Information Gain of Twitter 'Social Spambots'

Joaquin Quintana
Applied Computer Science
University of Colorado
Boulder, Colorado, USA
Joaquin.Quintana@Colorado.edu

Jason Weidner
Applied Computer Science
University of Colorado
Boulder, Colorado, USA
Jason.Weidner@colorado.edu

Abstract

We calculated the information gain for a few simple rules to establish the preliminary research needed to build a decision tree classifier for automated social spambot identification of spam accounts on Twitter. The rules selected can easily be calculated through the Twitter API and were identified by other researchers as effective for the identification of traditional Twitter spambots [3]. We validated our code by calculating the information gain on traditional Twitter spambots in a new dataset and compared our results to those published on older data sets of traditional Twitter spambots. We used our validated code to calculate information gain of the same attributes on a collection of *social spambots*, a new type of spambot developed to evade modern detection algorithms.

Our results show that the number of tweets authored by an account, the number of followers an account has, and the number of accounts an account follows are poor rules to use when identifying social spambots. This is generally in contrast with traditional Twitter spambots.

Furthermore, social spambots can be generated for different reasons and the information gain for these rules vary by social spambot subtype.

We combined the rules using a random forest algorithm to generate a relatively high accuracy classifier. Using a dataset of approximately 500 accounts, our classifier had an accuracy of 96% and uses only four rules.

The dataset we used is a subset of [1] based on rules proposed by [7] and presented in [3].

KEYWORDS

Social spam bot, twitter, fake followers, information gain, decision tree, random forest classifier

1. Introduction

Researchers have developed many techniques to identify the spambots that plague social media platforms [1,2,3,4,5,7]. However, new bot technologies are routinely being improved upon, developed specifically to evade known identification techniques.

Social spambots, a relatively new vintage of Twitter spambot, which are known to mostly tweet benign content while working in a highly coordinated fashion to promote specific malignant content such as political propaganda during an election. Such bots can influence national political opinion [1,2,3,4].

[1] shows that social spambots are different to traditional spambots, and are present on Twitter. The new social bots often evade detection by humans and state-of-the-art detection algorithms published in academic literature. The authors demonstrate that such bots evade detection through a series of analyses. In turn, the authors call for new methods to be developed that can successfully identify 'social spambots'.

[2] shows there is a recent (>2014) novel species of spambots specifically designed to mimic authentic

user behavior in order to evade state of the art bot identification routines.

Some researchers have shown that simple rules can be used to effectively separate authentic Twitter accounts from *traditional spambot* accounts using information gain calculations [3,5]. Related to this, researchers have also shown that not all simple rules are effective in the classification of Twitter spambots. One paper [3] showed that rules commonly promoted in popular media are often ineffective at identifying Twitter spambots while rules identified in the academic literature tended to be more robust [1,3].

2. Related Work

Cresci et al. [1,3] reports the creation of several datasets of fake and authentic Twitter accounts. In [3], fake accounts were purchased from third party services and researchers scraped the relevant account data. Authentic accounts were obtained by promoting the research on Twitter and through research advocacy groups, following up with individual account signups. Users were verified using CAPTCHA and evaluated by two sociologists.

[7] identified five simple rules, each with relatively high Information gain [3], that can be used in coordination to create classifier models. Here is Table 11 reprinted from [3] showing the information gain and rule data from [7] (emphasis added):

	feature description	evaluation metrics *I gain*	*Pcc*
Stringhini et al. [8]			
1	number of friends	0.263	0.030
2	number of tweets	0.621	0.289
3	content of tweets	0.444	0.740
4	URL ratio in tweets	0.401	0.353
5	$\frac{friends}{(followers^{*}2)}$	0.733	0.169
Yang et al. [2]			
1	age	0.539	0.436
2	bidirectional links ratio	0.905	0.875
3	avg. followers of friends	0.327	0.254
4	avg. tweets of friends	0.203	0.235
5	firends / med. foll. of friends	0.336	0.102
6	api ratio	0.544	0.635
7	api url ratio	0.058	0.113
8	api tweet similarity	0.460	0.748
9	following rate	0.355	0.214

Table 11: Evaluation of the single feature.

Figure 1: Table 11 from [3] reproduced, with emphasis added showing rules we implemented for information gain calculations.

[3] used rules proposed by [7] to create a variety of classifiers to identify traditional spambots.

