Social Bots Detection on Twitter

Qiwei Zhou
Tandon School of Engineering
Brooklyn, New York
qz729@nyu.edu

Jiao Wu
Tandon School of Engineering
Brooklyn, New York
Jw4321@nyu.edu

Abstract:
The objective of this project to detect the twitter bots using machine learning algorithms and compare the performance of three different machine learning algorithms, including Naïve Bayes, Random Forest, Logistic Regression.

Keywords—machine learning

I. INTRODUCTION

As the popularity of social networking and microblogging tools continues to grow, accounts controlled by automated programs, known as social bots (or bots for short), have risen from the horizon. Twitter, with the openness of its platform and API, has seen the rise of the machines. These bots automatically produce contents and even communicate with humans, trying to hide themselves among the mass population of Twitter. In this project, we are going to compare the performance of three different machine learning algorithms for detecting bot accounts.

II. MOTIVATION

Although plenty of bots on Twitter are benign, there are still bad bots that mislead, exploit, and manipulate online conversations with rumors, spam, phishing links and slander [1]. These bots randomly add other users as their friends and if the user follows back, the malicious contents will be displayed on his homepage [2]. They can also be alleged to help political candidates in the election [3], potentially undermining the democratic system. Therefore, we need to develop a method to detect these bots in social media, either to help Twitter manage the community [4], or to help human users identify who they are communicating with.

III. RELATED WORK

[1] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2015. The Rise of Social Bots. X, X, Article XX (201X), 11 pages.

[2] Z. Chu, S. Gianvecchio, H. Wang and S. Jajodia, "Detecting Automation of Twitter Accounts: Are You a Human, Bot, or Cyborg?," in IEEE Transactions on Dependable and Secure Computing, vol. 9, no. 6, pp. 811-824, Nov.-Dec. 2012. doi: 10.1109/TDSC.2012.75

[3] J. P. Dickerson, V. Kagan and V. S. Subrahmanian, "Using sentiment to detect bots on Twitter: Are humans more opinionated than bots?," 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014), Beijing, 2014, pp. 620-627.

[4] N. Chavoshi, H. Hamooni and A. Mueen, "DeBot: Twitter Bot Detection via Warped Correlation," 2016 IEEE 16th International Conference on Data Mining (ICDM), Barcelona, 2016, pp. 817-822.

[5] 15 Awesome Twitter Bots You Should Follow, Rajat Sharma. Available: http://beebom.com/best-twitter-bots/

[6] 12 Weird, Excellent Twitter Bots Chosen by Twitter's Best Bot-Makers,Lainna Fader. Available: http://nymag.com/selectall/2015/11/12-weirdest-funniest-smartest-twitter-bots.html

[7] Bot or Not: an end-to-end data analysis in Python, Erin Shellman. Available: http://www.erinshellman.com/bot-or-not/

IV. DATA

In this project, we need data to train and test our bot detecting machine learning algorithms. We collected account information for 100 different Twitter users, half of them are bots and the others are not bots. Here, the bot Twitter accounts are chosen from Internet search (confirmed via manual inspection) [5,6] and not bot accounts are collected from our own Twitter social network. With their account name known, we can exploit the Twitter API functions to gather their detailed user information. We used REST APIs to retrieve our required user information, ranging from profile, follower, friend lists, status etc., For each user, we called GET users/lookup method to collect user information and could query for up to 100 users per request. This is an efficient way to acquire detail information for a large number of accounts. Based on the previous observations of differences among human and bot accounts [3], we selected 20 data fields totally for each of the 100 users for future feature extraction and analysis. The details of dataset are saved as a CSV file for following parts of this project.

V. ALGORITHM(S) USED

In this project, we used three machine learning algorithms, Naïve Bayes, Random Forest and Logistic Regression.

