Twitter Bots Detection

Xinchun Li, Wei Cui

1. Question and Motivation

In the past several years, the bots issue on social media comes into people's eyes and leads to some critical problems [1,2,3]. Both the quantity and the harmful effect of bots motivated an increasing number of researchers to find efficient and effective approaches for solving this problem, with a goal to target bots and intervene in their malicious behaviors at the earliest time. In our research, we are specifically focusing on detecting bots on the Twitter platform.

We would like to explore several relevant questions to better understand online social behaviors: What are some telling features to differentiate bots from humans? How different are the spam groups serving different purposes? Do they have any common traits? What features are more generalizable, and what are more specific? What are different ways to predict bots, and their pros and cons? How the predictive accuracy changes if we change the data volume or predict across different domains? What are some vocabularies, nouns and adverbs bot tweets more more likely to include? And what are some ideas or insights we can draw from all these to better mitigate or prevent bot behaviors in online communities?

These are central questions that we have been thinking about a lot throughout our research, and is our primary motivation to work on this project. They are closely relevant to our own online footprint, and are at the same time critically important in today's era, when social media and internet has seamlessly pervaded people's daily life.

2. Past Work and Solutions

Based on previous studies [3,4,5], the definition of bots in our research is the accounts that post automatically produced contents and make malicious effects on social media platforms or populations. Two topics that have been frequently discussed in the related studies are listed below.

2.1. Tweet-level v.s. Account-level Detection

The account-level detection has a history that can be traced back to the spam detection. Studies in this field are focusing on the time series of bots' behavior and how to leverage it to do the detection. As the novel techniques have been used in bots production in recent years, however, the difficulty of using the previous behavior to do the classification between bots and human accounts has increased significantly. Because the bots producers can always find out the algorithm behind the detector and change their behavior to prevent being detected, the game between these two sides may never end. However, with the novel algorithms, like genetic algorithm[6], have been involved into the detection system, the account-level detection still has the advantage of using the history of the account.

Compared to most of the research that focuses on gathering the historical tweets of an account, including the manually labeling step, or involving the features that need calculation beforehand, some studies are focusing on using the tweet-level features to make the real-time prediction of bots. As the time sequence data may not be available in this kind of analysis, the text information has been paid more attention to. [7] focused on the features that relate to tweets, which include profile data (e.g. length of the profile name), sentiment lexicons, n-grams, and text analysis features (e.g. the number of spam words) in their research. This makes it possible for the detection to be made down to the tweet-level, so that the detectors do not have to gather the historical data of an account.

Besides the statistical features, the text content also contains rich information. [8] leveraged the bag-of-words and Dynamic Markov compression to detect the bots on the tweet-level. Other studies [9, 10] also included URLs that may point directions to malicious websites as a critical feature for the bots detection.

2.2. Feature Stability

As the bots issue has been spread drastically, developing different detection systems for each social media platform can be costly. Researches start to focus on the stability of the features across various platforms and timelines. [11] has identified 14 generic statistical features that can be applied for both Twitter and Facebook platform. These features can be classified into four categories: interaction-driven, post-driven, URL-driven, and tag-driven.

3. Dataset Summary and Collection

Most of our data has been collected from the [1]MIB (My Information Bubble) Project, funded by Registro.it and hosted at IIT (Institute of Informatics and Telematics) of the Italian National Research Council (CNR). The data is comprised of genuine and spambot Twitter accounts, annotated by CrowdFlower contributors. There are 8 subsets in total in this dataset which corresponding with different types of bots and 1 genuine account subset (Table 1), though we have used 6 out of 8 subsets of bots and 1 subset of genuine accounts. Also, the other part of our data is the Russian Trolls Tweets Dataset which is collected from Kaggle. Table 1 shows us a summary of dataset information.

Datasets Name	Objectives	Collection
Social Spambots 1	Retweet an Italian political candidate	Collected from MIB (My Information Bubble) project, funded by Registro.it and hosted at IIT (Institute of Informatics and Telematics) of the Italian National Research Council (CNR) to identify false information online.
Social Spambots 2	Advertise paid app	
Social Spambots 3	Advertise sales on amazon. com	
Traditional Spambots 1	Post malicious URLs	

[1]MIB Project: http://mib.projects.iit.cnr.it/index.html

Traditional Spambots 4	Spam job offers	
Fake Followers	Bought from Twitter online market	
Genuine Accounts	Voluntary, have been verified	
Russian Trolls	Interfere U.S. elections, part of the Twitter suspended list	Collected from Kaggle.com.

Table 1: Dataset objectives and sources

These datasets are all that have ever been used in our project, although later in the semester, we did not include Social Spambots 1 and Traditional Spambots 4 in our modelling and analysis, as Social Spambots 1 tweets are mainly in Italian, which is not convenient for text processing in which our language base is English[2], while Traditional Spambots 4 does not have tweets available. They were, however, used in our baseline models. As at that time, we would like to use data of different years and see if there's a time influence on our prediction accuracy.

Our data is classified into different groups serving different purposes. This is in fact helpful for our project, as we would like to analyze spams of various kinds and uncover broad implications and generalizable patterns.

4. Baseline Methods

For the baseline models, we selected all the numerical features existent in our dataset that supposedly would influence the botness of the accounts. We have dropped the features that have more than 70 percent of missing values as well as filled other missing values with the median or mode of the available values of according features depending on the value distribution and what makes sense intuitively.

We have so far merged the accounts and tweets files respectively for genuine accounts, social spambots #1, social spambots #2, social spambots #3 and traditional spambots #4 groups. Each account would have one or more tweets. And we cross used genuine accounts and one spam accounts at a time with different classifiers to test their performance. This is on the one hand, to examine whether a certain classifier would yield consistent performance over different combinations of data so that we get to know and think about the robustness of each classifier and how to interpret and improve that. On the other hand, it provides some insight on whether or not there's a time influence in Twitter users' online behavior, spam or not, as the data is gathered across different years, and it is possible that spam or genuine accounts have developed or shown different online behaviors and how they interact with other users.

[2]We use module 'spacy.lang.en' https://spacy.io/usage/adding-languages#101

For each combinations of data, we first assigned 1 as a new column “label” to spam users and 0 as a new column “label” to genuine users. Then we concatenated genuine and spam accounts and shuffled them. Then we used “train_test_split” in sklearn package to obtain the train and test set (70% and 30% respectively): X_train, X_test, y_train, y_test. After that, scaling is applied on X_train and y_train.

We trained four classifiers on each combination of data, Logistic Regression Classifier, Decision Tree Classifier, Naive Bayes Classifier and Support Vector Machine Classifier with mild parameter tuning. The evaluation metrics for the test set being reported are Accuracy, Recall, Precision, F1 Score and AUC Score. We set five cross validations (cv=5) for each process to avoid arbitrariness, so each metric gets an array of five results. The below shows the mean accuracy for all the classifiers on each combination of data.

