Terminator 6: Humans vs Bots (on Twitter)

Munaf Arshad Qazi
New York University
maq249@nyu.edu

Maksim Temnogorod
New York University
mt1697@nyu.edu

***Abstract*—The dynamics of social media have become increasingly complex. One topic of particular relevance is the existence of automated bots, whose activity has proven to have significant societal consequences. Twitter, used by over 300 million users, is home to around 20 million such bots. The task of differentiating between these accounts and genuine human users is made all the more challenging by their variety. We are interested in studying the behavior of different bots, in part as an effort to promote Twitter as an open and unbiased platform.**

***Keywords*—twitter, machine learning, logistic regression, random forests, NLP.**

I. Introduction

Social media is one of the most powerful platforms for people to voice and propagate their ideas. However, it is often difficult to know exactly who (or what) is listening, forwarding, or even generating a user's words. The existence of bots on social media platforms has gained significant attention in recent years with respect to inflated follower counts for political candidates and other public figures on Twitter [1]. While some work in the detection of bot accounts continues to be conducted with the explicit perspective of minimizing spam, the broader Twitter bot landscape is likely more complex, and the variety of purposes for which bots are built should be taken into account.

We have observed that among the different bots inhabiting twitter, the least interesting (though still potentially problematic) are those that at first glance resemble an ordinary person but are in fact fake, existing purely as follower-list fodder. They are easily identified by a low follower/following ratio [3] and little activity otherwise. Another class of bots are novelty accounts such as those that compose linguistic experiments or else parody some cultural trope, often to humorous effect. These are distinguishing by high follower/following ratios (perhaps comparable to celebrities), frequent and temporally regular tweets, but little like/retweet activity. Also, they are often honest about their bot status. We also recognize a third class, identifiable by high follower and friend counts and more interaction in the form of likes and retweets. This includes special interest/news aggregates as well as advertising and phishing accounts. These bots may be either helpful or malicious, deceptive or overt, and are probably most difficult to distinguish from human users based on these features alone. We intend to factor these different classes of bots into our approach.

II. Motivation

In addition to being behind the popularity inflation of political figures, the use of Twitter bots have been known to promote misinformation and even dramatically influence the stock market [2]. Given the vast popularity of social media and the anonymity of online interactions, it is unsurprising that bots are capable of such subversive activity. While the potential to challenge democracy is on the more critical end of the bot-danger spectrum, on the other end there is spam and phishing. It is important to continuously improve bot detection methods, in part by paying attention to the diversity of bots on Twitter.

III. Related Work

[1] J. Dickerson, V. Kagan and V. Subrahmanian, "Using Sentiment to Detect Bots on Twitter: Are Humans more Opinionated than Bots?", in International Conference on ASONAM, 2014.
[2] E. Ferrara, O. Varol, C. Davis, F. Menczer and A. Flammini, "The rise of social bots", Communications of the ACM, vol. 59, no. 7, pp. 96-104, 2016.
[3] E. Shellman, "Bot or Not: an end-to-end data analysis in Python", 2015.

IV. Data

Our dataset consists of 1056 Twitter accounts labeled as bots and 1176 Twitter accounts labeled as non-bots. The collection of accounts and manual labeling was largely crowdsourced, which contributes to the challenge of producing reliable models. Each of the 2232 accounts is represented by attributes (shown below with corresponding data type) pulled from the Twitter API.

The feature 'bot' represents the assumed label: 1 for a bot account, 0 for a non-bot. See the Twitter API for more information on the other attributes: https://dev.twitter.com/overview/api/users.

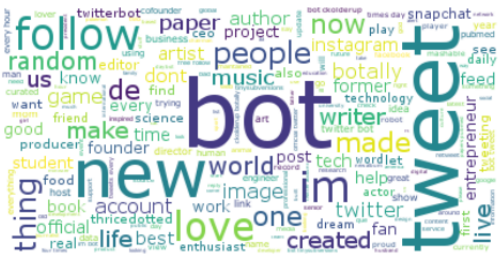
id	Int64
id_str	String
screen_name	String
location	String
description	String
url	String
followers_count	Int
friends_count	Int
listed_count	Int
created_at	String
favourites_count	Int
verified	Boolean
statuses_count	Int
lang	String
status	Tweet (JSON String)
default_profile	Boolean
default_profile_image	Boolean
has_extended_profile	Boolean
name	String
bot	Int

V. Algorithms Used

We evaluated a variety of models by training several different classifiers: namely, random forest, decision tree, logistic regression, and Bernoulli and multinomial Naive Bayes algorithms were used. We achieved our best performance by using a kind of stacked classifier approach in which the output of a Naive Bayes classifier trained on a set of account descriptions is used as a feature among the relevant attributes used to train another classifier.

