Unmasking Twitter Bots: Feature Engineering and Machine Learning for Bot Account Identification

Zeynep Babür†, Umut Bekdemir‡, Açelyanur Şen†, Şevval Özlem Çarkıt†, Oğuzhan Genç‡, Ayla Gülcü‡, Cihangir Gümüştaş*, Ece Gelal Soyak†
† Computer Engineering, Bahcesehir University, Istanbul, Turkey
‡ Software Engineering, Bahcesehir University, Istanbul, Turkey
$*$ Management Engineering, Bahcesehir University, Istanbul, Turkey

***Abstract*—Bot accounts on X (formerly known Twitter)[1] are a growing issue that limits and negatively impacts the browsing and sharing experience of Twitter users, and it is important to identify such accounts. In this work, we perform machine learning-based estimation of bot accounts on Twitter. Using Twitter's API, a data set is collected containing tweets and related metadata from various accounts. Feature engineering techniques are then applied to highlight relevant features such as sentiment analysis of the account's tweets, or the account's friend/follower ratio. Using these features to train and evaluate machine learning models, the likelihood of a given account being a bot is estimated. The performances of three different models are comparatively analyzed based on their f1 score, accuracy, precision, and recall. Analysis of feature importances shows the success of derived features in identifying bot accounts. This work demonstrates the potential of using feature engineering with tweet and profile properties to detect bot accounts on Twitter, and provides a foundation for further research on this topic.**

***Index Terms*—machine learning, social networks, bot detection, feature engineering**

I. Introduction

Twitter is a microblogging social network based on the sharing of tweets, which are short textual content. Twitter also allows its users to interact with each other by replying to tweets, quoting other users' tweets or retweeting other users' messages [1]. This social platform provides access to its services for all registered users through a web page, a mobile application, or an application programming interface (API). On the one hand, this form of communication facilitates the development of an environment that increases the user's involvement when using and combining materials. On the other hand, it also enables the development of tools that control accounts and mechanize tweeting [2]. Fully automated accounts, or bots, can be used to retweet interesting and relevant content for groups or to compile tweets about a subject.

Intentional bot account directing, targeting and attacks negatively affect the overall user experience of Twitter users. In addition, some Twitter bots use the platform's features for harmful purposes, including election meddling and blatant propaganda [3]. Twitter bots coexist with real people while disguising their automated origins by impersonating real people [4]. Many academic efforts have been committed to the compilation of pertinent data sets because recognizing bots in social media is essential to protecting the integrity of online conversation.

Although there are several public tweet data sets available [5] [6], these data sets mostly do not comply with the present data status or do not provide the level of detail necessary for a deep analysis. In this work, we create a new data set with comprehensive attributes pertaining to user accounts. In addition, we use sentiment analysis in Turkish language and feature engineering to identify a set of properties that assist in automatically identifying bot accounts. After employing a novel semi-automatic labelling strategy, we train four classical machine learning models, namely radial basis function (RBF), support vector machine (SVM), naive Bayes (NB), and random forest (RF), and compare the bot account prediction performances on the data set.

Our contributions can be summarized as follows:

- We construct a novel data set comprising novel features obtained by a statistical analysis on user and tweet data and by sentiment analysis (in Turkish language).
- We train supervised machine learning algorithms on the constructed data set (and on its small subsets) and evaluate bot detection performances.
- Our results show that the random forest classifier achieves at least 95% accuracy and precision as high as 99% regardless of the data set size, demonstrating the success of our feature engineering and data labeling algorithms.
- Feature importances reveal the success of derived features; the top three features impact the algorithm with 49%, 10% and 8% weights, respectively.

II. Related Work

Social media networks like Facebook and Instagram are highly popular platforms; inevitably, people or groups attempt to exploit these platforms to publish and spread pertinent content via bot accounts. Numerous works on bot detection have been proposed to date. Online social networks have been practiced in this manner using some basic public features like follower count, following count, or the count of

[1]In October 2022, Twitter was acquired, after which Twitter Inc. ceased to exist as an independent company and was merged with X Corp. To prevent possible inconsistencies and confusion in literature search results, we used the name Twitter throughout the text.

positive and negative tweets [7] or frequency of comments and posts [8]. There has been several approaches towards detecting bots including graph-based, Machine Learning (ML)-based, crowdsourcing-based, and anomaly-based techniques [9].