Mean Accuracy	Genuine + Spam1	Genuine + Spam2	Genuine + Spam3	Genuine + Spam4
Logistic Regression	0.93	0.93	0.97	0.98
Decision Tree	0.98	0.99	0.98	0.99
Naive Bayes	0.92	0.93	0.91	0.97
Support Vector Machine	0.64	0.68	0.66	0.66

Table 2: Baseline Models Accuracy Performance

From Table 2, we can see that the Logistic Regression classifier and Decision Tree classifier yield a relatively consistent performance over different combinations of data, and all the scores tend to remain pretty high. The Naive Bayes classifier yields reasonably good performance as well, and each cross validated group yields similar performance. Comparatively, the Support Vector Machine classifier shows volatile performance. All the SVM scores are relatively low and vary a lot for each combination of data and within it each cross validated group. As the features have been scaled to achieve normality, and there’s a reasonable level of parameter tuning, the poor performance of Support Vector Machine is not related to that.

5. Proposed Methods

Apart from using the existing features available in our dataset to build prediction models, we have been trying to identify and employ other features that make sense for our prediction and enrich our understanding of online bot behaviors.

5.1. Three Type of Features

Our proposed methods can be classified into three categories: simple numeric features based prediction, text based prediction and network based prediction. These three features are applied

into separate models, in order for us to understand which features yield better performance, which features yield more stable performance and which features make more sense and better help us uncover helpful insights.

5.2. Domain Adaptation

For each category of features, in order to explore model sustainability and feature generalizability, we have further built prediction models cross-domain, meaning that the bot group we include the training set is different from the bot group being tested on. This is done in order to compare the performance of cross-domain and within domain.

We would like to figure out the decision boundary and scope of each spam group via the performance of different combinations of train and test set. And it's likely that cross-domain prediction does not yield very good accuracy performance, for that each spam group serves a different purpose, and the features learned in the training set are not going to achieve high accuracy when they are being applied in the test set. Thus, on top of the cross-domain prediction, we have also experimented on adding different portions of test set to the train set, and comparing the test set accuracy. Intuitively, doing this would increase test set accuracy, as part of the test set features are also being learned in the models. We tried this idea out in different combinations of train and test set (we use five spam groups and one genuine group) over four machine learning methods: support vector machine, random forest, gradient boosting descent and logistic regression, to find out how the test set accuracy changes, the changing rate and degree.

5.3. Tweet level vs. Account level

We have explored both account level and tweet level on simple numeric features-based prediction and text-based prediction. On the account level, we combine all tweets of one user together and predict spam users, while on the tweet level, we look at tweets individually and predict spam tweets. The idea is to compare both performance and obtain helpful insights on predicting bot accounts vs. predicting bot tweets via slightly different structuring and sampling of data, while keeping the actual features and methods the same between the two.

5.4. Sampling-based without repetition

Note that, in order to avoid cheating (same data being used in the train and test set at the same time), all the train and test data have been sampled without repetition, so there is no repetitive data points used in the train and test set at one time.

6. Results

6.1. Text-based Method

6.1.1. Tweet level

On the tweet level, we cleaned the textual content, gave feature weights to bag-of-words, and vectorized text for preprocessing.

For each train set, there are 100,000 genuine tweets and 100,000 spam tweets from one group (such as social spambots 2); for each test set, there are 100,000 non-repetitive genuine tweets and 100,000 spam tweets from another group (such as social spambots 3). We have 5 * 4 cross-domain (train and test set spams are from different groups) combinations from five spam groups (social spambots 2, social spambots 3, traditional spambots 1, fake followers, Russian Troll) and 5 * 1 within-domain combinations from the same five spam groups (train and test set spams are from the same group).

We then added different portions of test set data points to the train set, 10,000 (1/20), 20,000 (1/10), 30,000 (3/20) and 40,000 (⅕) respectively. This has been repeated for all the 25 combinations above. Since the traditional spambots 1 dataset and fake followers dataset are smaller, there's no enough data point for use given when we add maximum number of non-repetitive test data to the train set. Therefore, we have used smaller data points for these two groups: 50,000 genuine tweets + 50,000 traditional spambots 1 as the train set without adding test data, 50,000 genuine tweets + 50,000 non-repetitive traditional spambots 1 as the test set; 70,000 genuine tweets + 70,000 fake followers tweets as the train set without adding test data, 70,000 genuine tweets + 70,000 non-repetitive fake followers tweets as the test set. And the added test data points are kept the same, 10,000, 20,000, 30,000 and 40,000 respectively.

For SVM models, we set "gamma='auto', max_iter=100, random_state=0"; For Random Forest models, we set "n_estimators=100, max_depth=20, criterion='entropy', random_state=0"; For Gradient Boosting models, we set "n_estimators=100, learning_rate=1.0, max_depth=5, random_state=0"; For Logistic Regression models, we set "penalty="l1",random_state=0".

The predictive accuracy for text-based methods is going up and down. It varies a lot between different methods, and between different train and test sets. But the general trend is that accuracy improves as we add portions of test data to the train set. When we first add a small amount of data, accuracy could be going up or down in a slightly drastic and unpredictable way. But as we further add more, accuracy gets stabilized, and the marginal effect becomes smaller. Besides, given that texts are diverse and subjective, it's hard to achieve very high accuracy consistently, such as over 0.85. There's not much room to improve if we get a fairly good accuracy result without adding any test data to the train set.