Naïve Bayes is a classification method using probability calculation. It is based on the (strong) assumption that all attributes are independent from each other. Although this assumption is not true in most cases, Naïve Bayes does give good results. There are several different ways of Naïve Bayes. In this project, we used both Bernoulli and Multinomial Naïve Bayes algorithms. Also, when dealing with continuous values, we also applied Gaussian Naïve Bayes.

Random Forest is an ensemble learning method. Single decision trees are likely to overfit on the training set. Random Forest is designed to handle this issue by growing base decision trees in a random effect. Each tree is constructed on a bootstrapped dataset and at each split, only a limited number of random features are considered. In this way, Random Forest is less vulnerable to overfitting. In this project, we will use Random Forest as a classification method.

Logistic Regression is a regression model in which the value we are using is categorical. Therefore, logistic regression is a useful classification method. Since we are interested in distinguishing bots from non-bots, we will be using binary logistic regression in our project.

With the help of scikit-learn python machine learning package, we don’t have to implement the algorithms on our own. In our implementation, we will run our features on these two different models and compare their performance.

VI. RESULT

In our experiment, 2232 different Twitter accounts were considered, where 1056 are bots and the other 1176 are non-bots. The features we analyzed were status, description, number of followers, friends, listed-counts, favorites; whether they had default profiles, default profile images and whether they have extended profile or were verified.

In our implementation, we mainly used python data analysis library, pandas to read and process data and scikit-learn API for machine learning algorithms. Pandas makes it simple to apply custom methods on the dataset for our feature extraction and scikit-learn provides the ease to construct and evaluate our estimator.

To distinguish between bots and non-bots, first, we need to find where they are different. Therefore, our first step was to do some exploratory data analysis.

After a quick look at the data, we found that the description and status fields are text data. We can use the Bag of Words model to do classification. There are three different vectorizers in the Bag of Words model in sci-kitl learn package, CountVectorizer, TfidfVectorizer and HashingVectorizer. We will try them all to see which one produces the best result.

Before we do feature extraction, first we filled the NaN field in the target columns with a special value. Then we divide the whole dataset into two sets for training and testing separately. Here are our accuracy results for the test set:

	BernoulliNB	**MultinomialNB**	**RF**	**LR**
CountVectorizer	75.81%	74.55%	70.25%	76.16%
TfidfVectorizer	75.26%	75.62%	72.04%	74.19%
HashingVectorizer	53.58%	74.73%	69.89%	73.66%

Table 1. Accuracy Results on Description

	BernoulliNB	**MultinomialNB**	**RF**	**LR**
CountVectorizer	71.33%	73.11%	75.63%	78.50%
TfidfVectorizer	72.40%	70.07%	70.79%	75.27%
HashingVectorizer	53.58%	56.45%	72.22%	74.01%

Table 2. Accuracy Results on Status

We can see that the Hashing Vectorizer is not working well in our model. We will not use it in the future. Also, the overall accuracy of all three machine learning algorithms are similar to each other, with Logistic Regression works a little better.

Before we train the model using other features, such as number of followers, friends, etc., we first attempt to find where they are different between bots and nonbots.

For the number of followers in two groups, we found that bots were more likely to have zero followers than non-bots(Figure1).

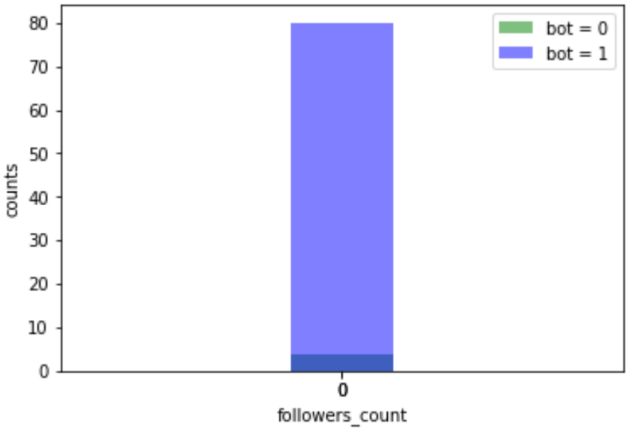

Figure 1. Number of zero followers in bots and non-bots.

