Topic: Twitter Bot Account Detection using account classification

Introduction

Motivation:

With the advent of online revolution, different social media platforms have become important part of our day-to-day activities. People have adopted virtual reality as crucial part of their life across all demographics and geographies. Hence, information on social media plays great part in shaping our thinking. There is so much information and personal thoughts floating all over social media. As per current statistics, there are around 330 million twitter and 2.7 billion Facebook users worldwide [1]. Out of these users there is a vast pool of "fake or bot accounts". These fake profiles are created to troll online, spread misinformation, change the voter's perspective, etc. Twitter provides flexibility to the developers to connect using API and create bot accounts. Objective of this feature was to provide more flexibility to the individuals and enhance their twitter usage. Initially their purpose was to help various businesses in reaching their costumers automatically and quickly. For e.g., Netflix bot tweets whenever new movie/series is available on the platform [2]. However, twitter API has been heavily misused. With advancement in technology various bot accounts are created on twitter very easily and quickly with malign intentions. Main purpose of these accounts is to mislead the society by spreading false news and creating trends to draw attention to harmful subjects [3]. Hence, chose this topic as differentiating these bots from humans is an essential task and requires detailed analysis.

In this project I intend to study twitter accounts data to identify features distinguishing bots from humans. Focus is to develop a model capable of reading account data like profile photo present or not, total number of followers, status updates, etc. to detect malicious activity. Additionally, we should not confuse bot accounts with fake human profiles. Both have same intentions but have different activities and hence in this project I focus only on identifying bot accounts.

Data

This dataset contains total 19 features [4]. Target feature is 0 or 1 for not bot or bot account, respectively. This dataset has total 2797 users with labels. Out of which 1476 rows are labeled as not bot account and 1321 are labeled as bot account. Below are the details for each feature:

Table 1 - Dataset Feature Details

S.No.	Column Name	Description	Data Type
1	id	Twitter Unique identifier for the account	Integer
2	screen_name	Name that appears on twitter screen of the user	Varchar
3	location	Place of this account	Varchar
4	description	Title or description added by user in their profile	Varchar
5	url	Twitter profile link for this user	Link
6	followers_count	Total Number of profiles following this user	Integer
7	friends_count	Total number of profiles this user is following	Integer
8	listed_count	The number of public lists is account is member of.	Integer
9	created_at	Time at with this profile was created	Datetime
10	favourites_count	Number of tweets liked by this user.	Integer

11	verified	Indicator if this account is a verified account or not.	Boolean
12	statuses_count	Total number of status posted by this user so far	Integer
13	lang	Language used by this user. Field holds the language code. For e.g., en for English, etc.	Varchar
14	status	Status posted by this user on the profile	JSON object
15	default_profile	If this user has default profile. No changes made by user to default twitter settings	Boolean
16	default_profile_image	If this user has default twitter profile picture.	Boolean
17	has_extended_profile	If this user has extended profile.	Boolean
18	name	Name used by user to register for twitter	Varchar
19	bot	Target feature indicating if this user is bot account or not	Integer

Problem Understanding:

This is a supervised learning problem as we have labeled dataset. Since we need to predict weather, the given account is a bot account or not, this is a binary classification problem and needs classification methods for identifying bot accounts. In this project four classification models will be used to label an account. Models that will be used are Random Forest, Logistic Regression and Naïve Bayes Classifier and Support Vector Machine. Based on accuracy best model will be selected.

Literature Review

Bot detection is a highly researched topic using both user level data and content posted by the users. Various machine learning and deep learning models are suggested in different papers for bot detection. We will focus on machine learning (ML) methods in this project utilizing account level features. In [5], they developed complex ML algorithm by using features like username length, followers to friend ratio with just 2.25% misclassification rate. Their dataset consisted of three groups each of traditional and social spam bots. For separated their bot detection into 3 analysis- profile, account activity and tweet or text mining. Account with unusual amount of activity like posting within seconds which is not possible by human user, hence were considered in identifying bots. They finally prepared a set of 16 binary input features and used Support Vector Machine (SVM) to perform classification. However, they applied logistic regression before SVM as LR is efficient way to prepare binary data and identify feature weights for SVM. They got overall accuracy of 97.75% using this method on all the datasets.