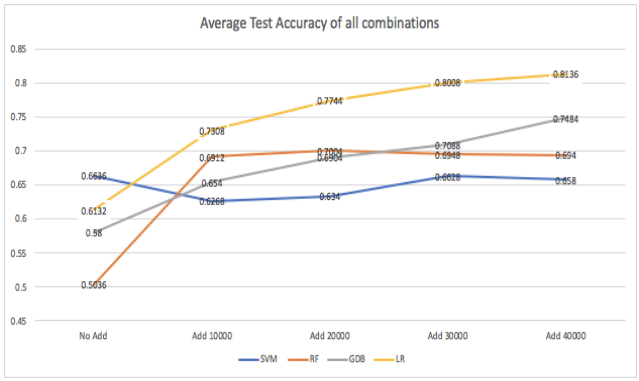


Chart 1: Mean Accuracy of different train and test sets

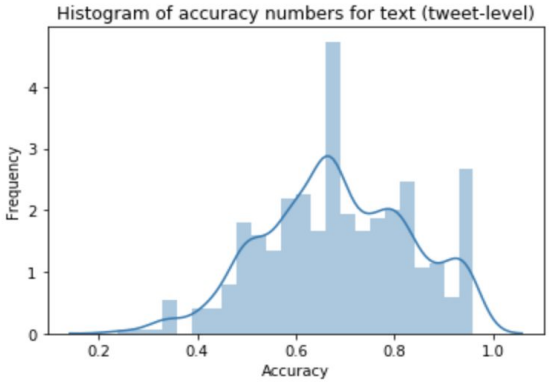
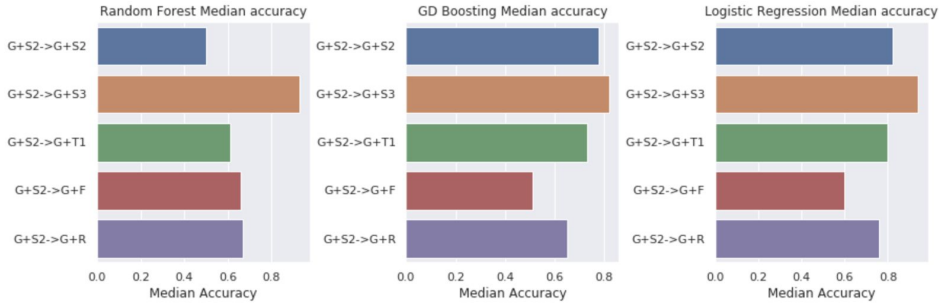


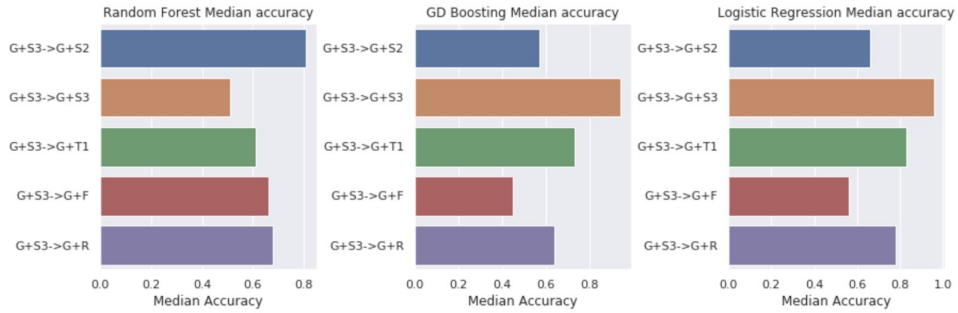
Chart 2: Histogram of textual accuracy results

SVM performance is unstable. For example, the accuracy of the same train and test set could go from 0.77 to 0.33, or from 0.24 to 0.74 when only a certain portion of test data are added to the train set. Although it might be that the new data is re-organized and re-classified when they are put into the model again, the internal functions work a bit differently, or that the randomly sampled data has some arbitrariness involved in itself, and that the messiness of text creates noise to a larger or less degree, overall SVM is not a very reliable method for our prediction and analysis, and the same could be said about other SVM models with different feature input (see in later sections).

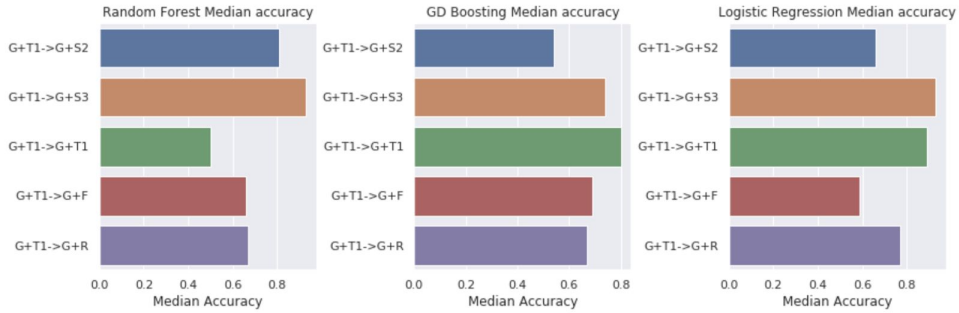
For the Random Forest, Gradient Boosting Descent and Logistic Regression methods, the below chart shows for each, the median accuracy of different train and test set combinations without adding any test data and with adding different amount of test data.



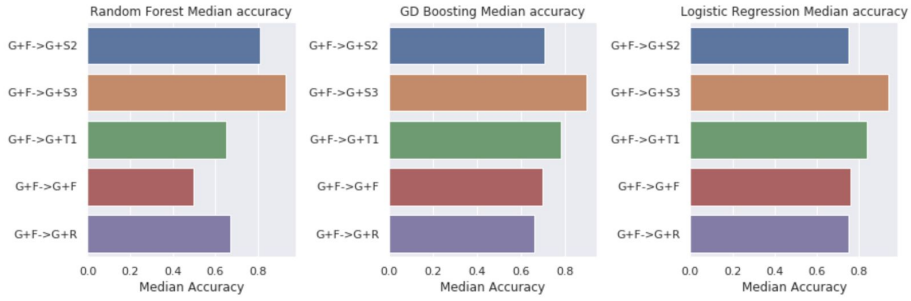
Using Social Spambots 2 in the train set to predict itself and other spam groups, we found that to predict Social Spambots 3 actually yields the highest performance, and to predict Fake Followers yields the lowest. It's a guess that Social Spambots 2 has wider vocabularies and scope than Social Spambots 3 does. And Social Spambots 2 has either narrower vocabularies and scope than Fake Followers does, or that it is not relevant contextually to Fake Followers.



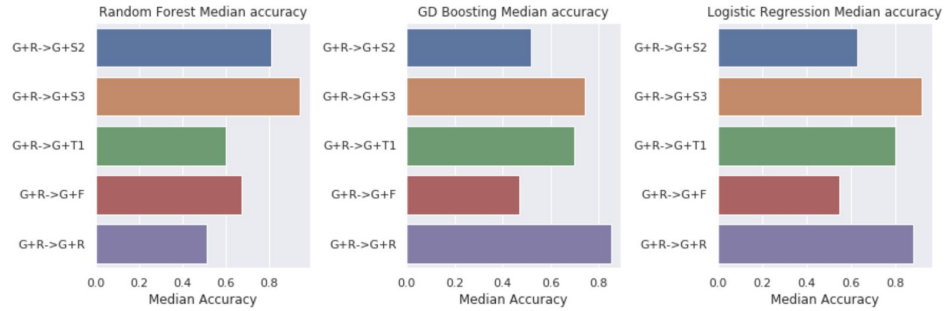
Using Social Spambots 3 in the train set to predict itself and other spam groups, we found that to predict itself yields the highest performance, and to predict Fake Followers yields the lowest. It's a guess that Social Spambots 3 has either narrower vocabularies and scope than Fake Followers does, or that it is not relevant contextually to Fake Followers.



Using Traditional Spambots 1 in the train set to predict itself and other spam groups, we found that to predict Social Spambots 3 yields the highest performance, and to predict Fake Followers yields the lowest. It's a guess that Traditional Spambots 1 has wider vocabularies and scope than Social Spambots 3 does. And Traditional Spambots 1 has either narrower vocabularies and scope than Fake Followers does, or that it is not relevant contextually to Fake Followers.