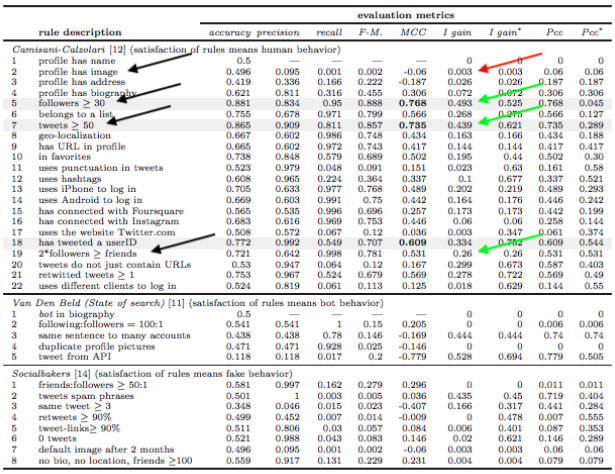
Here are metrics, reprinted from [3], with annotations added to emphasize that the Decision Tree and k-Nearest Neighbor classifiers were over 90% accurate when using only the five rules identified by [7]:

	algorithm	evaluation metrics *accuracy*	*precision*	*recall*	*F-M.*	*MCC*	*AUC*
Classifiers based on feature set by Yang et al. [2]							
RF	Random Forest	**0.991**	0.991	0.991	**0.991**	**0.983**	0.998
D	Decorate	**0.991**	0.988	**0.993**	**0.991**	**0.983**	0.998
J48	Decision Tree	0.990	0.991	0.989	0.990	0.980	0.997
AB	Adaptive Boosting	0.988	0.989	0.987	0.988	0.974	**0.999**
BN	Bayesian Network	0.976	**0.994**	0.958	0.976	0.936	0.997
kNN	k-Nearest Neighbors	0.966	0.966	0.966	0.966	0.932	0.983
LR	Logistic Regression	0.969	0.966	0.973	0.969	0.939	0.996
SVM	Support Vector Machine	0.989	0.985	**0.993**	0.989	0.976	0.989
Classifiers based on feature set by Stringhini et al. [8]							
RF	Random Forest	0.981	0.983	**0.979**	**0.981**	**0.961**	**0.995**
D	Decorate	**0.982**	**0.984**	**0.979**	**0.981**	**0.961**	0.993
J48	Decision Tree	0.979	**0.984**	0.974	0.979	0.953	0.985
AB	Adaptive Boosting	0.968	0.965	0.970	0.968	0.938	**0.995**
BN	Bayesian Network	0.953	0.953	0.953	0.953	0.907	0.985
kNN	k-Nearest Neighbors	0.954	0.961	0.946	0.953	0.907	0.974
LR	Logistic Regression	0.927	0.921	0.935	0.928	0.855	0.974
SVM	Support Vector Machine	0.959	0.967	0.950	0.958	0.917	0.958

Table 12: Performance comparison for 10-fold cross validation. Training set: *BAS*.

Figure 2: Table 12 from [3] reprinted with emphasis added to show two classifiers made using only the rules proposed by [7].

[3] went further by identifying many rules - some from the media, others from academia - and publishing the respective information gain. The table below is annotated to emphasize a few rules we calculate in our study (black arrows) as well as the related information gain (green is high gain, red is low gain).