In another paper [6], they implemented four classification algorithms namely decision tree, Multinomial Naïve bayes, Random Forest and Bag of Words. Their data consisted of account specific features like followers and friends count, location, description, status, etc. They used Spearman Correlation method for extracting useful input features from dataset. All the 4 algorithms were trained using following features selected in feature selection phase: number of followers, friends, screen name, description, location and verified. Bag of Words algorithms worked best on this dataset with highest accuracy of 95%.

In another paper [7], decision tree, random forest and multinomial Naïve Bayes algorithms are used for detecting twitter bots. Their dataset consisted of 24 features specific to each user

collected using Twitter API ‘Tweepy’. Features which were identified and selected were screen name, location, verified and description. These features were fed to 3 classifiers. Among all the models, Decision Tree gave the best accuracy of 93%.

Feature Selection

Initial Feature Selection

As discussed in previous sections our data had total 19 features including target column. But the dataset had mixed datatypes and many columns were string or textual columns. We can not input these columns directly to models; hence we performed some data cleaning and feature engineering before applying feature selection methods for classification. We started by checking missing values and found 5 columns had nulls. These columns were location, description, URL, status and has_extended_profile. Out of these columns’ location and url had highest missing values around 40%, and additionally these columns are text, and we cannot convert to categorical or continuous features, hence we dropped these columns. This was the first initial step based on data quality for feature selection. For rest we performed some feature manipulation and added new columns in dataset. We then normalized numerical columns like friends count, followers count, listed count, favorite count and status count. This was followed by encoding Boolean columns to integer. Verified, default profile, default profile image and has extended profile were converted to integer type. 0 when value is False or Null and 1 when True. This way we were able to handle missing values in “has extended profile” column. Next, we cleaned language column. It contained language codes for profile. We changed this column to categorical column with 21 different language codes.

In the dataset we had many text-based columns like status, description, name, and screen name. We know we cannot use them directly and can not convert them to categorical data. Hence, we used some natural language processing on these columns to get two new columns containing polarity and subjectivity of each column. These columns are derived from sentiment package of TextBlob library. New columns were in the range of -1 and 1 hence did not need further scaling. Finally, after our initial analysis and processing of dataset we dropped columns which are either not required or can not be fed to models. These columns include id and id_str, location, screen name, url, description, created_at, lang, status and name. We dropped id columns as we know it is just unique identifier for each row and will overfit training set and hence models will not be able to generalize on new data. For remaining text columns, we have already added new polarity and subjectivity columns. Also, we have dropped created_at column, it holds the date at which account was created and is not in format acceptable to models.

After initial selection we had 19 input features including new columns in our dataset. Our dataset had balanced target column.

Feature Selection Methods

We used below four feature selection methods in our project and compared results for each. Results will be explained in next section. We used Scikit-learn feature selection method to apply all the below methods.

1. Chi Squared: This method works with non numerical and categorical features. Since polarity and subjectivity can be negative, we scaled our dataset and passed only training data to the selector. This is a filter-based method, and we calculate chi-square metric between the target and variables and only select k best variables with maximum chi-squared values.
2. Recursive Feature Elimination: This is wrapper-based method. The goal of recursive feature elimination (RFE) is to select features by recursively considering smaller and smaller sets of features. We passed number of features to be selected as 10 and used Logistic Regression object as estimator. It calculates coefficient of each feature and based on number of features we want to select it prunes least important features. It performs the task iteratively until we get best results.
3. L1 based selection: This is an embedded method. Embedded methods use algorithms that have built in feature selection capabilities. We used l1 penalty which forces a lot of feature weight to be zero and here we used "SelectFromModel" method of scikit-learn for code implementation. We used SVC linear model with C as 0.01. Lower we go with C value very few features are selected. We used C as 0.01 and we got 5 features selected. For classification we can use either Logistic Regression or SVC and for regression lasso model can be used as meta estimator for feature selection.
4. Random Forest Selection: This is again an embedded tree-based method. We again use SelectFromModel meta-transformer to discard less important features using impurity calculated by random forest selector. This method gave us 6 features.