Using Fake Followers in the train set to predict itself and other spam groups, we found that to predict Social Spambots 3 yields the highest performance, and to predict Fake Followers and Russian Trolls yields the lowest. It's a guess that Fake Followers has wider vocabularies and scope than Social Spambots 3 does. And Fake Followers has either narrower vocabularies and scope than Fake Followers and Russian Trolls do, or that it is not relevant contextually to these two groups.



Using Russian Trolls in the train set to predict itself and other spam groups, we found that to predict itself and Social Spambots 3 yield the highest performance, and to predict Fake Followers yields the lowest. It's a guess that Russian Trolls has wider vocabularies and scope than Social Spambots 3 does. And Russian Trolls has either narrower vocabularies and scope than Fake Followers does, or that it is not relevant contextually to Fake Followers.

Let's look into the top word features generated by different train models. Let's take logistic regression for example. We found that textually, each spam group shows very different top word features. Chart 3 shows the top 20 word Features and their coefficients for Fake Followers and Russian Trolls respectively, as examples.

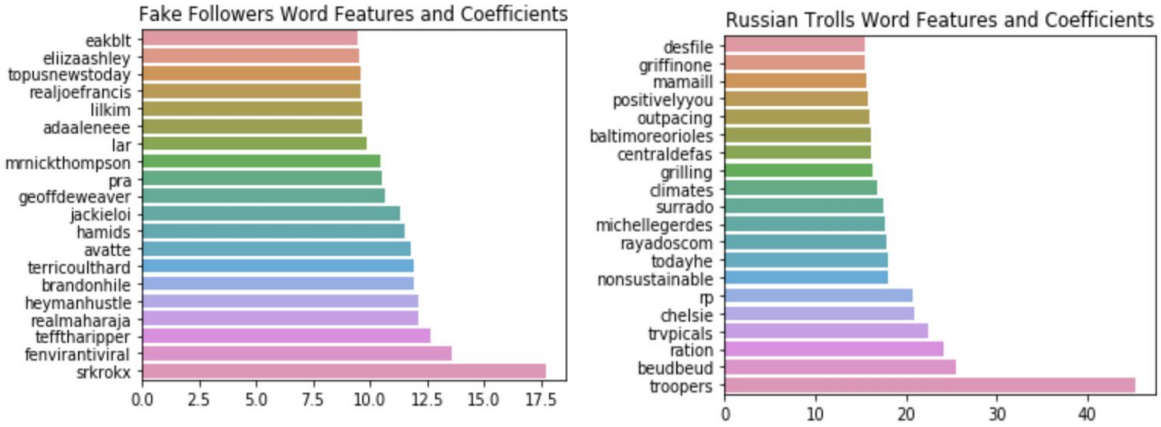


Chart 3: Top 20 word features for Fake Followers and Russian Trolls

Social Spambots 2 has top features like 'achieve', 'ff', 'vip', 'sarton', 'niceeee', 'dyer', 'proverb','plato', 'best', 'hiiiii', 'blakepiffin', 'achievements', 'machiavelli', 'biggboymanagement', 'followthemap','sincerebo', 'hiiii', 'dizzypsychic', 'cccoooolllll', etc. Some words make sense, some do not. Some have repetitive letters in it that look like exaggeration or nonsense. But the words are fairly diverse, and there're multiple topics being covered.

Social Spambots 3 has top features like 'confucius', 'conceit', 'holster', 'churchill', 'toupee', 'warren', 'healthtip', 'naomi', 'gretzky', 'arthur', 'tools', 'starr', 'firms', 'john', 'collegechat', 'jessica', 'overheard', 'rogers', 'puckett', 'overview', 'jean', 'marketing', etc. These words mostly make

sense, and they look a lot more neat. There's little typo or intended repetition of letters in words. It looks almost like words from normal human texts.

Traditional Spambots 1 has top features like 'seu', 'segue', 'venture', 'indica', 'tem', 'silver', 'sua', 'wwwpickyourpipecom', 'itus', 'shawnuometen', 'wwwwalkerkimberlyws', 'echibot', 'simplyhwn', 'vai', 'shanty', 'williamroth', 'ae', 'nternetmrktng', 'realinsane', 'startups', 'willareynolds', 'sxeo', 'serra', 'followers', 'jxe', 'buddaco', 'eu', 'mobsterworld', 'avrilxxxx', 'muito', 'powerofnowawareness', 'maxfana', 'adinahald', 'mj', 'ajsolis', 'quem', 'buck', 'veriiinn', 'vocxea', 'ueueue', 'xox', 'venturehype', 'dianaricketthomebiz', 'terrymslobodian', etc. These words are confusing. Most of them are never heard of or known any meaning of. Quite some just look like some random letters mixed together. We do not have a clear idea of what these words represent or any specific topic they are about.

Fake Followers has top features like 'bestselling', 'billallam', 'shrugged', 'honeymag', 'nf', 'rageofbahamut', 'agora', 'julianperretta', 'minotti', 'muskkey', 'meu', 'com', 'lookbooknu', 'aqui', 'benuptown', 'biomazinghcg','tem', 'tporter', 'agneskim', 'joanneleeeee', 'pmsoo', 'ugolord', 'crayoliiita', 'thanksgiving', 'mademoiselleffj', 'veracruz','cwyjc', 'zaynahlareb', 'inesjunqueira', 'youtube', 'sarahteuck', 'socialmboosthq', 'starforcehh', 'vdeo', 'rwanda', 'syrouche', 'diplo', 'lekker', 'dellytot', 'highkid', etc. These words are similar to the description of Traditional Spambots 1 word features.

Russian Troll Tweets has top features like 'thingsyoucantignore', 'alternativeacronyminterpretations','rejecteddebatetopics', 'dnc', 'hillary', 'maga', 'potus', 'oscarhasnocolor', 'beforeitsnews', 'christmasaftermath','hillaryclinton', 'conservatexian', 'survivalguidetothanksgiving','merkelmussbleiben', 'reallifemagicspells', 'todolistbeforechristmas', 'nineoh', 'rt', 'clinton', 'trumps','realdonaldtrump', 'blicqer', 'trump', etc. These words are easily observed to be politically related, mostly about Trump-Clinton Presidential Battle. The topic it relates is pretty specific. And the words are actually quite neat, unlike those of Traditional Spambots 1 and Fake Followers.

All of the above shows that, if a spam group has tweets that are more spam-like, more noisy, talk more nonsense and cover wider topics or no topic at all (ultimately abstract, just random letters here and there), it's going to make a wider decision boundary, and likely to predict well on other spam groups that are less spam-like; On the contrary, if a spam group has tweets that are more neat, more like human languages and are more specific in terms of topics or themes, it's going to make a narrower decision boundary, and likely to predict poorly on other spam groups that are more spam-like.