Then we went further to check which range of number of followers could be used to significantly distinguish these two groups. We found that there is a big difference of range of 0 to 200 followers between these two groups (Figure 2), showing that bots were more likely to have less than 200 followers than nonbots. However, nonbots were more likely to have more than 10000 followers than bots. We concluded that having few followers were more likely to be bots while having a large number of followers were more likely to be nonbots.

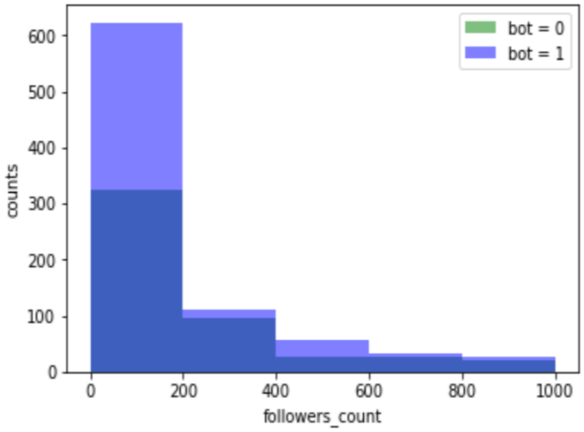

Figure 2. Number of followers with range of 0 to 1000 in bots and nonbots

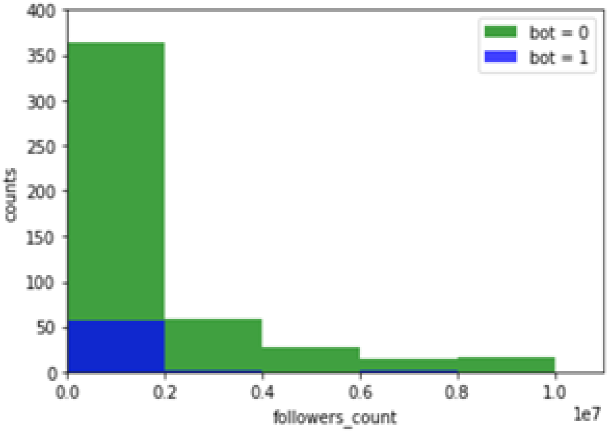

Figure 3. Number of more than 10000 followers in bots and nonbots.

Likewise, we also tested another feature, number of friends in these two groups and found that bots have more zero follower than nonbots (Figure 4).

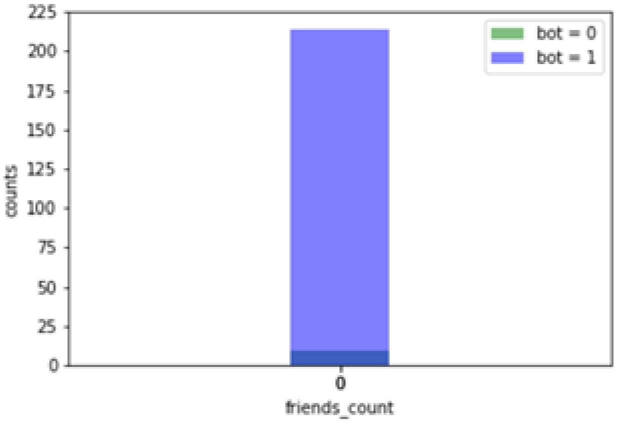

Figure 4. Number of 0 friend in bots and none bots.

After checking another range, it turned out that there were more bots than nonbots in range of less than 6 friends (Figure 5), whereas there were more nonbots than bots in range of 40 to 1600 friends (Figure 6). Similar result was obtained in range of more than 1600 friends. It determined that bots have less friends than nonbots.

