Penn Blockchain Conference
Bot Detection Algorithm Challenge - Synoptic
24th February 2024

Tommaso Campi
Pietro Del Bianco
Letizia Dimonopoli

Bot Detection Algorithm Using Semi-Supervised Machine Learning Methods and Benford’s Law.

1. Introduction

Synoptic is an innovative platform for exchanging information based on blockchain that aims to promote trust and transparency within its online community. To achieve this, Synoptic has implemented several features that help identify users, including:

1. Reputation Score: This metric reflects the reputation of the author of a post and is based on factors such as the confidence level assigned to the post and the number of upvotes and downvotes received.
2. Influence Score: This score indicates how influential a user is on the platform and is determined by factors such as the number of followers and the identity of those followers.
3. Vote Power: Each user has the ability to impact the reputation score of others through upvotes and downvotes. The weight of a user's vote depends on their activity level on the platform.
4. Confidence Level: This metric represents the level of confidence attributed to a user's post.

As Synoptic tries to foster a trustworthy and transparent online environment, one of the primary challenges it faces is the presence of bots. Detecting and eliminating bots is crucial for ensuring the reliability of a social network, particularly when discussions involve sensitive topics such as finance and cryptocurrencies.

In this paper, we aim to analyze the issues associated with bot detection and propose solutions to address them.

When considering this problem, several challenges may arise that could potentially affect traditional machine learning algorithms for bot detection. Some of these challenges include:

1. Imbalance of the Dataset: Typically, only a small percentage (around 9%) of users on a social network are bots, leading to a significant class imbalance in the dataset.
2. Volume of Data: Social networks generate vast amounts of data daily, making it challenging to process and analyze efficiently. For example, Twitter alone generates approximately 500 million tweets every day.
3. Feature Complexity: There may be numerous features to consider when detecting bots, leading to high computational costs and potential performance issues.
4. Differences Across Platforms: Different social networks may exhibit varying bot characteristics, requiring tailored detection approaches.

5. Corner Cases: Various scenarios, such as coordinated bot networks, highly or lowly active accounts, and AI-generated trends, present additional challenges for bot detection.

By addressing these challenges and developing robust detection algorithms, Synoptic can enhance the trustworthiness and reliability of its platform for all users.

2. Twitter dataset

To suggest a method to better identify the probability of a user being a bot within a social network, we decided to use a dataset from Twitter. The dataset originally consisted of 58 variables and more than 100,000 rows. Therefore, it was a comprehensive enough dataset to carry on our analyses. It was also chosen because there is a "BotScoreBinary" column, which helps us in our study in identifying whether a user is a bot or not. Specifically, our initial dataset has a bot percentage which equals 4%. However, to be able to use the variables from the guidelines, we performed feature engineering. The following are the variables we needed and how we created them:

1. User Weight Score. Since it represents the user's reputation within the network, it was computed doing the weighted average of the favorites_count, statuses_count and quotes variables. The favorites_count feature represents the number of favorites across all tweets by the user; the status_count represents the number of tweets by the user and the quotes represent the number of times the user has been quoted. This should give us an approximation of the importance a specific user has in the community. Since the quotes distribution was highly skewed, we converted it into a binary variable, thus having 1 if the user has quotes, and 0 otherwise. We further applied a log transformation to favorites_count and statuses_count to reduce their skewness. As requested by the guidelines, each user was assigned a user weight score which ranges from 1 to infinity.
2. Post Confidence Score. This feature reflects the author's confidence level in the content of the post. Assuming a normal distribution both for the bots and for the humans, the post confidence score variable was created by using a normal distribution around 0.75 and a standard deviation of 0.1. The lower mean and higher standard deviation might be appropriate to reflect the uncertainty regarding the confidence of the content of the post. The post confidence score will result in a variable that ranges from 50% to 100% for each post.
3. Influence Score. It represents the impact a specific user has in the community. We engineered it by computing the linear combination between the number of followers and the number of friends.
4. Voting impact. Since the upvotes and the downvotes affect the user's score, this feature has to be calculated in the following way: user_weight_score * voter_weight_score * (up ? 1 : -1). For the indicator function, we used a column called 5_label_majority_answer, which contains values which are categorized in the following classes: "Agree", "Mostly Agree", "No Majority", "Mostly disagree",