6.1.2. Account level

On the account level, we have gone through the same preprocessing steps and used bag-of-words features. As some of the datasets (Social Spambots 2 & Traditional Spambots 1) have more data points, we randomly selected 500 points in each dataset to make the balance. In

the across-domain part, we added 10(1/50), 20(1/25), 30(3/50) and 40(2/25) points from the test set into the training set in each step, respectively.

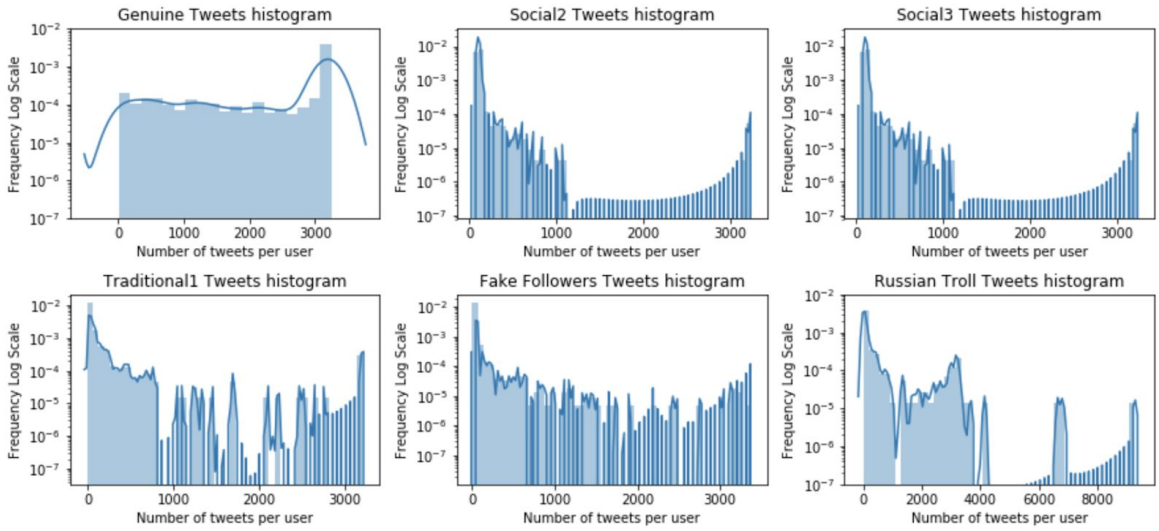


Chart 4: Numer of tweets per user histograms

Chart 4 shows the number of tweets per user for all groups. The histogram is log-scaled. As we can see, genuine tweets have a relatively uniform distribution, and even the higher end (3000 plus tweets per user) has more frequency. As the genuine users are the ones that volunteer to participate in the MIB project (refer to 3.Dataset Summary and Collection section) and have gone through machine verification, they are likely to be active users who tweeted a lot. And Social Spambots 2, Social Spambots 3, Traditional Spambots 1, Fake Followers and Russian Troll Tweets are clearly right skewed, to larger or lesser degree. Social Spambots 2 and Social Spambots 3 have almost the same distribution shape, which is interesting. And Russian Toll has a larger number of tweets per user than all the other groups do.

The performance of account level prediction on tweets is overall fairly good, better than that of tweet level prediction. Chart 5 shows the histogram of the accuracy of account level prediction on tweets with four methods being applied, without and with adding portions of test data to the train set. Accuracy numbers between 0.8 and 1 are fairly common, and almost all accuracy numbers are above 0.5.

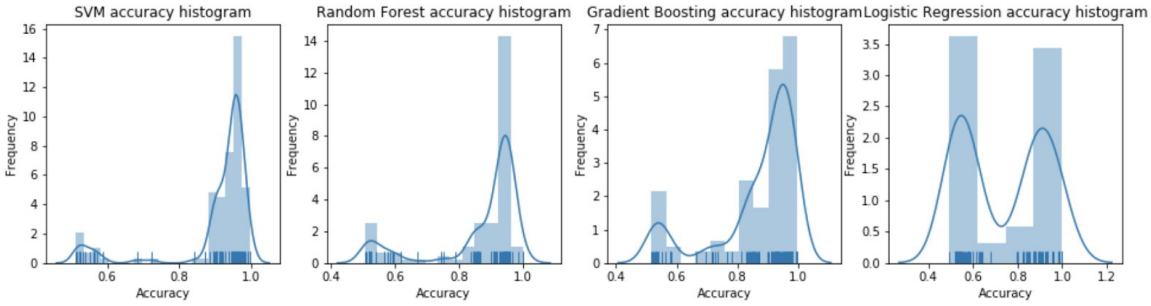


Chart 5: Textual accuracy histograms

All tweets from one user has certain styles and characteristics that can be better and more comprehensively learned if they are gathered together, while the random sampling of tweets is more scattered and does not take advantage of the consistency of characteristics for one user. Besides, as we know from above, different groups have different distributions in terms of the number of tweets per account. If we predict on account level, users that have more tweets have been given according weights, so do users that have fewer tweets, while predicting on tweet level gives every user equal weights, and that could lead to bias and inaccuracy. That's possibly the reasons we could use to explanation the discrepancy in accuracy results between account-level and tweet-level prediction.

6.2. Simple numeric features-based Method

6.2.1. Tweet level

On the tweet level, we have used some user account level features and tweet level features including 'statuses_count', 'followers_count', 'friends_count', 'favourites_count', 'listed_count', 'retweet_count', 'reply_count', 'favorite_count', 'num_hashtags', 'num_urls', 'num_mentions', etc. The assumption is that both user account level features and tweet level features make a difference in determining whether a tweet is spam or not, and the performance is expected to be better than the one obtained if we only use tweet level features or only user account level features.

For each train set, there are 100,000 genuine tweets and 100,000 spam tweets from one group (such as social spambots 2); for each test set, there are 100,000 non-repetitive genuine tweets and 100,000 spam tweets from another group (such as social spambots 3). We have 4 * 3 cross-domain (train and test set spams are from different groups) combinations from four spam groups (social spambots 2, social spambots 3, traditional spambots 1, fake followers) and 4 * 1 within-domain combinations from the same four spam groups (train and test set spams are from the same group). Since Russian Troll Tweets are obtained from a different source from other spam tweets, the numeric features are not as complete. Thus Russian Troll dataset is not used in here.

We then added different portions of test set data points to the train set, 10,000 (1/20), 20,000 (1/10), 30,000 (3/20) and 40,000 (⅕) respectively. This has been repeated for all the 20 combinations above. Since the traditional spambots 1 dataset and fake followers dataset are smaller, there's no enough data point for use given when we add maximum number of non-repetitive test data to the train set. Therefore, we have used smaller data points for these two groups: 50,000 genuine tweets + 50,000 traditional spambots 1 as the train set without adding test data, 50,000 genuine tweets + 50,000 non-repetitive traditional spambots 1 as the test set; 70,000 genuine tweets + 70,000 fake followers tweets as the train set without adding test data, 70,000 genuine tweets + 70,000 non-repetitive fake followers tweets as the test set. And the added test data points are kept the same, 10,000, 20,000, 30,000 and 40,000 respectively.