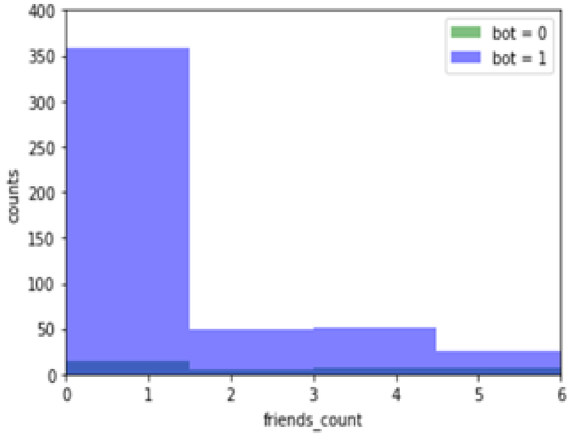

Figure 5. Number of less than 6 friends in bots and nonbots.

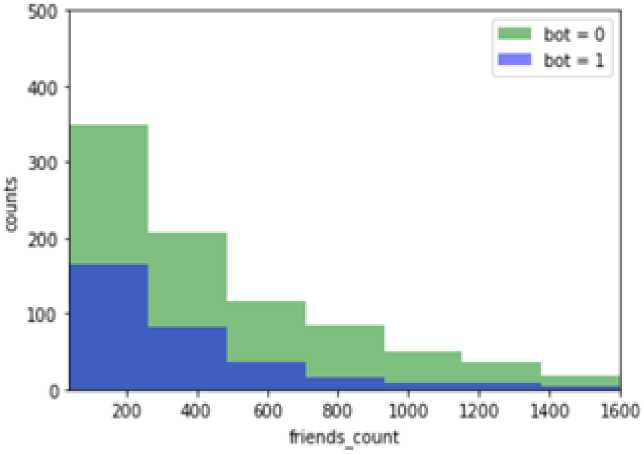

Figure 6. Number of friends with range 40 to 1600 in bots and nonbots.

Next, we checked the number of listedcount in two groups. Again, there were more bots having zero listed counts than nonbots(Figure 7) and we also found that there were more bots than nonbots having listed counts in range of 4 to 60(Figure 8). However, nonbots were more likely to have more than 200 listed counts than bots (Figure 9).

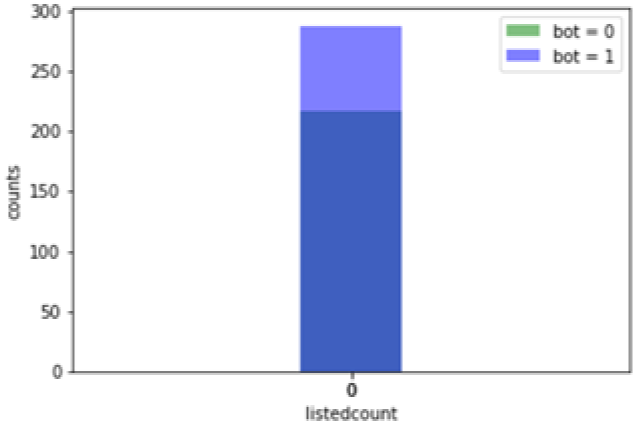

Figure 7. Number of zero listedcount in bots and nonbots.

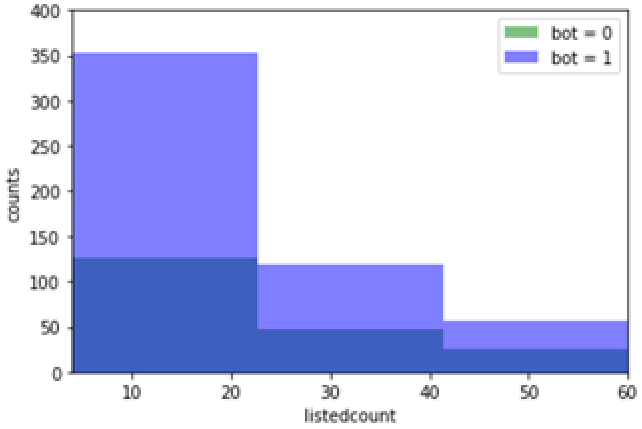

Figure 8. Number of listedcount with range 4 to 60 in bots and nonbots.