VI. Results

A. Lexical Analysis



To extract possible linguistic information from the feature vectors, we considered the 'description' attribute, which is essentially a user's brief bio. Initial analysis using word count vectors did not seem promising as a classification feature, but by normalizing the text (in this case, by removing punctuation and lowering case) and training on these instances, a Naive Bayes classifier was able to achieve 76.5% accuracy on a test set. Admittedly, the performance of this classifier alone does not support a reliable model, but as the preceding word cloud and the following list of ten most salient features demonstrates, the classifier was beginning to extract some relevant lexical information from the relatively limited dataset.

Most Informative Features

contains(bot) = True	1 : 0 = 27.0 : 1.0
contains(tinysubversions) = True	1 : 0 = 17.7 : 1.0
contains(tweeting) = True	1 : 0 = 17.0 : 1.0
contains(former) = True	0 : 1 = 12.6 : 1.0
contains(every) = True	1 : 0 = 11.5 : 1.0
contains(producer) = True	0 : 1 = 10.8 : 1.0
contains(husband) = True	0 : 1 = 10.8 : 1.0
contains(editor) = True	0 : 1 = 10.1 : 1.0
contains(writer) = True	0 : 1 = 9.8 : 1.0
contains(random) = True	1 : 0 = 9.8 : 1.0

Interestingly, the most informative words for identifying non-bots (marked by a high 0:1 ratio) describe familial or professional roles, while the presence of overt *bot*s who *random*ly post a *tweet* *every* given unit of time helps to identify bots across the dataset. (The influence of tinysubversions, the AI/art project of Darius Kazemi becomes apparent, too.)

B. Numerical Analysis

For our numerical analysis, we looked at strictly numerical and boolean features in the dataset. The features considered are shown below along with their data types.

followers_count	Int
friends_count	Int
listed_count	Int
favourites_count	Int
verified	Boolean
bot	Boolean

Note: We didn't consider temporal features like created_at, and some other boolean features like has_extended_profile / default_profile because for the preliminary analysis they seemed irrelevant. Also, keeping them in would have caused the algorithm performances to go down.

C. Cleaning and Preprocessing

Once the dataset was imported using pandas, we refilled nulls with '?' so they don't cause any further problems. All fields were verified to be the data type they're supposed to be and the irrelevant fields were munged away.

Next we concatenated the bots and non bots data frames and shuffled them using a random function. We then divided the data into an 80% Training Set and 20% Validation Set. The data set was shuffled uniformly and randomly for a least possible selection bias.

D. Visualization & Analysis

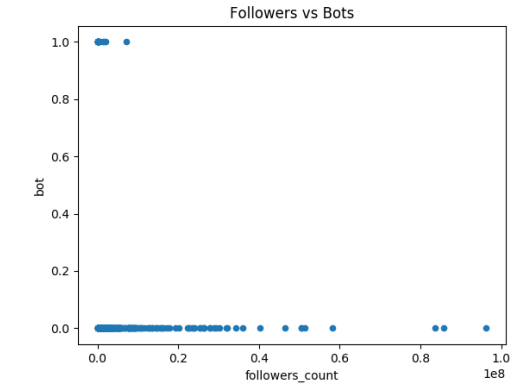
Next we analyzed the data to see how different features related to the probability whether an ac-

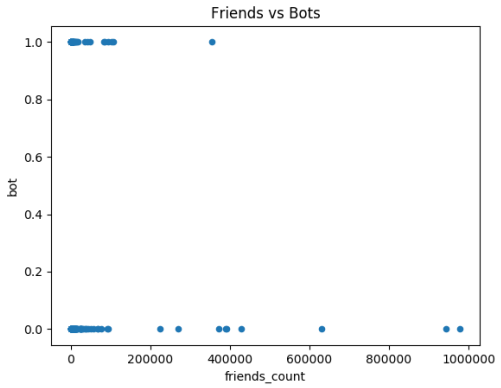
count is a bot or not. Some top correlating features include followers count, friends count and verified.

The major plots obtained are shown in the figure above. It is interesting to note how diverse the data set is in terms of number of followers when it talks about humans, but it isn't as diverse for bots. This is an important bias or possible hypothesis to consider for the model. This data variation is also visible looking at the friends count graph.

One more thing analyzed was the number of verified bots vs number of verified humans because about these we can be certain and the tags have been provided by twitter itself. Running our scripts gave us the results in table 2 and they uncovered that the data is kind of skewed in this sense too and we will have to blindly learn features from the data collected taking forward the multiple assumptions of people who helped gather it.

verified humans	509
verified bots	7





E. Model Performance

Considering the numerical features and possible models with them, we decided to work with a few well known algorithms to see how well they performed. The computations were done using Decision Trees, Random Forests, Logistic Regression (both Lasso and Ridge), and Naive Bayes (both Multinomial and Bernoulli) and the results and their ROC curves were compared against each other. The accuracy results for all these are presented in Table 3 and 4.

Name	Accuracy on Holdout Set
Random Forest	0.866
Decision Trees	0.839
Bernoulli NB	0.752
Multinomial NB	0.653

Logistic Regression was tried with different L1 and L2 regularizations along with different weights. The results are presented below.