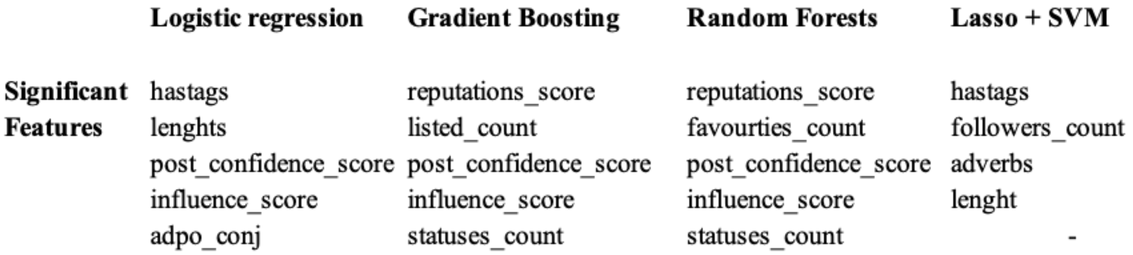
"Disagree". To follow the guidelines, we mapped the majority label upvotes and downvotes to a range of [-1:1].

Therefore, all of these variables were created through Twitter's dataset, but they are closely related to Synoptic features. For instance, the User Weight Score represents Synoptic's Reputation Score, which is mentioned earlier in this paper. The Post Confidence Score corresponds to the Synoptic's Confidence Level, which also ranges from 0.5 to 1. Regarding Twitter's agree and disagree categories, they correspond to Synoptic's upvotes and downvotes.

The cleaned dataset includes also the following self-explanatory variables: 'statement', 'tweet', 'followers_count', 'friends_count', 'listed_count', 'BotScoreBinary', 'mentions', 'replies', 'retweets', 'hashtags', 'Word count', 'Average word length', 'present_verbs', 'past_verbs', 'adjectives', 'adverbs', 'adpositions', 'pronouns', 'conjunctions', 'capitals', 'digits', 'length', 'symbols_percent'.

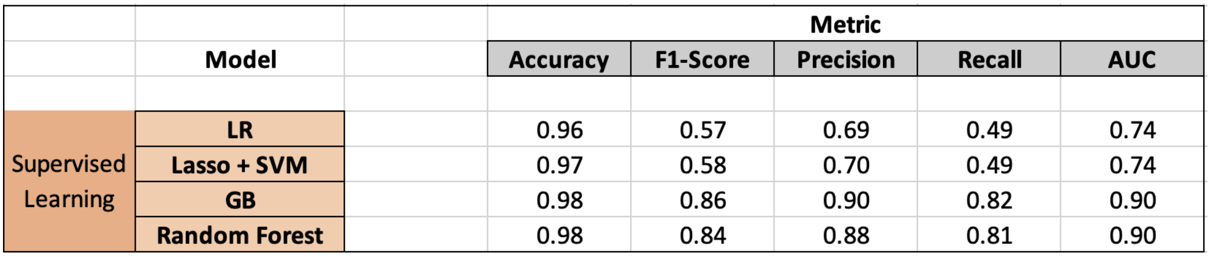
3. Supervised machine learning algorithms

We want to observe which variables are more influential in our dataset therefore we run the following supervised machine learning algorithms. Below you will find a table containing significant features for identifying bot against human user using different machine learning algorithms.



	Logistic regression	**Gradient Boosting**	**Random Forests**	**Lasso + SVM**
Significant Features	hastags	reputations_score	reputations_score	hastags
	lenghts	listed_count	favourties_count	followers_count
	post_confidence_score	post_confidence_score	post_confidence_score	adverbs
	influence_score	influence_score	influence_score	lenght
	adpo_conj	statuses_count	statuses_count	-

The following are the results of the algorithms:



			Metric				
	Model		**Accuracy**	**F1-Score**	**Precision**	**Recall**	**AUC**
Supervised Learning	**LR**		0.96	0.57	0.69	0.49	0.74
	Lasso + SVM		0.97	0.58	0.70	0.49	0.74
	GB		0.98	0.86	0.90	0.82	0.90
	Random Forest		0.98	0.84	0.88	0.81	0.90

