic location. Human accounts, on average, have more followers and followers. Additionally, human accounts frequently include geographic information in their account details.

VI. SUGGESTION

There numerous recommendations for further research are made, including benchmarking deep learning algorithms to compare prediction accuracy. Also, the further research can employ a hybrid detection methodology by utilizing account information (account level) and submitted content (tweet level).

REFERENCES

[1] Statista, “Leading countries based on number of Twitter users as of July 2021,” [Online] available at: https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/ [access on 30 Agust 2021]

[2] O. Varol, E. Ferrara, C. A. Davis, F. Menczer and A. Flammini, “Online Human-Bot Interactions: Detection, Estimation, and Characterization,” Proceedings of the Eleventh International AAAI Conference on Web and social media (ICWSM), pp. 280 - 289, 2017.

[3] C. S. K. Aditya, M. Hani'ah, A. A. Fitrawan, A. Z. Arifin and D.

Authorized licensed use limited to: Bursa Uludag Universitesi. Downloaded on January 30,2023 at 01:29:28 UTC from IEEE Xplore. Restrictions apply.

Purwitasari, “Deteksi Bot Spammer pada Twitter Berbasis Sentiment Analysis dan Time Interval Entropy,” Jurnal Buana Informatika, pp. 179-186, 2016.

[4] Z. Chu, S. Gianvecchio, H. Wang and S. Jajodia, “Detecting Automation of Twitter Account: Are You a Human, Bot, or Cyborg?” IEEE Transactions on Dependable and Secure Computing, vol. 9, pp. 811-824, 2012.

[5] R. Battur and N. Yaligar, “Twitter Bot Detection Using Machine Learning Algorithms,” International Journal of Science and Research (IJSR), vol. 8, no. 7, pp. 304 - 307, 2018.

[6] C. A. Davis, O. Varol, E. Ferrara, A. Flammini and F. Menczer, “BotOrNot: A System to Evaluate Social Bots,” International Conference Companion on World Wide Web, pp. 273 - 274, 2016.

[7] A. Mahmood and P. Srinivasan, “Twitter Bots and Gender Detection Using TF- IDF,” Conference - CLEF 2019, 2019.

[8] S. Kudugunta and E. Ferrara, “Deep Neural Networks for Bot Detection,” Information Sciences, vol. 467, pp. 312-322, 2018.

[9] A. A. Amrullah, A. Tantoni, N. Hamdani, R. T. Bau, M. R. Ahsan and E. Utami, “Review Atas Analisis Sentimen pada Twitter Sebagai Representasi Opini Publik Terhadap Bakal Calon Pemimpin,” Seminar Nasional Multi Disiplin Ilmu Unisbank, 2016.

[10] P. G. Efthimion, S. Payne and N. Proferes, “Supervised Machine Learning Bot Detection Techniques to Identify Social Twitter Bots,” SMU Data Science Review, vol. 1, 2018.

[11] A. Agarwal, B. Xie, I. Vovsha, O. Rambow and R. Passonneau, “Sentiment Analysis of Twitter Data,” Department of Computer Science Columbia University, 2014.

[12] L. Zhang, R. Ghosh, M. Dekhil, M. Hsu and B. Liu, “Combining Lexicon-based and Learning-based Methods for Twitter Sentiment Analysis,” HPL-2011-89, vol. 89, 2011.

[13] A. Java, X. Song, T. Finin and B. Tseng, “Why We Twitter: Understanding Microblogging Usage and Communities,” Proc. Ninth WebKDD and First SNA- KDD Workshop Web Mining and Social Network Analysis, 2007.

[14] X. Liu, K. Tang, J. Hancock, J. Han, M. Song, R. Xu, V. Manikonda and B. Pokorny, “SocialCube: A Text Cube Framework for Analyzing Social Media Data,” International Conference on Social Informatics, pp. 252 - 259, 2012.

[15] L. Dey, I. Verma, A. Khurdiya and S. B. Handini, “A Framework to Integrate Unstructured and Structured Data for Enterprise Analytics,” Information Fusion (FUSION), p. 1988 – 1995, 2013.

[16] N. M. S. Hadna, P. I. Santosa and W. W. Winarno, “Studi Literatur Tentang Perbandingan Metode Untuk Proses Analisis Sentimen di Twitter,” Seminar Nasional Teknologi Informasi and Komunikasi (SENTIKA 2016), 2016.

[17] E. Stanton, O. Varol, C. Davis, F. Menczer and A. Flammini, “The Rise of Social Bots,” Communications of the ACM, vol. 59, 2016.

[18] H. Wei, J. Hurwitz and D. Kirsch, Machine Learning for dummies, Hoboken, USA: John Wiley & Sons, Inc., 2018.

[19] S. Kudugunta, T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” In Proceedings of the 22nd ACM international conference on knowledge discovery and data mining, pp. 785-794, 2018

Authorized licensed use limited to: Bursa Uludag Universitesi. Downloaded on January 30,2023 at 01:29:28 UTC from IEEE Xplore. Restrictions apply.