Weight	Accuracy (L1)	Accuracy (L2)
100	0.6846	0.6801
1	0.6890	0.6801
0.01	0.6957	0.6801

Random Forest

This was the best overall performing algorithm with the best accuracy. This is a definite yes if complexity isn't an issue for the final model's implementation.

Decision Trees

Trees were the second best to perform. It is interesting to note that these trees were 99% accurate on the training set, so it is right to assume that they are over fitting here. Reducing the entropy or specifying depth and leaves can yield even better results.

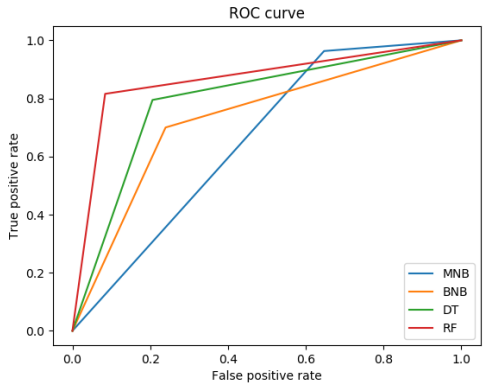
Logistic Regression

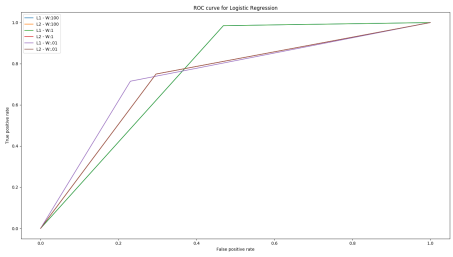
Logistic Regression results weren't as impressive. Even though a small weight yielded the most accurate Logistic Regression Model. Because regularization played a weak role in improving the model, we're assuming the model is under fitting and needs to be more carefully designed.

Naive Bayes

Even though Naive Bayes performed better that Logistic Regression, the results still were not as impressive with Multinomial Naive Bayes being the weakest of them all.

F. Receiver Operator Curve





The Receiver Operator Curves show how the models performed against each other. The best model was Random Forests with a very low false positive rate. The worst one although was Multinomial Naive Bayes, one must also consider that it had the highest true positive rate (if false positives weren't as important for the problem).

G. Bagging Classifiers

We decided to do some feature engineering as mentioned in the Lexical Analysis section and were able to calculate a Naive Bayes estimate of how the different words in account descriptions related to the account being a bot. With the results we were able to make a guess of whether an account is a bot. This guess was then incorporated with all the other numerical features and the numerical analysis was repeated. The accuracy results and roc curves look promising for a preliminary unoptimized run of this model.

Name	Accuracy on Holdout Set
Random Forest	0.942
Decision Trees	0.881
Bernoulli NB	0.897
Multinomial NB	0.671

Weight	Accuracy (L1)	Accuracy (L2)
100	0.875	0.709
1	0.875	0.709
0.01	0.839	0.709

Random Forest

This run of the random forest exceeded our expectations. We, however, should keep in mind that because the data was randomly shuffled, this could be a lucky run. Multiple runs of the model showed an average of around 92% which again is pretty impressive.

Decision Trees

It is interesting to note that these trees were 99% accurate on the training set again, so it is right to assume that they are over fitting here. Reducing the entropy or specifying depth and leaves can improve the model and reduce generalization error.

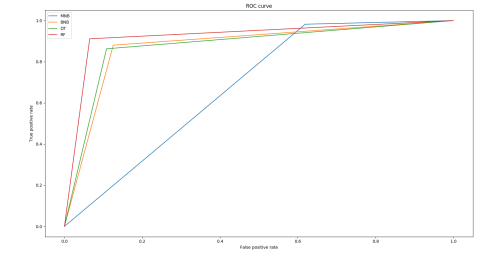
Logistic Regression

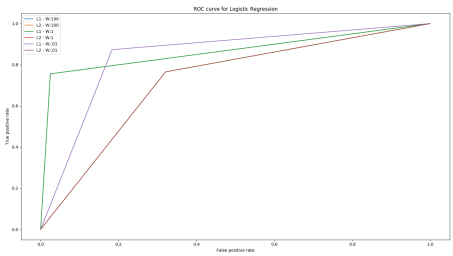
Logistic Regression results were pushed up ten folds for L1 regularization. The L2 regularized model struggled to perform yet again.

Naive Bayes

Bernoulli Naive Bayes performed better that Logistic Regression, however this time Logistic regression was able to surpass decision trees and Multinomial Naive Bayes proving itself to be still a strong competitor - not out of the race!

Receiver Operator Curves





The new ROC curve shows a huge tilt towards true positives for each classifier. This improvement in both accuracy and ROC score exhibits that using Naive Bayes for Lexical Analysis and Feature Engineering and then using the new feature to boost results by bagging classifiers was a smart move. Links to all ROC curves and graphs are provided in the next section and can be opened separately for better visibility and clarity.

VII. Code

The preliminary code can be found here (https://github.com/maqzi/Twitter-Bot-Detection). Also, all figures included in this document have been uploaded to github too and can be analyzed in depth from there. They can be accessed here.