	rule description	evaluation metrics								
		accuracy	*precision*	*recall*	*F-M.*	*MCC*	*I gain*	*I gain**	*Pcc*	*Pcc**
Camisani-Calzolari [12] (satisfaction of rules means human behavior)										
1	profile has name	0.5	—	—	—	—	0	0	0	0
2	profile has image	0.496	0.095	0.001	0.002	-0.06	0.003	0.003	0.06	0.06
3	profile has address	0.419	0.336	0.166	0.222	-0.187	0.026	0.026	0.187	0.187
4	profile has biography	0.621	0.811	0.316	0.455	0.306	0.072	0.072	0.306	0.306
5	followers ≥ 30	0.881	0.834	0.95	0.888	**0.768**	0.493	0.525	0.768	0.045
6	belongs to a list	0.755	0.678	0.971	0.799	0.566	0.268	0.275	0.566	0.127
7	tweets ≥ 50	0.865	0.909	0.811	0.857	**0.735**	0.439	0.621	0.735	0.289
8	geo-localization	0.667	0.602	0.986	0.748	0.434	0.163	0.166	0.434	0.188
9	has URL in profile	0.665	0.602	0.972	0.743	0.417	0.144	0.144	0.417	0.417
10	in favorites	0.738	0.848	0.579	0.689	0.502	0.195	0.44	0.502	0.30
11	uses punctuation in tweets	0.523	0.979	0.048	0.091	0.151	0.023	0.63	0.161	0.58
12	uses hashtags	0.608	0.965	0.224	0.364	0.337	0.1	0.677	0.337	0.521
13	uses iPhone to log in	0.705	0.633	0.977	0.768	0.489	0.202	0.219	0.489	0.293
14	uses Android to log in	0.669	0.603	0.991	0.75	0.442	0.164	0.176	0.446	0.242
15	has connected with Foursquare	0.565	0.535	0.996	0.696	0.257	0.173	0.173	0.442	0.199
16	has connected with Instagram	0.683	0.616	0.969	0.753	0.446	0.06	0.06	0.258	0.144
17	uses the website Twitter.com	0.508	0.572	0.067	0.12	0.036	0.003	0.347	0.061	0.374
18	has tweeted a userID	0.772	0.992	0.549	0.707	**0.609**	0.334	0.762	0.609	0.544
19	2*followers ≥ friends	0.721	0.642	0.998	0.781	0.531	0.26	0.26	0.531	0.531
20	tweets do not just contain URLs	0.53	0.947	0.064	0.12	0.167	0.299	0.673	0.587	0.403
21	retwitted tweets ≥ 1	0.753	0.967	0.524	0.679	0.569	0.278	0.722	0.569	0.49
22	uses different clients to log in	0.524	0.819	0.061	0.113	0.125	0.018	0.629	0.144	0.55
Van Den Beld (State of search) [11] (satisfaction of rules means bot behavior)										
1	*bot* in biography	0.5	—	—	—	—	0	0	0	0
2	following:followers = 100:1	0.541	0.541	1	0.15	0.205	0	0	0.006	0.006
3	same sentence to many accounts	0.438	0.438	0.78	0.146	-0.169	0.444	0.444	0.74	0.74
4	duplicate profile pictures	0.471	0.471	0.928	0.025	-0.146	0	0	0	0
5	tweet from API	0.118	0.118	0.017	0.2	-0.779	0.528	0.694	0.779	0.505
Socialbakers [14] (satisfaction of rules means fake behavior)										
1	friends:followers ≥ 50:1	0.581	0.997	0.162	0.279	0.296	0	0	0.011	0.011
2	tweets spam phrases	0.501	1	0.003	0.005	0.036	0.435	0.45	0.719	0.404
3	same tweet ≥ 3	0.348	0.046	0.015	0.023	-0.407	0.166	0.317	0.441	0.284
4	retweets ≥ 90%	0.499	0.452	0.007	0.014	-0.009	0	0.478	0.007	0.555
5	tweet-links≥ 90%	0.511	0.806	0.03	0.057	0.084	0.006	0.401	0.087	0.353
6	0 tweets	0.521	0.988	0.043	0.083	0.146	0.02	0.621	0.146	0.289
7	default image after 2 months	0.496	0.095	0.001	0.002	-0.06	0.003	0.003	0.06	0.06
8	no bio, no location, friends ≥100	0.559	0.917	0.131	0.229	0.231	0.004	0.004	0.079	0.079

Table 8: Rules evaluation.

Figure 3: Table 8 from [3] reprinted with emphasis added to show the rules we implemented and the published information gain values. Note, the dataset used with by [3] only included traditional spambots.

Importantly, all [3] analyses were done on traditional spambots rather than social spambots. The attributes emphasized by arrows are ones we have chosen to calculate for information gain in this paper.

[2] introduces a novel approach for identifying social spambots by creating, and comparing, the 'digital DNA' of Twitter account activity. Their technique involved researchers creating a strand of letters where each letter corresponds to a specific activity in a Twitter account (tweeting, retweeting, liking, etc). Researchers then compared strands similarities between accounts. They showed that bot activity was markedly different from non-bot activity, and that sub-classes of bots could be differentiated. The strand comparison algorithms used are commonly used in DNA and protein sequence analyses.

3. Data Set

3.1 Data source

The dataset used by Cresci in [1,2] was acquired from Github Bot Repository from Botometer Download datasets zip from Bot Repository.