Among the most recent works that studied identifying Twitter bots using ML-based techniques, [10] studied the problem as a multi-class bot detection, identifying different types of bots and their relationship with different features, validating on open data sets from Botometer. The work also analyzed the importance and the types of account features, however, did not use feature engineering. With the aim of reducing skewed sampling and inaccuracies with linguistic data, a multi-language model for bot detection has been proposed in [11]. The analysis relies on the density of tweet hashtags, mentions and URLs, but does not perform sentiment analysis to analyze tweet content. A new dataset was created using the graph-based friend network analysis approach in [12], to be used as a benchmark in bot detection studies. This approach is complex and not practical for every device, whereas utilizing up-to-date information holds a significant importance towards bot detection, as bot accounts might show relatively short life span compared to non-bot accounts. Most recently, a combination of user metadata attributes, features generated using NLP, graph analytics and social network representation of users was investigated in order to distinguish between genuine and spam accounts in Twitter [13]. The proposed method was tested using several supervised machine learning models, with the tree-based models performing best. Although there are approaches that focuses deep learning practices, they did not utilize tweet content to gain insight [4] or were validated on limited data sets [14].

III. Constructing the Data Set

In this section, we explain our system and methodology for collecting the data set for bot account analysis.

Our system comprises three main modules. The initial module is *data gathering*, where Twitter API is utilized to collect the tweets. Next, the data undergoes a parsing procedure wherein bot accounts are identified and *data labeling* is performed; in this phase, a csv file is created as the data set. The final module involves *training of ML models* on this data set, for identifying bot accounts. Figure 1 illustrates all modules.

A. Scraping Tweets

Twitter API is the main tool for automatically collecting tweets. While there are a number of Python libraries available for connecting to the Twitter API, we set out to use the two most commonly used tools, Tweepy and Snscrape, to collect our dataset. Tweepy [15] is a closed-source tool officially endorsed by Twitter, and Snscrape [16] is an open-source tool enabling data acquisition from various social media services. It must be noted that the change of Twitter data regulations imposed challenges in the data collection phase. During the process of our data collection, several modifications were implemented to the policies around Twitter

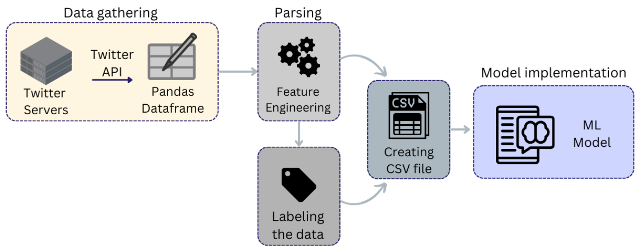

Fig. 1. Stages in bot detection process

API, and consequently to the Tweepy library. These changes included the introduction of a steep pricing model for API access for academic research; consequently, we switched to primarily using Snscrape library to connect to Twitter API. Snscrape utilizes Twitter's search functionality to extract the desired data by employing hashtags or non-hashtag words. Albeit working slightly slower, we found that Snscrape is able to scrape more and different features regarding tweets. Additionally, during the data scraping process, a specific time interval can be configured to make the collected data more specific. We have collected data over a five-month interval starting with January 1, 2023.

As part of our data collection strategy, we focused on tweets with popular hashtags. As popular hashtags are listed in Twitter's "Trending" section, some of these popular hashtags were used to manually search for relevant tweets. In addition, some hashtags lead to bot accounts with a greater possibility; as an example, hashtags with political phrases are likely to be found in the tweets from bot accounts (particularly because of the presidential elections in Turkey). In our work, the hashtags were manually selected in order to cherry-pick the tags that might potentially yield bot accounts.

For the selected hashtags, the corresponding tweets were collected along with their meta-data. For each tweet, user-related data such as the username, location, friends count and profile image have been extracted. In addition, the meta-data such as the posting time, text content, likes' count and user ID are collected. The collected fields for user-related and tweet-related data are presented in Table I and Table II, respectively.