Result

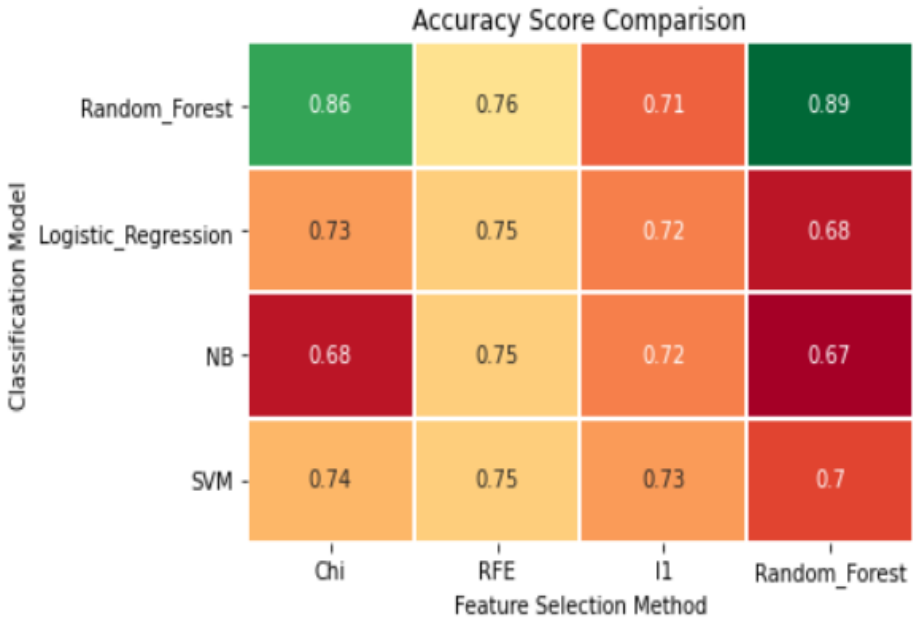
Classification Method Comparison

We have used four classification models in the project along with different feature selection methods. We created various graphs and metrics reports for all the methods. Below table shows the AUC scores for each of the methods along with all 4 feature selection methods.