3.2 Data files and attributes

The full dataset is approximately 465 MB zipped and is composed of user account and tweet data in CSV file formats. Each file contains a different user type. See table 1 for a summary of account information contained in the eight files we used for this study.

Number of accounts	**User type (per file)**
3,474	*Authentic*
991	*Social spambots #1*
3,457	*Social spambots #2*
464	*Social spambots #3*
1,000	*Traditional spambots #1*
100	*Traditional spambots #2*
433	*Traditional spambots #3*
1,128	*Traditional spambots #4*

Table 1: A summary of the number of accounts and user type for the data set provided by [1].

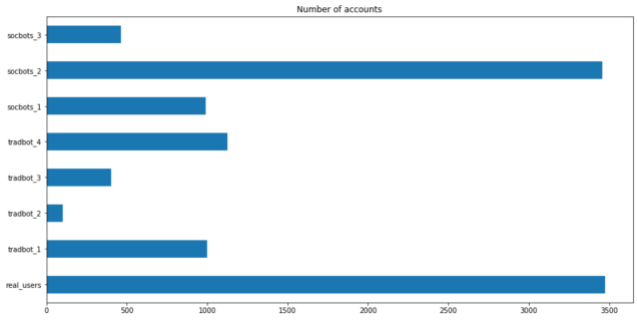


Figure 4: Bar chart showing our exploratory analysis which confirmed the number of accounts reported by [1].

The data set included information for user and tweet data.

The attributes for user data include ID, statuses_count, followers_count, friends_count, favourites_count, listed_count, default_profile, defalt_profile_image, geo_enabled, profile_use_background_image, utc_offset, is_translator, follow_request_sent, protected, verified, notifications, contributors_enabled, and following.

The attributes for tweet data include user_id, truncated, in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, geo, contributors, retweet_count, reply_count, favorite_count, favorited, retweed, possible_sensitive, num_hashtags, num_urls, and num_mentions.

Some of the data stored in these attributes has been deprecated with Twitter API upgrades and were filled with NaN values. See figure 4.

No longer supported (deprecated) attributes

Field	Type	Description
utc_offset	null	Value will be set to null. Still available via GET account/settings
time_zone	null	Value will be set to null. Still available via GET account/settings as tzinfo_name
lang	null	Value will be set to null. Still available via GET account/settings as language
geo_enabled	null	Value will be set to null. Still available via GET account/settings. This field must be true for the current user to attach geographic data when using POST statuses / update
following	null	Value will be set to null. Still available via GET friendships/lookup
follow_request_sent	null	Value will be set to null. Still available via GET friendships/lookup
has_extended_profile	null	**Deprecated**. Value will be set to null.
notifications	null	**Deprecated**. Value will be set to null.
profile_location	null	**Deprecated**. Value will be set to null.
contributors_enabled	null	**Deprecated**. Value will be set to null.
profile_image_url	null	**Deprecated**. Value will be set to null. NOTE: Profile images are only available using the profile_image_url_https field.
profile_background_color	null	**Deprecated**. Value will be set to null.
profile_background_image_url	null	**Deprecated**. Value will be set to null.
profile_background_image_url_https	null	**Deprecated**. Value will be set to null.
profile_background_tile	null	**Deprecated**. Value will be set to null.
profile_link_color	null	**Deprecated**. Value will be set to null.
profile_sidebar_border_color	null	**Deprecated**. Value will be set to null.
profile_sidebar_fill_color	null	**Deprecated**. Value will be set to null.
profile_text_color	null	**Deprecated**. Value will be set to null.
profile_use_background_image	null	**Deprecated**. Value will be set to null.
is_translator	null	**Deprecated**. Value will be set to null.

Figure 4: Attributes of the Twitter API that had been deprecated with the most recent API release (v 2.0). We chose rules that avoided using these attributes so the data could easily be included in automated spambot detection.

The social spambot data were collected using various methods such that each file of social bot data was of a similar type. See below for a summary of the kind of social bot present in each file.

File	Type
Traditional #1	*Training set of spammers used by Yang et al.*
Traditional #2	*Spammers of scam URLs*
Traditional #3	*Automated accounts spamming job offers*
Traditional #4	*Another group of automated accounts spamming job offers*
Social #1	*Retweeters of Italian political candidate*
Social #2	*Spammers of paid apps for mobile devices*
Social #3	*Spammers of products on sale at Amazon.com*

Table 2: A summary of the kind of bots in each subset of the data. This is reprinted from [3].