TABLE I
User-Related Features

username	verified	favourites_count
screen_name	profile_url	friends_count
user_id	description	profile_image
user_location	link_url	listed_count
followers_count	statuses_count	created_at

B. Feature Engineering and Labeling the Data

Once the initial set of user and tweet related features is collected, we used Python data manipulation library, Pandas,

TABLE II
TWEET-RELATED FETURES

time	user_id	tweet
tweet_id	retweeted	reply_count
like_count	tweet_url	hashtags
mentions	photo, video	cleaned_tweet

to analyze the data and derive new features that would potentially improve the bot account detection capability.

Previous work showcased that bot accounts tweet more compared to normal accounts [17], [18]. Thus, we generated a new feature, named *tweet_times*, which investigates tweeting times and has a value of 1 if the tweet density is above a threshold. We also introduced the *tweet_content* feature, which represents tweets' similarity. This feature is motivated by the study in [19], which states that "seemingly unrelated bot accounts usually tweet nearly identical content at nearly the same time".

In addition, we performed sentiment analysis on the latest 40 tweets of each user. The analysis weighs the emotional tendencies into three categories, namely positive, negative and neutral. Our aim was to identify whether the account's positive or negative tendencies are dominant, as bots tend to either promote (i.e., positive attitude) or demote (i.e., negative attitude) what they comment on. In addition to sentiment analysis, we also took into account lower-impact features such as having numbers at the beginning or end of the username, the account creation date being close to the present date, or the account not having location information or a description. This analysis yields *sentiment_analysis*, *user_ends_numeric*, *user_begins_numeric*, *date_historic*, *user_lacks_location* and *description* features.

Finally, we examined the accounts' followers/friends ratio. The bounds for this ratio have been discussed in [20]; in our work, we grouped all accounts based on their follower count and checked whether their follower/friends ratio exceeded the thresholds. Table III provides the descriptions of these derived features.

TABLE III
DESCRIPTION OF DERIVED BOOLEAN FEATURES

Feature	**Definition**
tweet_times	Is tweeting times above a threshold?
tweet_content	Is tweet content repetitive?
sentiment_analysis	Does the user clearly promote or demote something in their tweets?
user_begins_numeric	Does the user's handle begin with a number?
user_ends_numeric	Does the user's handle end with 5 consecutive numbers?
date_historic	Was the account created less than a year ago?
user_lacks_location	Does the user lack the location information in the profile?
description	Does the user's profile lack a description?
followers_friends	Is the user's followers/friends ratio outside of the average boundary values?

We use the derived features in Table III, along with the *verified* feature from Table I (indicating whether the account has the blue, yellow or grey tick next to account name), in our algorithm for flagging the bot accounts and thus labeling the data. Table IV shows the number of accounts that satisfy each such feature and the corresponding percentage of such accounts in the data set, where the features marked with the (*) symbol represent the derived features. These statistics reveal that while some attributes are rarely found in all accounts, they are prominent and common among potential bot accounts. Hence, we place more weight on such attributes in our bot computation algorithm than some other attributes that are less important and more frequent. As it is not possible to certainly state whether an account is a bot, there has been several iterations of the calculation, where we had to rely heavily on trial and error. Specifically, the accounts that were deemed as potential bots were manually examined and our bot calculation formula was consequently revised for better accuracy. With the latest calculation formula, we estimate that 13,681 of the 307,713 accounts in our data set are potential bots; this number corresponds to 4.45% of the data set.

TABLE IV
DISTRIBUTION OF FEATURES USED FOR LABELING IN THE DATA SET

Feature	**Number**	**Percentage**
*tweet_times	527	0.17%
*tweet_content	12697	4.13%
*sentiment_analysis	56786	18.5%
*user_lacks_location	169986	55.2%
*user_ends_numeric	28189	9.16%
*user_begins_numeric	6076	1.97%
*followers_friends	38234	12.4%
*date_historic	80976	26.3%
*description	96880	31.5%
verified	1741	0.57%

C. Final Data Set

Python Pandas library is used to store the collected user information in DataFrames to make it easier to read the data and apply tabular processes. Gathered and labelled data is preprocessed, so that ML models can operate easily on the data set. Parameters that are too complex are dropped and some of the features (e.g. from Table I) are merged to gain additional insight. As an example, the average number of tweets that a user sends was stated in the literature as 2.6 [21], but an observation on our data set revealed that *statuses_count* value was mostly extremely low or extremely high in accounts that had higher percentage of being a bot. As a result, a new feature, *statuses_count_per_day* is created to inspect this issue. The relationship between the *friends_count* and *followers_count* features and the account's age (*created_at*) is investigated, and *followers_friends_ratio*, *historic_friend* and *historic_follower* features are introduced.