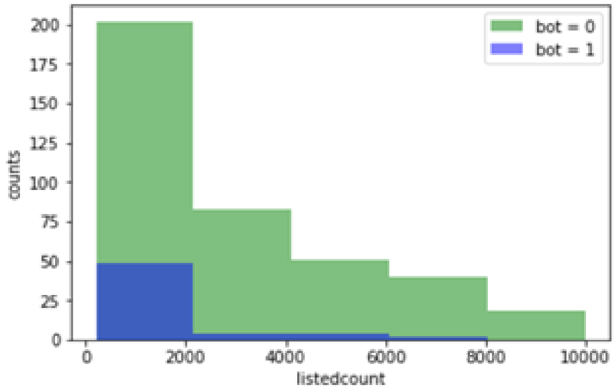

Figure 9. Number of more than 200 listedcount in bots and nonbots.

It also showed that having more than 16000 listed counts were more likely to be nonbots (figure 10). Based on this finding, we concluded that bots were more likely to have less listed counts than nonbots.

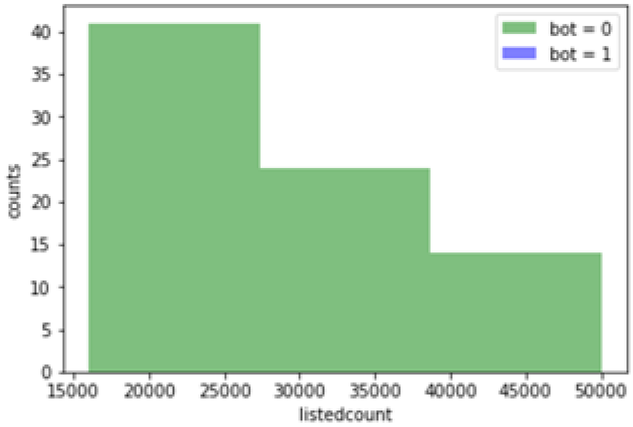

Figure 10. Number of more than 16000 listed counts in bots and nonbots.

Similar analyzed was done in favourites_count. We got similar conclusion that bots were more likely to have less than 5 favourites_count than nonbots (figure 11 and figure 12), whereas nonbots were more likely to have more favourites_count than bots (figure 12).

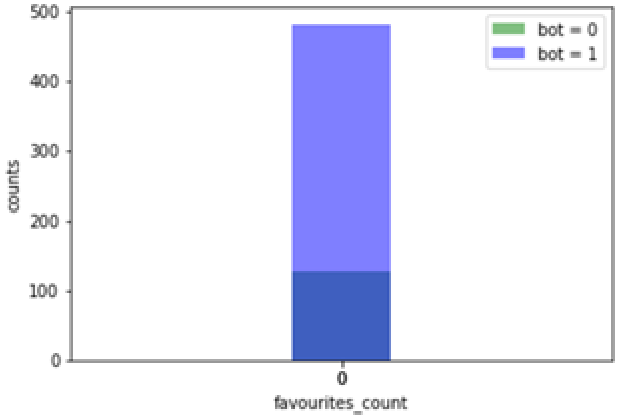

Figure 11. Number of zero favourites count in bots and nonbots.

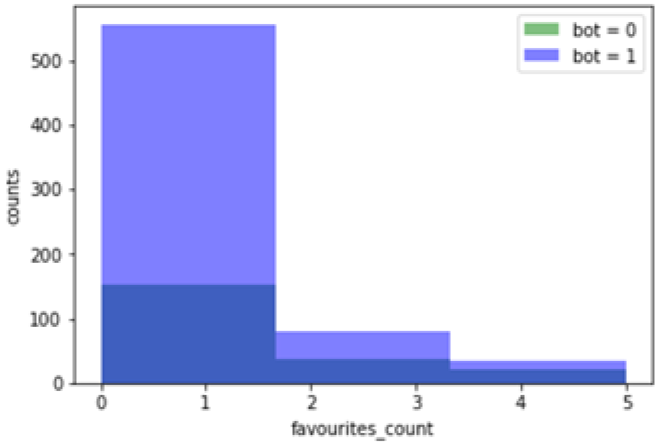

Figure 12. Number of less than 5 favourites_count in bots and nonbots.