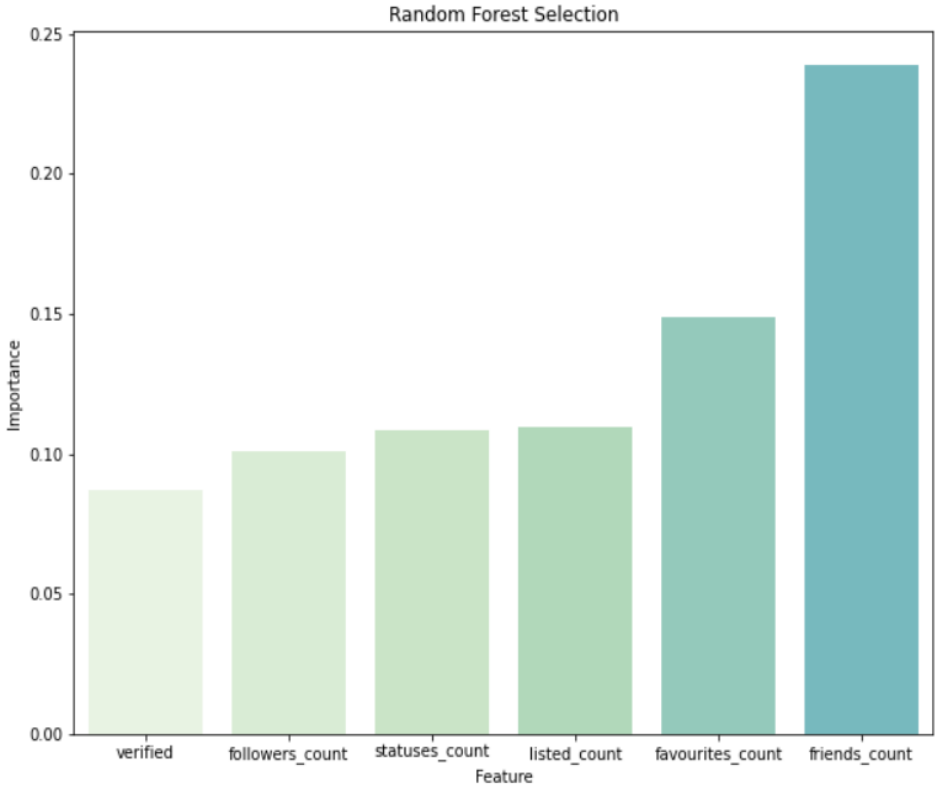
Table 2: AUC Comparison

	Random Forest	Logistic Regression	Naïve Bayes	SVM
Chi-Squared	0.94	0.81	0.81	0.84
RFE	0.84	0.84	0.84	0.84
L1 Based	0.78	0.80	0.80	0.79
Random Forest Selection	0.95	0.83	0.74	0.86

Accuracy comparison for different methods:

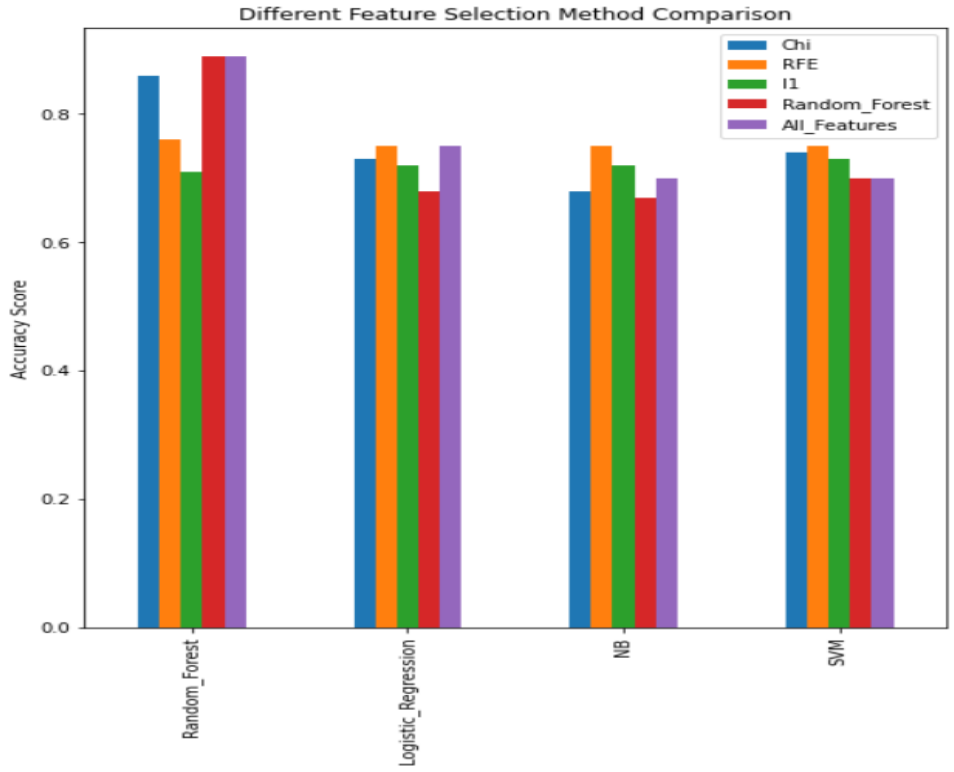


Detailed ROC plots, cumulative gain charts and lift charts along with confusion matrix for each method are present in the Jupyter notebook. Here we present the concise results for comparison. We have highlighted the highest value in the above table. We found that we got best results with Random Forest model using its in built feature selection method. It selects below 6 best features.



Different feature selectors selected different variables. We have discussed these in detail in next sections. Also to note that feature selection should be performed after training data split, as running on entire dataset results in data leakage issue and overfitting.

Selected vs All Feature Comparison



Bar charts for accuracy scores for different methods when run with selected and using all the features. We can see that Random Forest Selection and classification works best for our dataset.