In addition, some of the discrete-valued features from Table III such as *user_ends_numeric*, *user_begins_numeric*, *description* and *date_historic* have been further analyzed to reveal further information for ML. Towards this, the former two features have been merged under *name_numeric* feature, to indicate the count of numbers in the account name. The scope of the *description* feature is altered, to hold the floating point value representing the sum of the positive and negative weights of the sentiment analysis of the tweet content. Finally, *date_historic* is altered into a timestamp to hold the account's creation date. The updated set of features and the corresponding formulations are listed in Table V.

Table VI shows the list of all 20 features in our final data set.

TABLE V
NEW/UPDATED FEATURES BEFORE MODEL TRAINING

Feature	**Definition**
statuses_count_per_day	(status_count / age of account)
followers_friends_ratio	followers_count / friends_count
historic_friend	friend_count / age of account
historic_follower	followers_count / age of account
date	Timestamp of the account creation date
name_numeric	Numeric values in username
description	Sentiment analysis (sum of positive and negative weights)

TABLE VI
THE FINAL SET OF FEATURES IN OUR DATA SET

favourites_count	friends_count	followers_count
profile_image	date_historic	listed_count
historic_friend	historic_follower	followers_friends
user_lacks_location	link_url	verified
tweet_times	tweet_content	sentiment_analysis
description		statuses_count
statuses_count_per_day		name_numeric
followers_friends_ratio		

IV. EXPERIMENTS

We evaluate the bot detection performances using three machine learning models, namely Support Vector Machine (SVM) with Radial Basis Function (RBF) kernel, Bernoulli Naive Bayes (NB) and Random Forest (RF).

A. Mini Data Sets

First, we construct two smaller data sets by randomly selecting a subset of the large data set, in order to showcase how the performance scales in large as well as small data sets. The first mini data set (Data set-1) consists of 5,000 bots and 5,000 normal accounts. The second mini data set (Data set-2) contains the same number of accounts, but 3,000 of them are bots and 7,000 are normal accounts. Both data sets consist of the 20 features listed in Table VI. In both data sets, we used a 70%-30% train-test split.

On the mini data sets, in addition to the training and testing partitions, an additional validation set consisting of 200 accounts (100 bots and 100 normal accounts) was created from accounts *not* included in the two mini data sets. The purpose of the validation set was to observe whether the model overfits or deviates when tested against new data.

In our experiments, Support Vector Machine (SVM) model was used with the Radial Basis Function (RBF) Kernel, as this kernel is known to perform better with non-linear data sets. Before training the Support Vector Machine (SVM) model, as our data set includes a variety of features with different scales of values, and due to SVM's sensitivity to the range of features in a data set, feature scaling is applied.

When training the Random Forest (RF) model on the imbalanced data set, random forest's *class weight* feature is also used, so that the model split accounts for the lower number of bot accounts. Grid search and Bayesian search are used to tune the hyperparameters of the RF model; both approaches yield similar results. After parameter tuning, the following configuration is used in our tests: *max_depth*=5, *min_samples_split*=5, *n_estimators*=300, *min_samples_leaf*=1.

Regarding the Naïve Bayes (NB) classifier, we experimented with all three types of models, namely Gaussian, Bernoulli and multinomial, and proceeded with the Bernoulli Naïve Bayes (NB) model, as it yielded the better results, and is known for providing high performance with small training data [22]. Hyperparameter tuning with this model did not change the parameters.