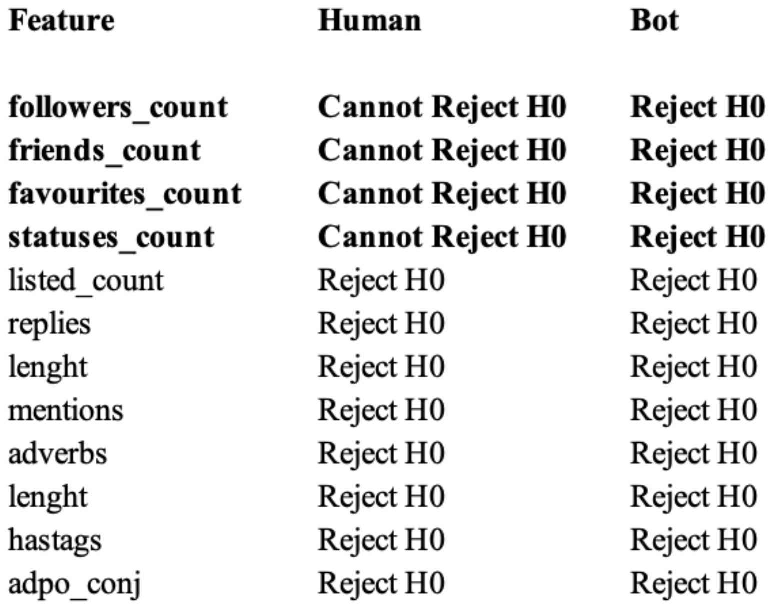
4. Benford's Law

To be able to classify bots and humans using semi-supervised machine learning, we first evaluated feature importance by applying Benford's Law. Benford's Law was invented in 1881 and since then it has been implemented in various fields, comprising network intrusions and online social networks. Benford's Law states that in many naturally occurring sets of numbers, the leading digits are not evenly distributed, with smaller digits (like 1) appearing

more frequently than larger digits (like 9). It has been demonstrated that Benford’s Law can be applied to Twitter data, therefore we decided to use it in our analysis. The following are the hypotheses:
Null hypothesis ($H0$) = a feature obeys the first significant leading digit (FSLD) distribution.
Alternative hypothesis ($H1$) = a feature violates the FSLD distribution.
In the context of Twitter numerical features, such as followers_count and friends_count, analyzing the distribution of significant leading digits can provide insights into the behavior of user accounts.

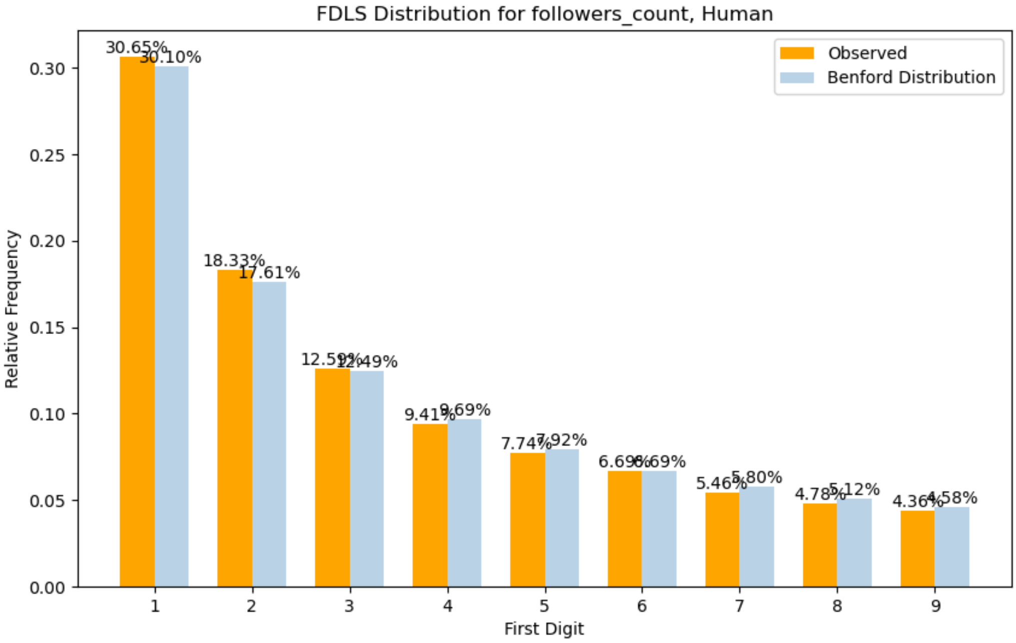
There are multiple advantages in using Benford’s law in our study. The main one is that it analyzes leading digits instead of absolute numbers because each number has a unique First Significant Leading Digit (FSLD), which ranges from 1 to 9. This allows a more uniform and predictable distribution. Furthermore, the fact that parameter fitting is not a requirement for Benford’s law implementation makes it a very straightforward method to use. It makes it better when compared to nonuniform distributions. One of our main issues was that our Twitter dataset is unbalanced, but this isn’t a problem with Benford’s law because since it only cares about the FSLD distribution, the fact that data sizes might not be in proportion is unimportant. Benford’s law can thus be employed with imbalance datasets, and even though other advanced FS methods such as Random Forests could, Benford’s law is not as computationally expensive. Moreover, note that Benford’s law can produce similar results when compared to Random Forest, with the advantage of being more computationally efficient. This gives the opportunity to find bots faster and in real time. Finally, another issue we encountered with Twitter’s dataset was that in the numerical features there is no difference between humans and bots, however Benford’s law has also the capacity to uncover anomalies through FSLD.