For SVM models, we set “gamma='auto', max_iter=100, random_state=0”; For Random Forest models, we set “max_depth=5, criterion='entropy', random_state=0”; For Gradient Boosting models, we set “n_estimators=100, learning_rate=1.0, max_depth=3, random_state=0”; For Logistic Regression models, we set “penalty="l1",random_state=0”.

The results are interesting to look at. First, the SVM method performance is rather volatile, with accuracy level ranging from 0.4 to 0.86 over different combinations of prediction. And when we added portions of test data to the train set, the accuracy keeps going up and down drastically. And there does not seem to be any traceable pattern regarding adding what percentage of test data would help elevate test set accuracy, as for different combinations of prediction, the result varies greatly.

We conducted student t-test with [3]scipy.stats package. Table 3 provides the P-value of the accuracy values of no added test data and when a number of test data is added with SVM method. As we can see, the P-value is pretty large, meaning that there’s no difference between adding different numbers of data and not adding any test data at all. And looking from the scatterplot, it’s obvious that adding test set data into train set under SVM method does not help boost the accuracy level at all.

Accuracy Correlation	Add 10,000	Add 20,000	Add 30,000	Add 40,000
P-value	0.50	0.36	0.73	0.93

Table 3: P-value of accuracy correlation

Random Forest, Gradient Boosting Descent and Logistic Regression all have a much more stable performance within-domain, even though different train tests and test sets generate accuracy levels that could be very different, which exactly could help us understand the boundaries and generalizability of different spam groups. But when we add portions of test data to the train set, these three methods do not seem to change a lot. In fact, the accuracy level remains quite static. Below is a plot that plots the median accuracy of different train and test set combinations without adding any test data and with adding different amount of test data with these three methods. y-axis means different train and test sets. For example, “G + S2 -> G + S3” means that the train set is Genuine tweets plus Social Spambots 2 tweets, and the test set is Genuine tweets plus Social Spambots 3 tweets.

[3]From scipy.stats import ttest_ind, ttest_ind_from_stats https://docs.scipy.org/doc/scipy/reference/stats.html

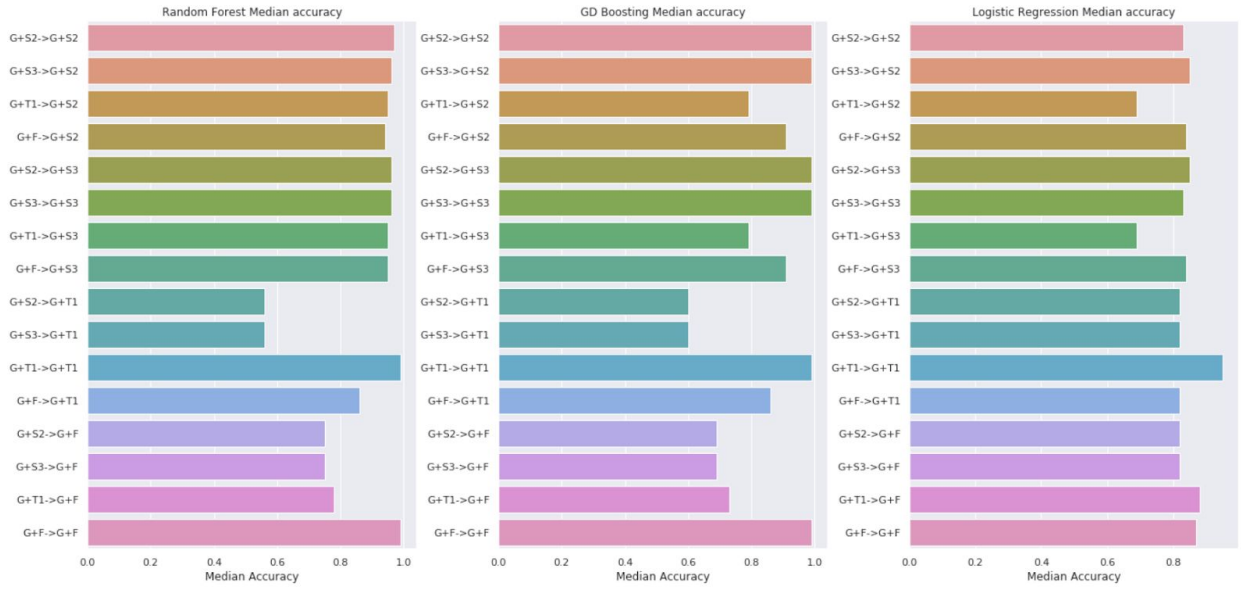


Chart 4: Mean text-based accuracy (tweet-level) bar chart

From Chart 4, we can see that Random Forest, Gradient Boosting Descent and Logistic Regression all yield relatively good performance. For Random Forest and Gradient Boosting Descent methods, Social Spambots 2 predicting Traditional Spambots 1 and Social Spambots 3 predicting Traditional Spambots 1 seem to yield an obviously lower performance, which is around 0.5-0.6, compared to other combinations of prediction. And also, for Gradient Boosting Descent and Logistic Regression methods, Traditional Spambots 1 predicting Social Spambots 2 and Traditional Spambots 1 predicting Social Spambots 3 seem to yield a lower performance, though to a lesser extent, which is around 0.65 to 0.75, compared to other combinations of prediction, as well. We can speculate that Social Spambots 2 and Social Spambots 3 are fairly different from Traditional Spambots 1 regarding simple numeric features used, and Traditional Spambots 1 is slightly more generalizable in regards to being bots than Social Spambots 2 and Social Spambots 3.

For Random Forest and Gradient Boosting Descent methods, other spam groups (Social Spambots 2, Social Spambots 3, Traditional Spambots 1) predicting fake followers tweets seems to also perform less well, though not significantly so. But it's not the case the other way around.