Results of the models using the mini data sets are shown in Table VII. We observe that all models provide higher accuracy but with lower precision when operating on the data set having 30% bots. Analysis of the confusion matrix showed that the models flag a greater number of false positives when trained on this data set. SVM with RBF yields the highest precision, on the balanced mini data set, and RF gives the highest accuracy, on the imbalanced mini data set (*class_weight* applied).

TABLE VII
MODEL PERFORMANCES ON THE MINI DATA SETS

Data Set	**Model**	**Accuracy**	**Precision**	**Recall**	**F1**
Data set-1	RF	92%	0.946	0.889	0.915
Data set-1	**RBF_SVM**	**92%**	**0.95**	**0.88**	**0.92**
Data set-1	NB	87.8%	0.946	0.808	0.872
Data set-2	**RF**	**95.3%**	**0.902**	**0.949**	**0.925**
Data set-2	RBF_SVM	92%	0.94	0.80	0.86
Data set-2	NB	92%	0.899	0.826	0.861

B. Large Data Sets

We repeat our measurements on large data sets. Again, we construct two data sets with different bot-normal account ratios. Data set-3 has 50% bots and 50% normal accounts, 13,581 each in number, in total comprising 27,162 rows. Data set-4 has 30%-70% ratio of bot and normal accounts, respectively; this data set contains 13,581 bot accounts and 31,689 normal accounts, summing up to 45,270 rows. We train the same ML models; results are presented in Table VIII.

TABLE VIII
MODEL PERFORMANCES ON THE LARGE DATA SETS

Data Set	Model	Accuracy	Precision	Recall	F1
Data set-3	**RF**	**96.9%**	**0.949**	**0.993**	**0.97**
Data set-3	RBF_SVM	89%	0.86	0.94	0.90
Data set-3	NB	82.9%	0.920	0.722	0.809
Data set-4	**RF**	**98%**	**0.971**	**0.979**	**0.975**
Data set-4	RBF_SVM	92%	0.92	0.88	0.87
Data set-4	NB	89%	0.895	0.733	0.806

We observe that the Support Vector Machine model is adversely affected by the growth and imbalance of the data set, making the model more difficult to learn. Feature importances with SVM demonstrate that the most impactful features are *tweet_content*, *sentiment_analysis*, *tweet_times*, *description* and *name_numeric*; and the remaining features have a weight of less than 0.14.

With the Naive Bayes model, we observe that the performance worsens on the larger data set. In addition, model precision remains the same for NB regardless of the percentage of bot accounts in the large data set. We interpret these to mean that the model does not gain insight when the parameters are evaluated independently. *followers_friends_ratio, followers_count*, and *favourites_count* are the most important features according to our experiments with NB.

The Random Forest model exhibits better performance on the larger data set (45,270 accounts). The optimized hyperparameter values are *max_depth*=10, *min_samples_split*=5, *n_estimators*=200, *min_samples_leaf*=1. The model performs better than the other models in both percentages of bot accounts in the data set. The accuracy that can be attained is as high as 98% on the data set having 30% bot accounts.

Figure 2 shows the learning curve and ROC curve of the best-performing RF model on Data set-4. Figure 3 demonstrates the confusion matrix on this model, for both training and testing sets. We observe only ∼99% of true positives, demonstrating success in bot detection, with false positives (mispredicted normal accounts) as low as ∼2%.

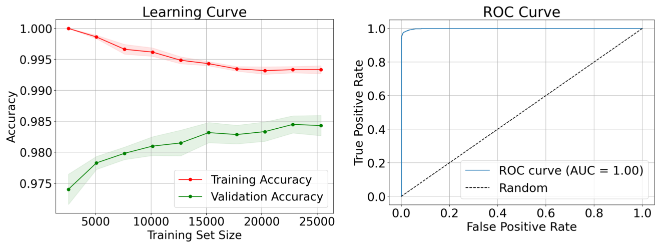

Fig. 2. Learning Curve and ROC Curve of Random Forest Model

We present the feature importances for Random Forest on Data set-4 in Figure 4. We observe that *tweet_content* is the most important feature, followed by *description* formed upon sentiment analysis and *date* indicating how recently the account was created. 7 out of 20 features have less than 1% importance individually. Our deeper analysis on bot and normal accounts revealed that the value of *historic_follower* is much higher in bot accounts compared to normal users, implying bot accounts have high numbers of followers per the time that elapsed since account creation.