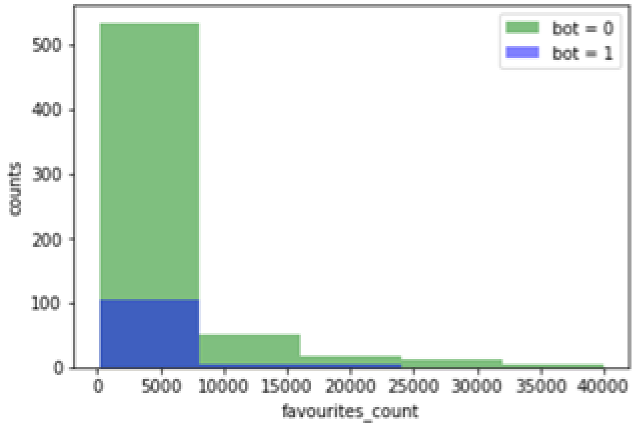

Figure 13. Number of more than 150 favourites_count in bots and nonbots.

We also analyzed the feature of whether the account was verified or not and concluded that there were more bots which were not verified and there were more nonbots which were verified (figure 13).

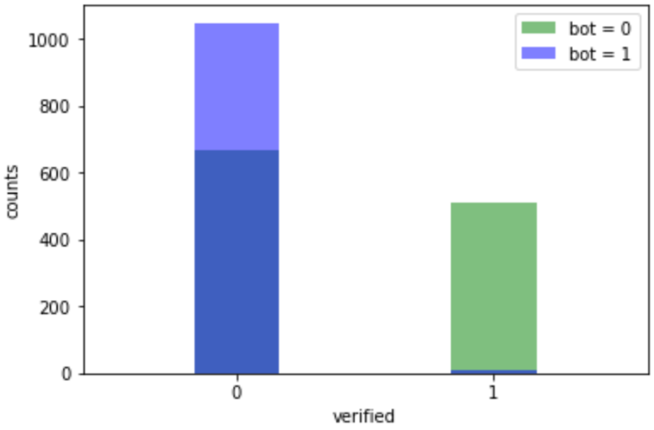

Figure 14. Counts of verified or non-verified in bots and nonbots.

For the feature whether they had default profiles or not, we found that bots had a default profile (more than half of bots did) while 75% of nonbots did not have a default profile (figure 15).

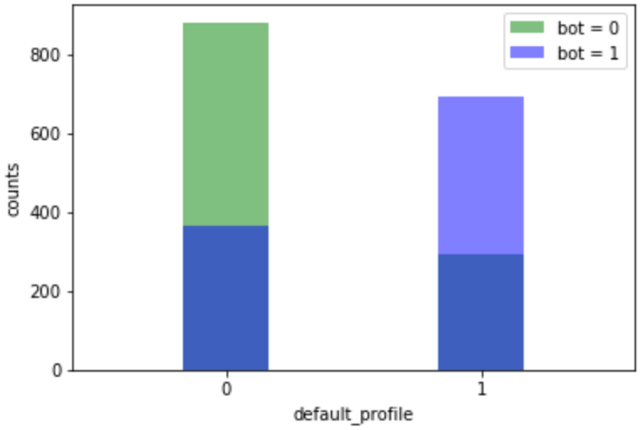

Figure 15. Counts of having default_profile or no default_profile in bots and notbots.

We also checked whether they had default profiles images or not in bots and nonbots. We concluded that both did not have default profile images and there was no significant different between these two groups (Figure 16).