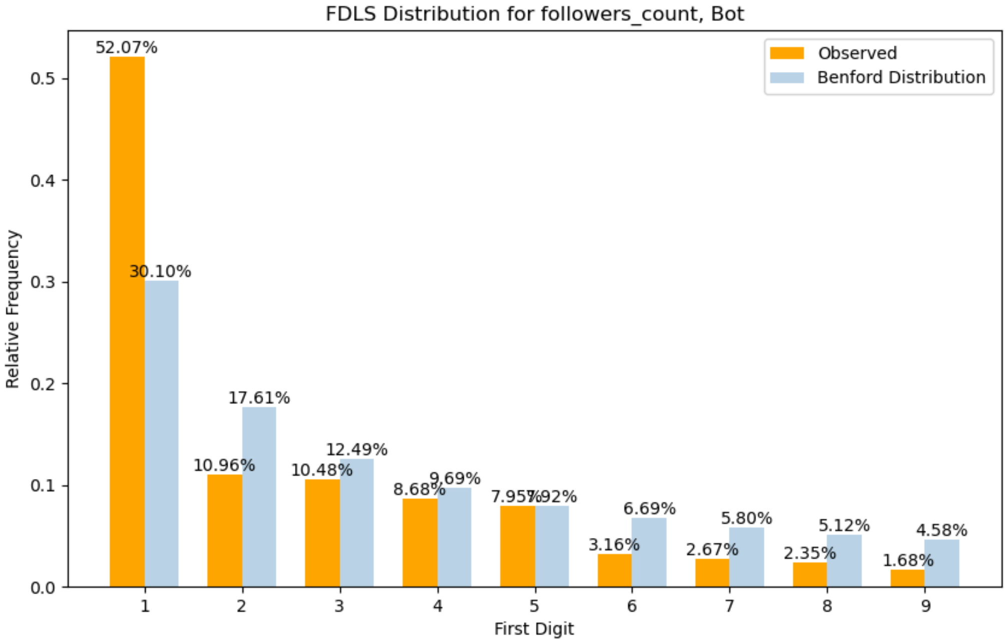
We have implemented Benford’s law for the variables which we listed below with their respective results:



Feature	**Human**	**Bot**
followers_count	**Cannot Reject H0**	**Reject H0**
friends_count	**Cannot Reject H0**	**Reject H0**
favourites_count	**Cannot Reject H0**	**Reject H0**
statuses_count	**Cannot Reject H0**	**Reject H0**
listed_count	Reject H0	Reject H0
replies	Reject H0	Reject H0
lenght	Reject H0	Reject H0
mentions	Reject H0	Reject H0
adverbs	Reject H0	Reject H0
lenght	Reject H0	Reject H0
hastags	Reject H0	Reject H0
adpo_conj	Reject H0	Reject H0

The features in bold are significant. Graphically, we can see the example of the FSLD distribution for the variable followers_count, that shows that Benford's Law is working:





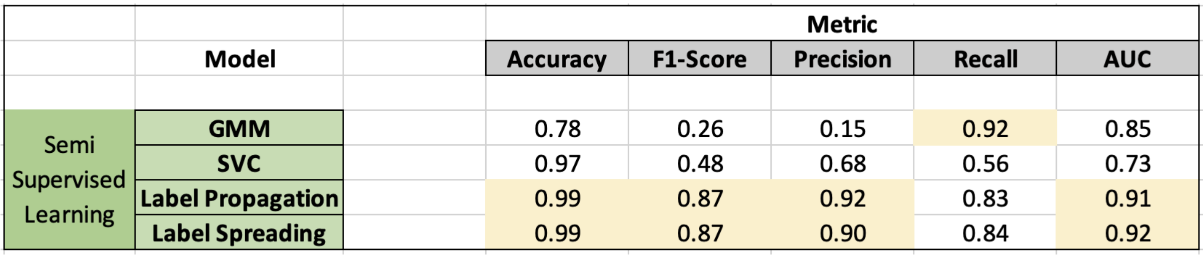
5. Advantages of semi-supervised methods and differences against supervised and unsupervised methods

We decided to implement semi-supervised methods because they show several advantages. First, semi-supervised methods can often achieve good results with relatively small labeled datasets, making them more efficient than fully supervised methods that require large amounts of labeled data. They are also more robust to noise in the labeled data, as the model can learn from the unlabeled data. Finally, as we demonstrated in our analysis, semi-supervised methods reveal to be more cost-effective and with better performance compared to traditional supervised methods.

Using the variables that we previously identified through Benford's Law, there are a few semi-supervised models that we decided to use in our study.

First, we realized a Gaussian Mixture Model (GMM). GMM is able to model the underlying distribution of the data. It is considered suitable to evaluate continuous data. Second, we implemented a semi-supervised SVM (S3VM), which applies the maximum margin principle to construct a binary classifier by leveraging both labeled and unlabeled datasets. Third, we moved to a semi-supervised label propagation. This method is a semi-supervised machine learning model that operates on graphs. It utilizes a subset of nodes with known labels to propagate these labels to all nodes in the graph until a stable labeling is reached. Finally, we have semi-supervised label spreading. The label spreading method is similar to the LP method, but with a key difference: the labeled data points assigned to vertices can change as the iteration progresses.

Our results are the following:



			Metric				
	Model		Accuracy	F1-Score	Precision	Recall	AUC
Semi Supervised Learning	GMM		0.78	0.26	0.15	0.92	0.85
	SVC		0.97	0.48	0.68	0.56	0.73
	Label Propagation		0.99	0.87	0.92	0.83	0.91
	Label Spreading		0.99	0.87	0.90	0.84	0.92

6. Research limitations

There are some limitations to our study. This however shapes the path for further developments.

1. For instance, some bots have adversarial behaviors that prevent existing algorithms to identify them.

    1. They could be created in such a way that they mimic human behavior so closely that it can be difficult for algorithms to detect the difference from real human users;
    2. Bots can also be set to adopt content variation, thus being programmed to vary the shared posts, both considering the content and the vocabulary or writing style;
    3. Randomization can also represent an issue, especially when the bot is programmed to have random interaction patterns, because it makes them more natural and thus they might seem more human;
    4. Some remain dormant or have limited activity for prolonged periods of time, making it harder for algorithms to identify them;
    5. If there are new bots, with a very tiny activity level, algorithms may find it complex to distinguish them from human users;

6. Finally, it's important to underline that bots can be programmed to have an adaptive activity that understands the mechanism of detection algorithms, thus managing to adapt to them.

These corner cases need to be taken into consideration when considering why current solutions might not be good enough.

2. From our results with Twitter dataset analysis, we understood that additional features can be suggested for a more complex bot detection algorithm. For instance, we could consider the following additional variables: URL features, the length and special characters present in usernames, the account age, the email address related to the account, the activity time of the user, the number of posts of the user, the listed count, geolocalization, the profile picture, whether there are any duplicate profiles, a spam word count (through SpaCy), the ratio between followers and following. Furthermore, a semantic analysis could be done to analyze the content uploaded in Synoptic's tournaments.