3.3 Data cleaning and integration

The data is organized by authentic versus bot account. As in [3], authentic account data are in separate files from bot account data. During our analysis we created a new attribute 'bot' and concatenated bot and non-bot account information so that the analysis could be completed.

In our specific case authentic data was concatenated with traditional bot #1 for the base analysis. Then in follow-up analyses, authentic data was concatenated with traditional bot #2, bot #3, traditional bot #4, social bot #1, social bot #2, and social bot #3 -- so that seven different sets of information gain calculations were generated.

4. Main Techniques Applied

4.1 Information Gain

First, we identified seven rules that could be easily used, given Twitter's API, to effectively (based on [3]) separate traditional Twitter spambots from authentic user activity.

The rules we chose are summarized below.

Rule
Following >30 accounts
Following >100 accounts
Has >30 followers
Has >100 followers
Friends/followers^2 < 0.05
Has authored >50 tweets
Has authored >1,000 tweets

Table 3: A summary of the rules we generated for information gain calculations. Rules were generally proposed by [7] and used in [3] to evaluate a different data set.

Then we integrated bot and authentic user account information to generate subsets of data for analysis.

Subsets of data
Authentic users + traditional spam bots #1
Authentic users + traditional spam bots #2
Authentic users + traditional spam bots #3
Authentic users + traditional spam bots #4
Authentic users + social spam bots #1
Authentic users + social spam bots #2
Authentic users + social spam bots #3

Table 4: A summary of the subsets of data we generated using the files present in our dataset.

We validated our information gain code by comparing our results for the subset including traditional spambot #1 and the results in [3] (which were only on traditional spambots found in

a different dataset). Once validated, we used our code to generate information gain results for the remaining subsets.

The results of all rules were binary, so some data processing was used to generate a sensible attribute for the friends/follower^2 ratio. In that specific case, the cumulative distribution function was generated for the ratio for each of the bot and non-bot populations individually. Then a threshold value was chosen as an estimate to maximize the difference between the two CDFs. In this way if a specific user account has a friends/follower^2 ratio less than the threshold, then it will maximize the effectiveness of the ratio as a differentiator. For our dataset, a threshold of 0.05 was used.

4.2 Random Forest Classification

We sought to use a modern technique which works well for classifying data that may not obviously show correlation and would not over predict the data when provided. Ensemble methods are useful in this situation or when there is significant diversity in the data as we saw in past research and in our analysis using the differing groups of bots in our information gain computations. This is, from set to set some rules worked for identification and in others they did not. This stochastic behavior is ideal for ensemble classification. To this we used a well-known machine learning algorithm which is becoming commonplace in data mining projects. The classifier we chose to use is an ensemble learning algorithm known as a Random Forest Classifier. This algorithm constructs a multitude of decision trees from subsets of the data and uses these for training. A majority vote is created on which tree works best and is reported [8]. An example of a simplified version of this classification model is shown in figure 5.

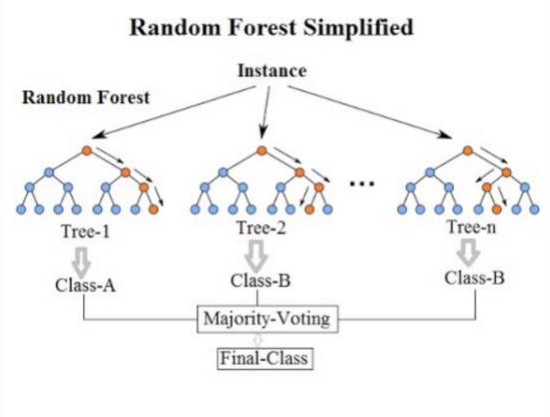


Figure 5: Random Forest Classification model simplified. Credit for image: Venkata Jagannath

For the Random Forest Classifier data was concatenated and marked as bot or not as was done for the information gain except thresholds were not needed. Data underwent preprocessing to transform classes from raw feature vectors into a representation that are more suitable for the downstream estimators for model classification. Four attributes which have been shown to have promising results with ensemble classifiers were chosen to train and test the model [3] which are shown in table 5.