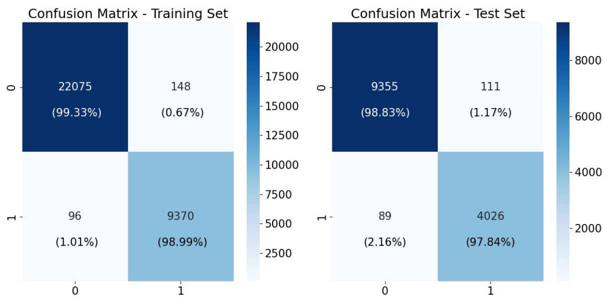

Fig. 3. Confusion Matrix of Random Forest Model

V. CONCLUSIONS AND FUTURE WORK

In this work, we utilized feature engineering in order to improve the accuracy in labeling and accurately detecting Twitter bot accounts. We constructed a data set by scraping a comprehensive set of attributes regarding Twitter accounts, and upon analysing certain statistics on this data set and utilizing a semi-automated strategy, we have labeled bot accounts. We then trained and evaluated classical machine learning models on this data set and assessed their bot detection performance.

In social networks, the risk of an individual reading biased content is that this content may easily be disseminated to others. Therefore, our findings may have practical implications for brands, politicians, and societal decision makers.

ACKNOWLEDGEMENT

The authors would like to thank Bilge and ClinicWall for their assistance and insightful discussions throughout the duration of this study.

REFERENCES

[1] J. Rodríguez-Ruiz, J. I. Mata-Sánchez, R. Monroy, O. Loyola-González, and A. López-Cuevas, "A one-class classification approach for bot detection on Twitter," *Computers & Security*, vol. 91, p. 101715, 2020.
[2] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia, "Who is tweeting on Twitter: human, bot, or cyborg?" in *Proceedings of the 26th Annual Computer Security Applications Conference*, 2010, pp. 21–30.
[3] A. Deb, L. Luceri, A. Badaway, and E. Ferrara, "Perils and challenges of social media and election manipulation analysis: The 2018 US Midterms," in *Companion Proceedings of the 2019 World Wide Web Conference*, 2019, pp. 237–247.
[4] S. Kudugunta and E. Ferrara, "Deep neural networks for bot detection," *Information Sciences*, vol. 467, pp. 312–322, 2018.
[5] K.-C. Yang, O. Varol, P.-M. Hui, and F. Menczer, "Scalable and generalizable social bot detection through data selection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, pp. 1096–1103.
[6] M. Mazza, S. Cresci, M. Avvenuti, W. Quattrociocchi, and M. Tesconi, "RTbust: Exploiting Temporal Patterns for Botnet Detection on Twitter," in *Proceedings of the 10th ACM Conference on Web Science*, 2019, pp. 183–192.