Table 3: AUC Comparison with All Features

	Random Forest	**Logistic Regression**	**Naïve Bayes**	**SVM**
Chi-Squared	0.94	0.81	0.81	0.84
RFE	0.84	0.84	0.84	0.84
L1 Based	0.78	0.80	0.80	0.79
Random Forest Selection	0.95	0.83	0.74	0.86
All Features	**0.96**	**0.84**	**0.83**	**0.83**

Both the above graphs display the comparison of each feature selection method when all the features are selected. In some cases, we see accuracy have reduced after applying feature selection however AUC score has gone up or at least comparable after feature selection. Finally,

we tune the model with the best results which was Random Forest. Below is code snippet for tuned model:

RandomForestClassifier(n_estimators=600,random_state=1,criterion='entropy', max_depth=40,min_samples_leaf=5)

This model was trained with 6 selected features and gave 90% accuracy and 96% auc score. This model had max depth of 40 which helps in generalizing the model and adding minimum sample leaf as 5 increase accuracy as it indicated model when to split. Trees will split only when node had minimum 5 leaf. This same model was run with all the features, model had 89% accuracy and 96% auc score. This indicated using selected features increased overall performance.

Discussion

Method Comparison

In our problem we had to classify different twitter accounts based upon different features related to the account. We selected four classification models to perform this task. These classification models were random forest, logistic regression, Naïve Bayes, and Support vector machine.

Let us start with the analysis of Logistic Regression, which is widely used for binary classification. It predicts the class based on probabilities generated by logistic function. Even though it is simple and easier to implement and has in built capabilities of feature selection using coefficients and p value of the feature. But this method has major disadvantage that it assumes linearity in data and works best when all the target and input features are linearly dependent. We selected this model as one of the methods for classification as we had enough observations compared to number of input features and with initial assumption that features might be linearly separable. Features like if account is verified or not gave direct relation with target feature bot, indicating dataset might be linear. Even our correlation matrix displayed good correlation between verified and non bot account. But after analyzing the results we found data is not completely linear and there are input features which are dependent on each other. Hence, logistic regression does not seam to be best choice for this dataset. This is evident from the performance metrics as well. Maximum accuracy it could achieve was 0.75. We also tested Logistic Regression against different feature selection methods, but it did not show any improvement in terms of both generalization and accuracy. It gave better performance when all the features were used. It gave worst performance when features selected using random forest method was used. Major reason behind this is random forest does not give linearly separable features and hence Logistic Regression could not perform better and had worst accuracy of 0.68.

Next model we used was Naïve Bayes Classifier. Since NB is simple with very few parameters and is light weight model giving very fast prediction, we selected this model. But results were completely off. Performance of Naïve Bayes was the weakest among all the models. The major reason behind this could be that we had mix of datatypes in input features i.e., we had both continuous and binary features. Hence, we used Gaussian NB Classifier, but since not all the features were continuous, model could not predict accurately. Another major issue we think is one of the major drawbacks of naïve bayes classifier. It assumes “input feature independence”. This is

most of the time not the case in real world problems. Despite its speed, this model assumes that all the input features not dependent and hence does not solve the problem efficiently. It is also heavily dependent on training data and forms bias and which results in poor generalization. Even in our dataset by just looking into correlation matrix we can say that some features are heavily dependent on each other besides the target feature. Final evaluation metrics with all the features and with selected features both confirm this issue with Naïve Bayes Classifier. It gave worst accuracy of 0.67 with features selected by random forest, which was lowest among all the methods.

Third method we used was Support Vector Machine for classification. We used even linear SVC model for L1 based feature selection, which gave us 5 features out of 18 input features. We observed that results of SVM were slightly better than naïve bayes and logistic regression but were not able to meet the expectations. Despite using in feature selection this model could not surpass other methods we used. This model worked best with recursive feature elimination giving 0.84 auc and 0.75 accuracy score. But with random forest selection its generalization was best having 0.86 auc. It implies that it was able to work well with nonlinear and codependent dataset. But still, it has its limitations. It does not work well with large datasets and is not computationally good when dataset has more than 1000 rows. It is more effective in domains where dimensions are high, and observations are less. In our case we have balanced dataset and with more than 1k records. This might be reason it is not giving best results.