Attributes	Reasons for success
Number of friends	*Spambots do not have thousands of friends*
Number of tweets	*Spambots have sent less than 20 tweets*
friends/followers^2	*Spambots have a high friends/followers^2 ratio value (i.e., lower ratio values imply legitimate users).*
Favorites count	*The number of Tweets this user has liked in the account's lifetime.*

Table 5: A summary of the attributes used for Random Forest Classification.

Each set of data sets tested were split into two sets. One containing 30% of the data which was

used for testing the model and the other 70% of the data was used for training. This was performed so we could ensure we were not performing both modeling and validation on the same fraction of data.

Subsets of data in Random Forest
Authentic users + traditional spam bots #1
Authentic users + traditional spam bots #2
Authentic users + traditional spam bots #4
Authentic users + social spam bots #1
Authentic users + social spam bots #3

Table 6: A summary of the subsets of data generated for Random Forest Classifier.

5 Tools

The group used Python and Jupyter notebook for code development. The Jupyter notebook is available on the shared github account. We used git and github for version control.

Pandas was used for data cleaning and wrangling due to its speed of processing large data volumes. In our particular dataset, some collections of tweets are 1 GB csv files when unzipped.

Matplotlib and Seaborn were used for visualization and Numpy for statistical analysis and data visualization.

Scikit-learn was used for building the random forest classifier and for testing the models ability to accurately classify our data using built in classification reports such as confusion matrices, recall, precision, support and f1-score.

Information gain functions were coded from scratch.

6 Key Results

6.1 Information Gain

First, we calculated the total information of each subset of data. See below for a summary of the results.

Subset of data	Info Dataset
Traditional #1	*0.7665*
Traditional #2	*0.1842*
Traditional #3	*0.4814*
Traditional #4	*0.8034*
Social #1	*0.7637*
Social #2	*0.9999*
Social #3	*0.5231*

Table 7: A summary of the total information in the subsets of data we used in our analyses.

Next, we calculated the information gain for the seven rules we selected, for each of the traditional spambot subsets of data. See tables 7-10 for summaries of the results.

Traditional Spambots #1

Rule	Info attr	Info gain
Following >30	*0.7618*	*0.0047*
Following >100	*0.7391*	*0.0274*
>30 followers	*0.7622*	*0.0043*
>100 followers	*0.7602*	*0.0064*
Friends/followers^2 > 0.05	*0.7271*	*0.0394*
Authored >50 tweets	*0.3942*	*0.3723*
Authored >1,000 tweets	*0.3980*	*0.3686*

Traditional Spambots #2

Rule	Info attr	Info gain
Following >30	*0.1833*	*0.0008*
Following >100	*0.1208*	*0.0633*
>30 followers	*0.1731*	*0.0110*
>100 followers	*0.1188*	*0.0653*
Friends/followers^2 > 0.05	*0.1798*	*0.0043*
Authored >50 tweets	*0.1835*	*0.0006*
Authored >1,000 tweets	*0.1191*	*0.0650*

Traditional Spambots #3

Rule	Info attr	Info gain
Following >30	*0.4199*	*0.0615*
Following >100	*0.3940*	*0.1074*
>30 followers	*0.4281*	*0.0533*
>100 followers	*0.3537*	*0.1277*
Friends/followers^2 > 0.05	*0.4327*	*0.0487*
Authored >50 tweets	*0.4178*	*0.0636*
Authored >1,000 tweets	*0.4713*	*0.0101*

Traditional Spambots #4

Rule	Info attr	Info gain
Following >30	*0.7988*	*0.0046*
Following >100	*0.7722*	*0.0312*
>30 followers	*0.7868*	*0.0166*
>100 followers	*0.7597*	*0.0438*
Friends/followers^2 > 0.05	*0.7750*	*0.0284*
Authored >50 tweets	*0.5017*	*0.3072*
Authored >1,000 tweets	*0.3733*	*0.4302*

Our results show that even among traditional spambots, bot identification depends strongly on the automated processes used to generate and utilize the bot accounts. For example, traditional spambots #1 and #4 had relatively large information gains for accounts that authored >50 or >1,000 tweets. But these rules were generally ineffective for the other traditional spambots. The accounts in Traditional Spambots #1 are the oldest of the larger dataset from [3], which were originally analyzed in Yang et al (see [3] for citation) in 2010 and traditional spambots #4 is a collection of accounts that promoted job advertisements.

Put another way, traditional spambots #2 and #3 author >50 or >1,000 tweets with similar frequency as authentic Twitter users.