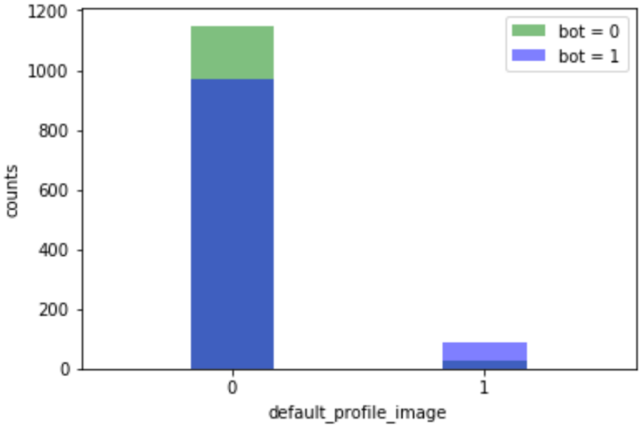

Figure 16. Counts of having default profile images or no default profile image in bots and notbots.

Finally, we checked whether they had extended profiles or not in bots and nonbots. We concluded that almost both did not have extended profiles and there was no significant different between this two groups (Figure 17).

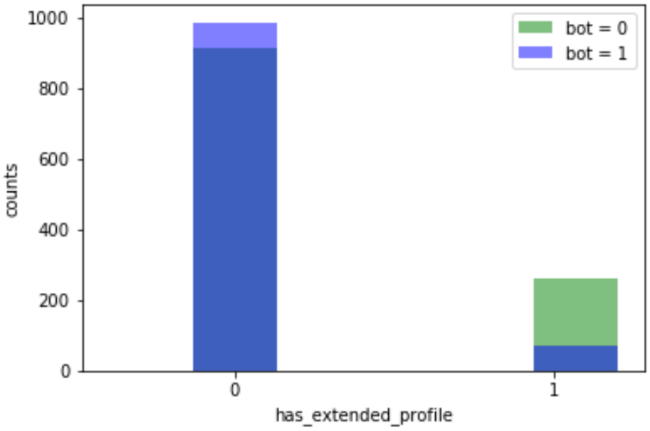

Figure 17. Counts of having extended profile or no extedned profile in bots and notbots.

Based on above findings, we attempted to train a model using Naïve bayes, Random forest, and Logistic Regression. The results of accuracy and other performance parameters were shown as follows:

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	95.24%	95.24%	100%	97.56%	50.00%
MultinomialNB	95.24%	95.24%	100%	97.56%	50.00%
RF	95.24%	95.24%	100%	97.56%	50.00%
LR	95.24%	95.24%	100%	97.56%	50.00%

Table 3. Results on number of zero followers

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	83.54%	83.54%	100%	91.03%	50.00%
MultinomialNB	83.54%	83.54%	100%	91.03%	50.00%
RF	83.54%	83.54%	100%	91.03%	50.00%
LR	83.54%	83.54%	100%	91.03%	50.00%

Table 4. Results on number of less than 11 followers

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	97.03%	0.00%	0.00%	0.00%	50.00%
MultinomialNB	97.03%	0.00%	0.00%	0.00%	50.00%
RF	96.04%	0.00%	0.00%	0.00%	49.49%
LR	97.03%	0.00%	0.00%	0.00%	50.00%

Table 5. Result number of more than 100000 followers

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	92.86%	92.86%	100.00%	96.30%	50.00%
MultinomialNB	92.86%	92.86%	100.00%	96.30%	50.00%
RF	92.86%	92.86%	100.00%	96.30%	50.00%
LR	92.86%	92.86%	100.00%	96.30%	50.00%

Table 6. Result on number of zero friend

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	94.66%	94.66%	100.00%	97.26%	50.00%
MultinomialNB	94.66%	94.66%	100.00%	97.26%	50.00%
RF	94.66%	94.66%	100.00%	97.26%	50.00%
LR	94.66%	94.66%	100.00%	97.26%	50.00%