Chart 5: RF (Genuine + Social2)

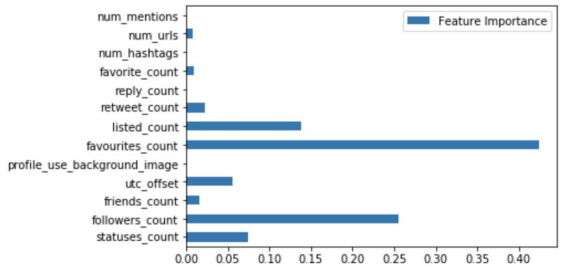
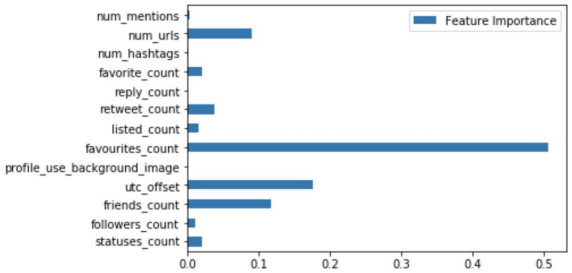


Chart 6: RF (Genuine + Traditional 1)



To understand the discrepancy of the the test set accuracy level between “G+S2 -> S+T1” and “G+T1 -> G+S2”, we look at the feature importance of the train set, with the Random Forest method, as shown in Chart 5 and Chart 6. For both predictions, “favorites_count” has dominating importance, accounting for 0.42 and 0.5 respectively. But for “G+S2 -> S+T1”, “followers_count”, “listed_count”, “statuses_count” have reasonable importance, while for “G+T1 -> G+S2”, “utc_offset”, “friends_count” and “num_urls” have reasonable importance.

	Social Spambots 2	Traditional Spambots 1	Genuine
Median followers_count	5.0	198.5	341.0
Median listed_count	0.0	1.0	2.0
Median statuses_count	56.0	16.0	6609.0
Median utc_offset	-14400.0	0	-14400.0
Median friends_count	39.0	956.0	319.0
Median num_urls	0.0	1.0	0.0

Table 4: Statistics of a few numeric values comparison

From Table 4, given the above partial statistics and the purpose of spam groups, we speculate that Traditional Spambots 1 is more maturely developed, and it covers wider decision boundaries and showcases more bot characteristics than Social Spambots 2 does. And given that the purpose of Traditional Spambots 1 is to post malicious urls online and influence communities, while the purpose of Traditional Spambots 2 is to advertise a paid app, the former could more salient and aggressive bots than the latter. The same thought process and analysis could be done for other predictions, and conclusions would be summarized in the end.

6.2.2. Account level

On the account level, we used features that are correlated with the user's profile, which including 'statuses_count', 'followers_count', 'friends_count', 'favourites_count', 'listed_count', 'verified'. As some of the datasets (Social Spambots 2 & Traditional Spambots 1) have more data points, we randomly selected 500 points in each dataset to make the balance. In the across-domain part, we added 100(⅛), 200(⅖), 300(⅜) and 400(⅘) points from the test set into the training set in each step, respectively.

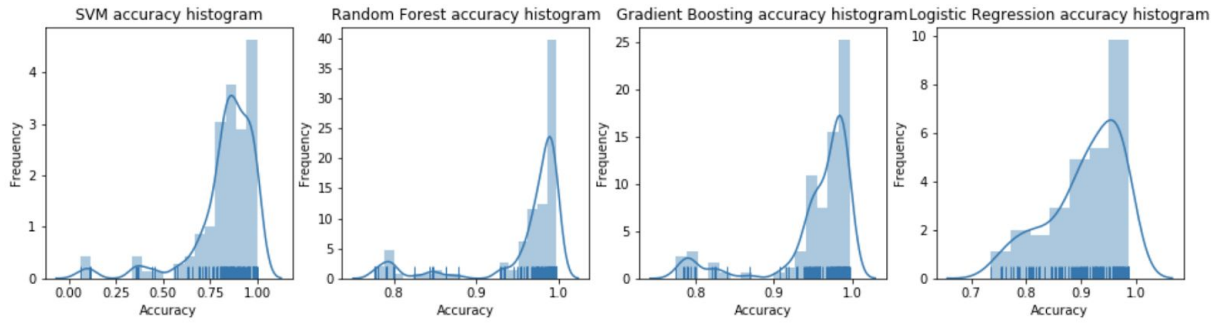


Chart 7: Accuracy histogram for account level numeric features based models

Due to adding reasonably large amount of data points, the accuracy level improves a lot when we added 100(⅛) points from the test set to the train set, and then gets stabilized as we further added data, as shown in Chart 8. The accuracy level from all the methods is mostly consistently high, ranging from 0.8 to close to 1, as shown from Chart 7. That the same reasons discussed in 6.1.2 why account-level prediction on numeric values is higher than tweet-level prediction on numeric values also apply here.

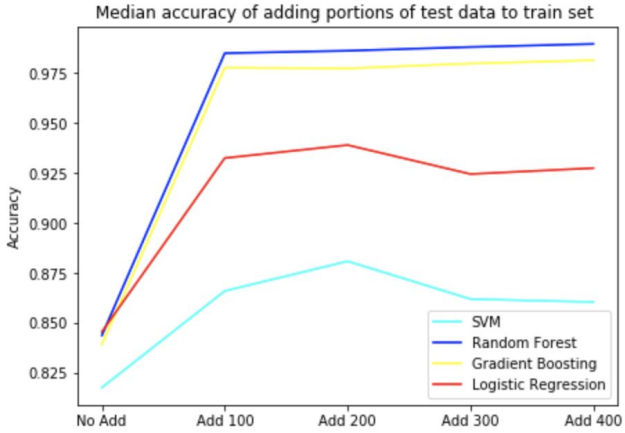


Chart 8: Mean accuracy with and without adding test data

6.3. Network-Based Method

For the network-based method, we have used pagerank and hits as features to build models, with the idea that pagerank and hits are possibly features that can differentiate bots from humans. We sampled 20000 tweets from each of the spam groups: social spambots 2, social spambots 3, traditional spambots 1 and fake followers, and 80000 tweets from the genuine group. The directed network was built on the basis of user replying to each other, one of the main user interactions on social media. This can be done by extracting “user_id” and “in_reply_to_user_id” columns from tweets. Thus, the nodes of the network consist of all the user ids from the tweets we sampled, as well as the outside user ids that the users in our tweets reply to, and the edges are multiple tuples of user ids and the user ids they reply to respectively.

If we looked at the network statistics, there are 32174 nodes and 33091 edges. The average in-degree and average out-degree are 1.0285. When we divide train set and set set into 70% and

30% of all data points respectively with 5 Cross Validations (cv=5), and apply different machine learning methods onto them, the test set accuracy looks like the following:

Network-Based	SVM	Random Forest	Gradient Boosting	Logistic Regression
Mean Test Accuracy	0.90	0.96	0.96	0.90

The performance of all models looks pretty high, but dubious. We then calculated the median pagerank and median hits of both spams and humans ('label'=1 and 'label'=0), the result is surprising, which is that they are exactly the same: 2.5700175422880653e-05 for both parameters for both sides. This is telling that the features do not make much sense, and the models' high performance does not indicate the legitimacy of the relationship between pagerank, hits and the label. Thus, this method does not hold credibility. But it's still meaningful to try it out on other datasets with distinctively different pagerank and hits index, and the according results and conclusions may differ.