Relatedly, traditional spambots #3 can be identified (i.e. have large information gains) using rules related to the number of followers and the number of accounts they follow. In contrast, such rules were ineffective for the other traditional spambot subsets of data.

We also calculated the information gain for the seven rules for the three social spambot datasets. See tables 11-13 for summaries.

Social Spambots #1

Rule	Info attr	Info gain
Following >30	*0.6063*	*0.1574*
Following >100	*0.6919*	*0.0718*
>30 followers	*0.7106*	*0.1245*
>100 followers	*0.7624*	*0.0531*
Friends/followers^2 > 0.05	*0.7624*	*0.0013*
Authored >50 tweets	*0.7291*	*0.0346*
Authored >1,000 tweets	*0.5044*	*0.2593*

Social Spambots #2

Rule	Info attr	Info gain
Following >30	*0.9998*	*0.0002*
Following >100	*0.3557*	*0.6443*
>30 followers	*0.2264*	*0.7736*
>100 followers	*0.3837*	*0.6163*
Friends/followers^2 > 0.05	*0.2714*	*0.7286*
Authored >50 tweets	*0.8016*	*0.1984*
Authored >1,000 tweets	*0.3746*	*0.6254*

Social Spambots #3

Rule	Info attr	Info gain
Following >30	*0.5224*	*0.0007*
Following >100	*0.5137*	*0.0094*
>30 followers	*0.5211*	*0.0019*
>100 followers	*0.5096*	*0.0135*
Friends/followers^2 > 0.05	*0.5187*	*0.0044*
Authored >50 tweets	*0.5216*	*0.0014*
Authored >1,000 tweets	*0.5129*	*0.0102*

Our results show that the accounts in social spambots #3 would be very difficult to differentiate for authentic Twitter users if a classifier was limited to just these seven rules (i.e. the information gain is quite small for each attribute evaluated). These accounts were used to promote products on sale at Amazon.com.

Conversely, many of the rules (6 out of 7) prove effective for the accounts in social spambots #2. These accounts were used to promote paid apps for mobile devices. And only two rules (>30 followers and authored >1,000 tweets) were effective in identifying the accounts in social spambots #1. These accounts were used to retweet content published by a prominent Italian political candidate and are a highly studied area in research. This social bot group is hard to identify and has serious repercussions when left unidentified. The bot group is difficult to identify as our study and others have reported and are in need of further research.

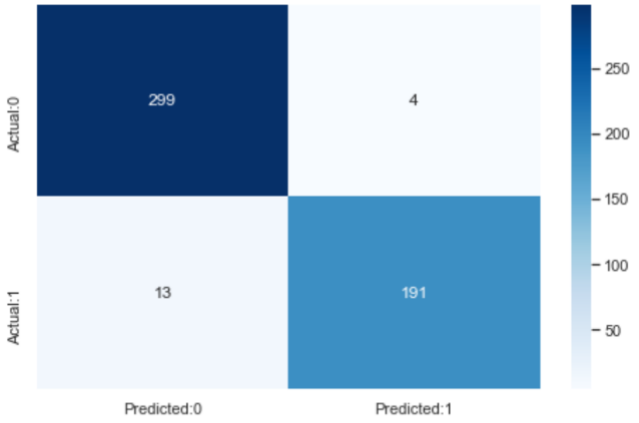
6.2 Random Forest Classifier

The Random Forest Classification using the four attributes (table 5) performed better than expected with our test data reporting accuracy scores ranging from 0.959 to 0.997 (Table 6).

In figure 5, we present the confusion matrix and the classification report for the Authentic users + traditional spam bots #1 data set. Briefly, the confusion matrix can be interpreted as:

- ***True Negative:*** *number of real users determined by the classifier to be real users*
- ***True Positive:*** *number of bots determined by the classifier to be bots*
- ***False Positive:*** *number real users determined by the classifier to be bots*
- ***False Negative:*** *number bots recognized by the classifier to be real users*

With these rules in mind, we can see from the confusion matrix that of the 299 real users only 4 were misclassified as bots and of the 191 bots only 13 were misclassified as real users.



Classification Report: =

	precision	recall	f1-score	support
0	0.96	0.99	0.97	303
1	0.98	0.94	0.96	204
accuracy			0.97	507
macro avg	0.97	0.96	0.96	507
weighted avg	0.97	0.97	0.97	507

Figure 5: Confusion matrix and classification report for Random Forest Classifier using Authentic users + traditional spam bots #1.