Finally let us discuss about the best models for our problem which was Random Forest Classifier. Not only for classification, but this method also performed best among different feature selection methods. It is a tree-based model which uses ensemble of many trees to predict final output. This feature enables RF to give better predictions. Another major benefit of this model is it even works good without any parameter tuning. Additionally, as mentioned previously our dataset had mix of continuous features and Boolean features (converted to categorical). Random Forest tends to handle such data very well. Hence, we can see that in the model results. Another major advantage of random forest is that it can handle noise and do not require feature scaling. Final benefit which majorly helped us getting better results for our dataset was capability of random forest to handle non-linear relationships well. It has its in-built feature selection process which helps in eliminating irrelevant features, thus saving additional computation required for feature selection separately. We got good generalization and accuracy i.e., around 0.89 using this model for our dataset.

Feature Comparison

We used four different feature selection methods and each of the method uses different approach for selecting the features. We have already shared the results of each method in previous section. First method we used was chi squared method. It uses statistical approach of checking relationships between input features and target features for selecting columns. Top 10 features selected by this method were favorites count, description subjectivity, listed count, name subjectivity, followers count, status subjectivity, default profile image, has extended profile image, default profile and verified in ascending order of importance. This method works well with categorical and continuous data and hence is applicable and convenient for our problem. But except for SVM this method did not outperform model with all the features. For all the

classification problems using all the features gave better accuracy except SVM. This method was easily able to find our most relevant features for target. But one important thing to highlight is the order of feature importance. Chi method gave highest importance to verified and default profile column. Verified column when marked true can indicate that account is non bot, but it is not necessary that if account is not verified then it will be bot. Hence this method again will work better with linearly separable dataset.

Next up we used recursive feature elimination method of feature selection using logistic regression estimator. Top 10 features this method selected were verified, status count, default profile, has extended profile, lang category, description subjectivity, status subjectivity, status polarity, name subjectivity and name polarity. It uses estimator recursively in predicting important features. It starts with all the features and then eliminating features to come with predictor with highest performance. After analyzing the results, it appears this method works well with logistic regression, NB and SVM. In these models, selected features outperformed the results with all the features. Reason behind this is that random forest has its own feature selector and selects different set of models however in RFE we are using logistic regression as meta estimator and till some extent assumptions of LR, NB and SVM appears similar i.e., linearity and feature independence.

We then used L1 penalty-based feature selection. In this method we use meta estimator which takes l1 penalty. We have use linear SVC with 0.01 as penalty and we got 5 features. Higher the C value more features it selects. Features selected were verified, has extended profile, status subjectivity, description subjectivity and default profile. It assigns weights to each feature and we can see negative weights as well. Negative means feature value related to label with value 0, in our case non bots. We got negative weights for verified, has extended profile and status subjectivity. Status subjectivity feature was created using status text. It highlights sentiment of the status. It also shows that more subjective the status more chances are it is posted by non bot account. Bots will mostly post negative and nonfactual status. Finally default profile had positive weight as we know more are the chances account is bot if it uses default profile settings.

Finally, we used random forest selector. It selected 6 features from dataset. Verified, followers count, status count, listed count, favorites count, friends count. Random forest selected different set of variables than above methods. It has some similarity with chi squared features. However, the order of features or importance is totally opposite. Random Forest gave verified column least importance whereas chi squared marked verified has most important feature. Friends count column was marked most important. This feature was not selected by any feature selection method. This is unique observation made by random forest. This relation was not obvious as friends count had very low correlation as well. We can relate to real world problem as well, when we have less friends on twitter chances are high that account will be a bot. Random Forest identifies such relationships and outperforms all other feature selector. But when Random Forest ran with all the features, accuracy is similar but generalization and efficiency increases. This is a tradeoff when using feature selection. Low values in Followers count and favorites count are also indicative of bots. But friends count signifies bots in better way, as there can be a case when someone follows the bot account, but chances are very low that bot will add any friend. Hence friend count is more important feature in selection.