7. Challenges

7.1. Project Adjustment and Alternative Methods

As we progressed, it sometimes happened that we found it difficult to implement certain things we planned, or that the plans did not work out well. For example, originally we were focusing on predicting bots and improving predictive performance. We later found that our baseline models have a high performance already with simple features. If we just want to achieve high predictive accuracy, it does not make sense to try other methods. But we surely cannot stop there. Thus, we adjusted our project goal and scope accordingly, and proceeded to try other methods, with an intention to analyze, learn, obtain more and deeper insights apart from achieving the best predictive accuracy possible.

7.2. Pipelines, Memory and Runtime

For in-domain and cross-domain predictions, we ultimately managed to get a lot of prediction results. This benefitted from us building sound machine learning pipelines, removing redundancies and excessive codes and using Flux as a cloud computing technology to help run massive volumes of data. This was not the case from the beginning to halfway of the semester, when we were having some difficulties in efficiently and effectively reusing codes and generating result files with different input data.

7.3. Visualization and Summarization of Results

In order to show our results and better communicate our insights, there is no way to not include visual charts and plots. We found it hard to make proper visualization that is neat, understandable and supports sense-making. In the end, we couldn't say our visualization is perfect, but it is better than what we were doing previously. A central shift of mind is to take it more as a tool to communicate ideas, rather than an alternative way to showcase result statistics. Thus, there is no need to fill the chart with data. In fact, it is strongly discouraged. On top of

that, we have made progress in using the right visualizations. We, the project authors, are responsible for processing and extracting information, and conveying the messages to readers in a clear, concise and understandable way.

7.4. Organizing Thoughts and Insights

Last but not least, we have been able to do better in organizing and summarizing insights. Doing it could be daunting, overwhelming and requires a lot of patience. But to constantly ask ourselves questions like what is the main line that connects all pieces of the project together, what is our goal and what are some helpful conclusions we have gotten so far helps a lot. Besides, it makes sense to always document thoughts and findings, or even unsuccessful attempts every step along the way, so that we don't lose them and know how to do better eventually. Break the project into small parts and make titles and subtitles. Ultimately, we just need to fill in each part well.

8. Conclusions

Account-level prediction yields higher accuracy than tweet-level prediction, possibly due to the reason that account-level takes into complete characteristics of each user with all tweets being concentrated and ingrains those into the model-building. Besides, weights have been given to active and inactive users to remove biases.

SVM gives relatively poor performance compared to the other methods we have used. For the most part, Random Forest, Gradient Boosting and Logistic Regression models have fair and stable performance that are well over 0.65. The upper-bound could be very close to 1.0.

The decision boundary and scope of different spam groups can be derived based on our analysis. Chart 9 visually showcases the relationship between groups we draw, with the size of each circle representing a rough speculation of the range of each group.

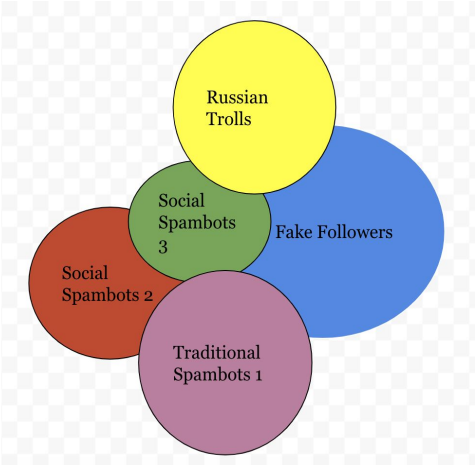


Chart 9: Decision boundary of each spam group

Regarding the textual feature of spams, if it is more messy, more abstract, covers more topics, has more spelling errors and involves more nonsense, it's more likely to make decent train set

that can predict well on other spam groups. On the contrary, if the text is more specific, more neat and more like human languages, it is more likely to predict poorly on spam groups that are more spam-like.

Between text-based and numeric values-based predictions, the latter yields an overall more stable predictive accuracy, as numeric values-based features tend to be more objective, less diverse and less prone to change than text-based features. But texts offer a broader, fresher, more direct and more interesting perspective into understanding how bots leave their footprint online, and how they try to negatively influence users and communities.

Adding portions of test data to the train set does help improve the test set accuracy. It helps to try adding different amount and document the accuracy level, so that we know after which point the marginal benefit becomes smaller. In real life, the improvement on accuracy vs. cost of adding portions of test data to the train set is a trade-off, depending on how accuracy we want the prediction to be and how much we are willing pay to to gain extra data.

Acknowledgement

We would like to sincerely thank Prof. Ceren Budak at University of Michigan, School of Information for her valuable guidance on our project and all the classmates involved in the course SI699 Big Data Analytics at University of Michigan, School of Information in Winter 2019. This project could not have been done without all of their support.

Reference

[1] Xiao C, et al., 2015. Detecting clusters of fake accounts in online social networks. The eighth ACM workshop on artificial intelligence and security . 91–101.

[2] Wang, Alex Hai, 2010. Detecting Spam Bots in Online Social Networking Sites: A Machine Learning Approach. Data and Applications Security XXIV, LNCS 6166 . 335–342.

[3] Subrahmanian, VS, et al., 2016. The DARPA Twitter Bot Challenge. IEEE Computer. Preprint arXiv:1601.05140.

[4] Gurajala, Supraja, 2015. Fake Twitter accounts: profile characteristics obtained using an activity-based pattern detection approach. 2015 International Conference on Social Media & Society. 9-12.

[5] Varol, Onur, et al., 2017. Online Human-Bot Interactions: Detection, Estimation, and Characterization. Intl. AAAI Conf. on Web and Social Media (ICWSM).

[6] C. Stefano, P. Marinella, S. Angelo, T. Stefano, On the Capability of Evolved Spambots to Evade Detection via Genetic Engineering, Online Social Networks and Media, 2019(9): 1-16.

[7] B. Wang, A. Zubiaga, M. Liakata, and R. Procter. Making the most of tweet-inherent features for social spam detection on twitter. 6th WWW workshop on 'Making Sense of Microposts', 2015.

[8] I. , I. Minambres Marcos, C. Laorden, P. Galan-Garcia, A. Santamaria-Ibirika, and P. G. Bringas. Twitter content-based spam filtering. In International Joint Conference SOCO'13-CISIS'13-ICEUTE'13, 2014: 449–458.

[9] K. Kamalanathan, K. Preethi, An Integrated Approach to Spam Classification on Twitter Using URL Analysis, Natural Language Processing and Machine Learning Techniques, IEEE Students' Conference on Electrical, Electronics and Computer Science, 2014.

[10] M. Juan, A. Lourdes, Detecting Malicious Tweets ini Trending Topics Using a Statistical Analysis of Language, Expert Systems with Applications, 2013(40): 2992-3000.

[11] A. Faraz, A. Muhammad, A Generic Statistical Approach for Spam Detection in Online Social Networks. Computer Communications, 2013(36): 1120-1129.