On a side note, feedback loops could be designed to constantly improve the algorithm's accuracy and effectiveness. The feedback loop for our bot detection algorithm would involve continuously collecting data on user behavior, using this data to detect potential bots, and then gathering feedback on the algorithm's performance to improve its accuracy and effectiveness over time. This iterative process of detection, verification, feedback gathering, and algorithm update helps the algorithm adapt to new patterns of bot behavior and improve its ability to detect bots on the platform.

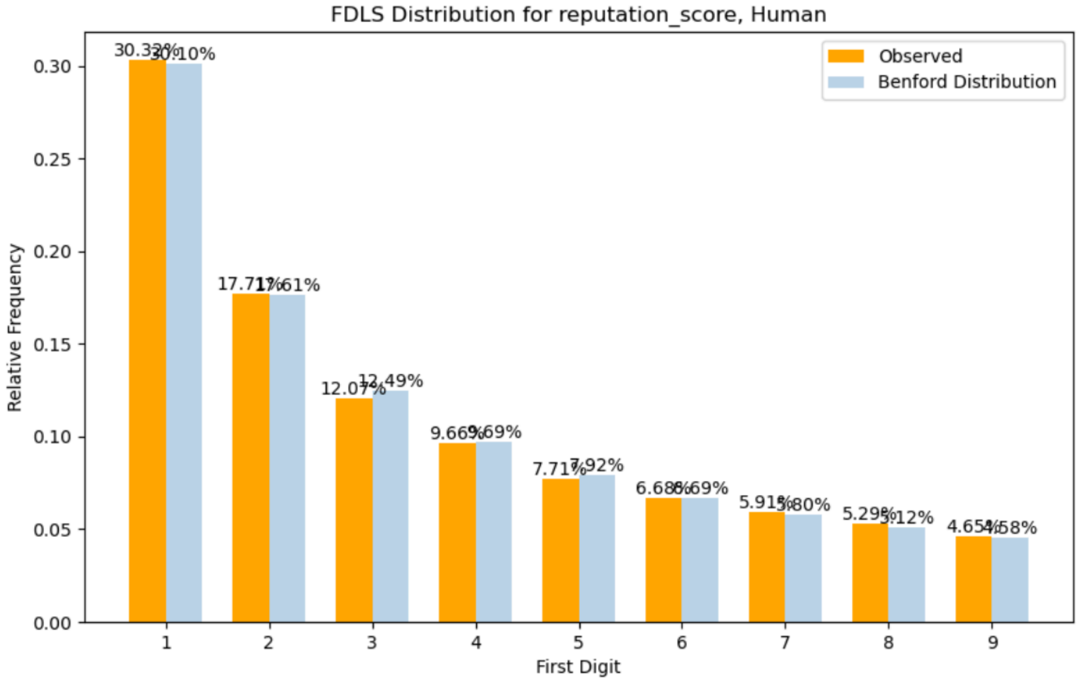
3. Our bot detection algorithm should be performant in detecting malicious activities. One of the malicious and most complex bot activities we have not already mentioned is represented by the existence of botnets, which are networks of bots. In fact, botnets are networks of compromised computers or devices that are controlled by a central server. They are considered malicious because they can be used to carry out large-scale attacks without the knowledge or consent of the compromised device owners, leading to widespread harm and disruption.

4. Finally, generative AI technologies can affect the malicious activity types and suggest solutions accordingly. In fact, generative AI technologies could track the changes in malicious activity types (for instance the fact that scammers are easily created) and a potential solution to that could be represented by implementing a blockchain mechanism where, if the bot is detected, it is forever deleted from the blockchain network.

The main limitation of this study is the lack of an authentic Synoptic dataset. A further analysis would be required, and we suggest a case study to be implemented.

Furthermore, you should consider exploiting other ways to compute your metrics, such that they follow Benford's Law. In particular, we recomputed the user weight score based on the power of two, thus getting that the null hypothesis couldn't be rejected, as we wanted to demonstrate.

Below are the graphical results:



7. Conclusions

Once we identified the significant features thanks to Benford’s Law, we implement four semi-supervised algorithms: GMM, SVC, label propagation and label spreading. We believe our findings will help in detecting bots and reducing cyber threats.