Table 7. Result on number of less than 7 friend

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	72.68%	72.68%	100.00%	84.18%	50.00%
MultinomialNB	72.68%	72.68%	100.00%	84.18%	50.00%
RF	72.68%	72.68%	100.00%	84.18%	50.00%
LR	72.68%	72.68%	100.00%	84.18%	50.00%

Table 8. Result on number of listed count between 4 to 60

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	97.47%	0.00%	0.00%	0.00%	50.00%
MultinomialNB	97.47%	0.00%	0.00%	0.00%	50.00%
RF	96.20%	0.00%	0.00%	0.00%	49.35%
LR	97.47%	0.00%	0.00%	0.00%	50.00%

Table 9. Result on number of more than 3000 listed count

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	78.29%	78.29%	100.00%	87.82%	50.00%
MultinomialNB	78.29%	78.29%	100.00%	87.82%	50.00%
RF	78.29%	78.29%	100.00%	87.82%	50.00%
LR	78.29%	78.29%	100.00%	87.82%	50.00%

Table 10. Result on number of zero favourites

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	90.41%	0.00%	0.00%	0.00%	50.00%
MultinomialNB	90.41%	0.00%	0.00%	0.00%	50.00%
RF	87.67%	0.00%	0.00%	0.00%	48.50%
LR	90.41%	0.00%	0.00%	0.00%	50.00%

Table 11. Result on number of favourites between 2000 to 40000

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	91.67%	0.00%	0.00%	0.00%	50.00%
MultinomialNB	91.67%	0.00%	0.00%	0.00%	50.00%
RF	70.83%	0.00%	0.00%	0.00%	38.64%
LR	79.17%	0.00%	0.00%	0.00%	43.18%

Table 12. Result on number of more than 10000 favourites

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	69.53%	60.62%	98.07%	74.93%	71.44%
MultinomialNB	69.53%	60.62%	98.07%	74.93%	71.44%
RF	69.53%	60.62%	98.07%	74.93%	71.44%
LR	69.53%	60.62%	98.07%	74.93%	71.44%

Table 13. Result on whether they were verified or not

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	70.61%	69.08%	66.41%	67.72%	70.33%
MultinomialNB	70.61%	69.08%	66.41%	67.72%	70.33%
RF	70.61%	69.08%	66.41%	67.72%	70.33%
LR	70.61%	69.08%	66.41%	67.72%	70.33%

Table 14. Result on whether they had a default profile or not

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	56.27%	82.61%	7.34%	13.48%	53.00%
MultinomialNB	56.27%	82.61%	7.34%	13.48%	53.00%
RF	56.27%	82.61%	7.34%	13.48%	53.00%
LR	56.27%	82.61%	7.34%	13.48%	53.00%

Table 15. Result on whether they had a default profile image or not

	Accuracy	Precision	Recall	F1	AUC
BernoulliNB	52.517%	49.37%	90.35%	63.85%	55.04%
MultinomialNB	52.517%	49.37%	90.35%	63.85%	55.04%
RF	52.517%	49.37%	90.35%	63.85%	55.04%
LR	52.517%	49.37%	90.35%	63.85%	55.04%

Table 16. Result on whether they had an extended profile or not

Looking at the results above, we can see that accuracy and other parameters of all three machine learning algorithms are same. Also, we can see that feature whether they have a default profile images or not and feature whether they have an extended profile or not are not working well in our model because of poor accuracy. We will not use it in the future.

VII. CODE

https://github.com/qiweizhou94/ML_TwitterProject

VIII. VIDEO LINK

(Create a 5-10 min video of your project presentation. We want all the members to present. Include a link to your presentation video. You will upload this video to youtube and paste the link here. Do not attach the video. For more info on contents see the Final Project writeup.)

IX. EVALUATION

(Please comment on what went well and what didn’t go well in your experiment. If you had more time what could have been improved?)

X. CONCLUSION