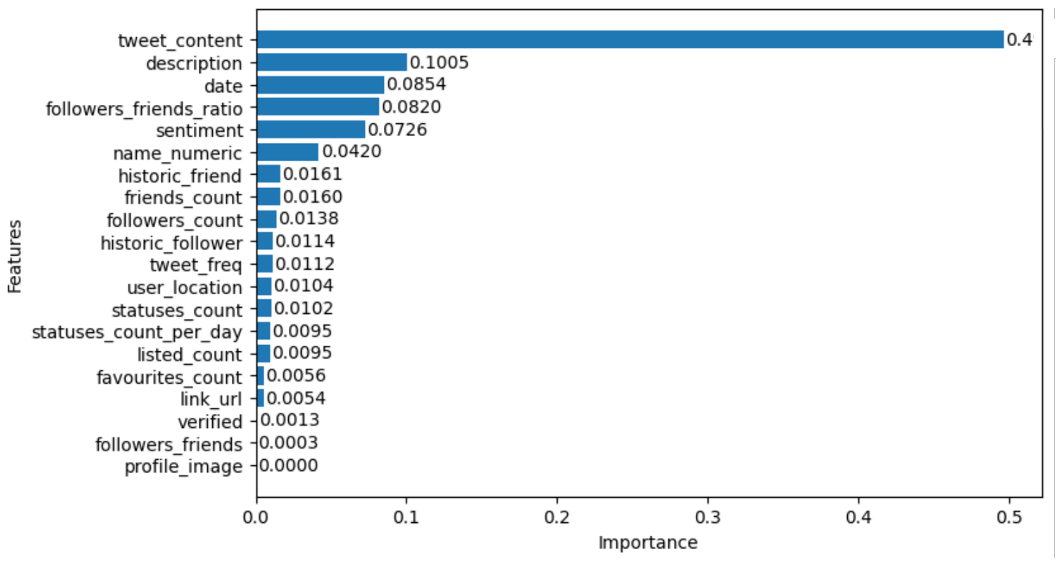

Fig. 4. Random Forest Feature Importances

[7] M. Heidari, H. James Jr, and O. Uzuner, "An empirical study of machine learning algorithms for social media bot detection," in *2021 IEEE International IoT, Electronics and Mechatronics Conference (IEMTRONICS)*, 2021, pp. 1–5.

[8] G. C. Santia, M. I. Mujib, and J. R. Williams, "Detecting social bots on Facebook in an information veracity context," in *Proceedings of the International AAAI Conference on Web and Social Media*, 2019, pp. 463–472.

[9] M. Orabi, D. Mouheb, Z. Al Aghbari, and I. Kamel, "Detection of bots in social media: A systematic review," *Information Processing Management*, vol. 57, no. 4, p. 102250, 2020.

[10] I. Dimitriadis, K. Georgiou, and A. Vakali, "Social botomics: A systematic ensemble ml approach for explainable and multi-class bot detection," *Applied Sciences*, vol. 11, no. 21, p. 9857, 2021.

[11] J. Lundberg, J. Nordqvist, and M. Laitinen, "Towards a language independent Twitter bot detector," in *DHN*, 2019, pp. 308–319.

[12] S. Feng, H. Wan, N. Wang, J. Li, and M. Luo, "Twibot-20: A comprehensive Twitter bot detection benchmark," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 4485–4494.

[13] M. Chakraborty, S. Das, and R. Mamidi, "Detection of fake users in Twitter using network representation and NLP," in *Proceedings of 14th International Conference on COMmunication Systems NETworkS (COMSNETS)*, 2022, pp. 754–758.

[14] C. Monica and N. Nagarathna, "Detection of fake tweets using sentiment analysis," *SN Computer Science*, vol. 1, pp. 1–7, 2020.

[15] [Online]. Available: https://www.tweepy.org/

[16] JustAnotherArchivist, "Justanotherarchivist/snscrape: A social networking service scraper in python." [Online]. Available: https://github.com/JustAnotherArchivist/snscrape

[17] X. Yuan, R. J. Schuchard, and A. T. Crooks, "Examining emergent communities and social bots within the polarized online vaccination debate in Twitter," *Social media+ society*, vol. 5, no. 3, pp. 1–12, 2019.

[18] C.-F. Chen, W. Shi, J. Yang, and H.-H. Fu, "Social bots' role in climate change discussion on Twitter: Measuring standpoints, topics, and interaction strategies," *Advances in Climate Change Research*, vol. 12, no. 6, pp. 913–923, 2021.

[19] A. Shevtsov, M. Oikonomidou, D. Antonakaki, P. Pratikakis, A. Kanterakis, P. Fragopoulou, and S. Ioannidis, "Discovery and classification of twitter bots," *SN Computer Science*, vol. 3, no. 3, apr 2022.

[20] Z. Gilani, R. Farahbakhsh, G. Tyson, and J. Crowcroft, "A large-scale behavioural analysis of bots and humans on twitter," *ACM Transactions on the Web (TWEB)*, vol. 13, no. 1, pp. 1–23, 2019.

[21] M. Yaqub, "How many tweets per day 2022 (New Data)," Apr 2023. [Online]. Available: https://www.businessdit.com/number-of-tweets-per-day/: :text=Average%20Number%20of%20Tweets%20per

[22] P. Gamallo and S. Almatarneh, "Naive-bayesian classification for bot detection in Twitter." in *CLEF (Working Notes)*, 2019.