Dataset Classified	Accuracy Scores
Authentic users + traditional spam bots #1	***RF Test Score:*** *0.978* ***RF Train Score:*** *0.992*
Authentic users + traditional spam bots #2	***RF Test Score:*** *0.994* ***RF Train Score:*** *1.0*
Authentic users + traditional spam bots #4	***RF Test Score:*** *0.997* ***RF Train Score:*** *1.0*
Authentic users + social spam bots #1	***RF Test Score:*** *0.959* ***RF Train Score:*** *0.981*
Authentic users + social spam bots #3	***RF Test Score:*** *0.973* ***RF Train Score:*** *0.986*

Table 14: The attributes used for Random Forest Classifier and associated test and training accuracy scores.

7 Applications

Our results provide key insights into spambot behavior on Twitter. First, we confirm there is a diverse ecosystem of spambot activity and that not all spambots are the same. Even with our analysis - which only uses a few thousand user accounts - we show that it may be possible to classify less sophisticated spam bots using a simple model built with only a few, but effective, rules.

Second, we also provide information gain calculations that are consistent with the claim that some classes of social spambot are quite good at evading detection (social spambot subset #3, for example).

Third, our results suggest that a set of rules may be effective at identifying a certain class of spambots while a different set of rules is more effective with a different class of spambot. This insight provides evidence that it might be useful for a company like Twitter to have multiple classifiers, each targeting known classes of spambot.

This work led us to discuss the possibility of Twitter choosing, intentionally, to avoid deleting known bots. If a bot account is deleted, then it provides some feedback to bot creators about the classification models Twitter is using to identify spam. So instead of deleting accounts, Twitter might choose to discreetly decrease the messaging amplification of spam accounts. In this way Twitter engineers can reduce the social impact of spam while making it harder for spam creators to know when they have been detected. Such an approach has the downside that Twitter would be supporting infrastructure that maintains an increasing amount of (less effective) spam, but allow them to control what is actually happening on their platforms.

As can be seen, information gain can be useful for a select set of bots but the stochastic nature seen in new social bots is quickly making these techniques obsolete and more aggressive identification is required. Here we show, as other academic researchers have, that ensemble methods could be one solution to this problem.

The five datasets analyzed via the Random Forest Classifier all showed a high level of accuracy and ability to differentiate bots from real users regardless of their type. Furthermore, the set of attributes selected for modeling the data was something we found to be important. Continued research is needed to combat the ever-evolving social media bots and ensemble classification methods (as has been shown here) paired with algorithms for attribute selection could be one piece to that puzzle.

REFERENCES

1 Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2017. The Paradigm-Shift of Social Spambots. *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion* (2017). DOI:http://dx.doi.org/10.1145/3041021.3055135

2 Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, Maurizio Tesconi, "Social Fingerprinting: Detection of Spambot Groups Through DNA-Inspired Behavioral Modeling," in IEEE Transactions on Dependable and Secure Computing, vol. 15, no. 4, pp. 561-576, 1 July-Aug. 2018, doi: 10.1109/TDSC.2017.2681672.

3 Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, Maurizio Tesconi, Fame for sale: Efficient detection of fake Twitter followers,Decision Support Systems, Volume 80, 2015, Pages 56-71, ISSN 0167-9236, https://doi.org/10.1016/j.dss.2015.09.003.

4 Stefano Cresci. 2020. A decade of social bot detection. *Communications of the ACM* 63, 10 (2020), 72–83. DOI:http://dx.doi.org/10.1145/3409116

5 Besko and Carley. 2020. TBot-Match: Social Bot Detection with

6 Zhen Guo, Jin-Hee Cho, Ing-Ray Chen, Srijan Sengupta, Michin Hong, and Tanushree Mitra. 2020. Online Social Deception and Its Countermeasures: A Survey. *IEEE Access* 9 (December 2020), 1770–1806. DOI:http://dx.doi.org/10.1109/access.2020.3047337

7 G. Stringhini, C. Kruegel, G. Vigna, Detecting spammers on social networks, in: 26th Annual Computer Security Applications Conference, ACSAC '10, ACM, 2010, pp. 1–9.

8 Jiawei Han, Micheline Kamber, and Jian Pei. 2012. *Data mining: concepts and techniques*, Amsterdam: Elsevier.