Result Interpretation

One of the major conclusions from all the classification and feature selection methods we used is that our dataset is not linear and requires models to handle complex relationship. We can not just rely on statistical linear models like chi-squared and logistic regression for selection and classification. Main objective of this project was to use account level details to categorize that account as bot account or human account. What do we call a bot account? One that is not operated by humans but by machine. It is a computer program controlling to spread false propaganda or news. Main objective of such accounts is to make a trending hashtag on twitter or influence most other twitter users. They do not have vast network on twitter and work on silos. If we look at such accounts, they might not have profile pictures, most of the type they will have default settings on and most important they will have very few network connections. One of our final random forest methods selected verified account, followers count, status count, listed count, favorites count, and friends count as most important features. This result is close to real world scenarios. We know an account is marked as verified when twitter receives a form from user and then twitter performs necessary background checks for that user. There will never be a case when user marked as verified will be bot account. Hence, this column is clear relation with non bot account. If account is verified at first step itself, it can be marked as non bot. This is easily done by tree-based models like Random Forest. Most of the cases this column can be root node as it clearly divides the user base. When looking at other columns containing different counts related to user account, we can say that bots will generally have less friends count or follower's count. They might have unrealistic number of status counts. As bots do not manually posts status and hence do not have human limitations in typing and posting status. Even though we had 18 features after some nlp and feature engineering, random forest still used columns present originally in dataset showing that those features were enough for classification.

Having said that, we cannot ignore the bias machine add to the real-world problem. We can have non bot accounts with no friends or followers at all. Even I have a twitter account with 5 followers and 4 friends. Also, there are many human users who do not change default settings. Even many twitter employees have no profile picture set in their accounts. I also use twitter just for surfing and checking the global updates and for most of the time I do not add any image. These all are the cases of false positives and are the tradeoff we must accept when dealing with such problem. Models will always be data dependent and invariably add bias to the problem. As a data miner and developer, we can try to implement best models and methods and take advantage of those methods. Even in that process we find users who are not bots but are nearly invisible and use default settings classified as bots and will add to false positives, but we cannot avoid the larger benefit we get using the models that is to find bots and label them. There is only an extent models can work after that data interpretation and analysis plays a major part in further finding the actual bots and non bots from the data presented. Once data is labelled users can further dig into the account details and activities to be aware that this is an account which is having some anomaly in its features.

To conclude we can say that feature selection significantly improves model performance. We can further add some feature engineering on selected features to enhance model predictions.

References

[1] Y. Lin, “10 Twitter Statistics Every Marketer Should Know in 2021 [Infographic],” *Oberlo*, 22-Mar-2021. [Online]. Available: https://www.oberlo.com/blog/twitter-statistics. [Accessed: 06-Apr-2021].

[2] “Twitter Bot Detection Guide to Protecting Your Brand,” *IZEA*, 23-Oct-2019. [Online]. Available: https://izea.com/2018/10/09/twitter-bot-detection-101/. [Accessed: 06-Apr-2021].

[3] K. Hao, “Nearly half of Twitter accounts pushing to reopen America may be bots,” *MIT Technology Review*, 10-Dec-2020. [Online]. Available: https://www.technologyreview.com/2020/05/21/1002105/covid-bot-twitter-accounts-push-to-reopen-america/. [Accessed: 06-Apr-2021].

[4] C. Jain, “detecting twitter bot data,” *Kaggle*, 01-Dec-2018. [Online]. Available: https://www.kaggle.com/charvijain27/detecting-twitter-bot-data. [Accessed: 06-Apr-2021].

[5] Efthimion, Phillip George; Payne, Scott; and Proferes, Nicholas (2018) "Supervised Machine Learning Bot Detection Techniques to Identify Social Twitter Bots,"SMU Data Science Review: Vol. 1: No. 2, Article 5. Available at: https://scholar.smu.edu/datasciencereview/vol1/iss2/5

[6] R. Battur and N. Yaligar, “Licensed Under Creative Commons Attribution CC BY Twitter Bot Detection using Machine Learning Algorithms,” *International Journal of Science and Research (IJSR) ResearchGate Impact Factor*, vol. 8, 2018, doi: 10.21275/ART20199245.

[7] V. Shelke, C. Praveen, Shetiye, K. Avinash, Gulve, and M. Tech Student, “Twitter Bot Detection Using Machine Learning,” *IJSRD -International Journal for Scientific Research & Development*|, vol. 8, pp. 2321-0613, 2020, Accessed: Apr. 06, 2021. [Online]. Available: http://www.ijsrd.com/articles/IJSRDV8I50085.pdf.