Political astroturfing on Twitter

Identifying spam-campaign collaborators on social media

Sergej Roswall Bogachov

Handed in: May 31, 2021

Contents

Abstract

Astroturfing in the context of political disinformation/spam campaigns on social media is a rising issue. Paid groups of workers are being deployed on social media by malicious actors in order to spread disinformation to destabilize countries, elections, companies, and products. In general, social media platforms and other tech companies need the required knowledge and tools to implement countermeasures and detection techniques to tackle such novel threats. In this project, we study the application of machine learning techniques to the domain of spam-campaigns in order to automatically detect the malicious users. We show that a typical inauthentic account reveals several significant indicators, such as hashtag similarities, account lifespan and the sentiment of the tweet. We show that it is possible to achieve high-precision results when using these indicators in a ML detection model. Furthermore we discuss some possible techniques that the spammers can use to avoid detection, using the capabilities that the Twitter platform provides.

Introduction

The concept of astroturfing and propagation of false information in order to persuade public opinions is not new. In 1986, the US senator Lloyd Bentsen coined the term “astroturfing”, which he used to describe “the artificial grassroots campaigns created by public relations (PR) firms”[2]. Today, the practice of astroturfing has spread to other domains, and is especially utilized online, in order to manipulate the public’s perception of political opinions, public figures, companies and product reviews. There is a broad consensus in literature, that astroturfing in fact is an effective tool in influencing opinions, due to various psychological factors[2][3][4]. Authors of [4] boil down the effectiveness of astroturfing to: “people believe in what most of the other people say”.

Despite of the effectiveness of astroturfing, the resources required to execute a coordinated and persistent campaign should not be underestimated. Historically, only big lobby organizations or companies have been associated with such campaigns, as they can afford to continuously finance the astroturfing efforts until the disinformation has been propagated sufficiently enough to the public. It has also been discovered that astroturfing is being utilized by state-sponsored actors in order to polarize public opinions on political matters and thereby meddle with other countries internal affairs[5][6]. One of the recent and most prominent cases of such a state-sponsored spam-campaign, is the russian interference in the 2016 US election. It is suggested that the Internet Research agency (IRA), a St. Petersburg based company specializing in online influencing, has controlled 3,814 inauthentic Twitter accounts (trolls) engaging in astroturfing on US politics in the period of 2012-2017[7][5]. In this project, we focus on the activity of a subset of these accounts during the 2016 US election. More specifically, our goal is to study the application of machine learning techniques in the context of the russian spam-campaign, in order to produce and evaluate detection models suitable for identification of spam-campaign collaborators. During this study, we hope to get answers and insights into the question; *is it possible to leverage machine learning techniques to establish a common profile of known spammers from the russian IRA campaign and use the results to identify yet unknown spam-campaign participators?*

We will start out by briefly scoping the project, describe its limitations and also provide some definitions on the terms used throughout the report. We follow up with a description of the data, together with an explanation of how we obtain a baseline dataset. Then, we perform a correlation analysis of different hypothesis about the authentic and inauthentic accounts in order to derive usable variables for the detection model. The results of the analysis will be used directly in the training and evaluation of the models. We will briefly touch upon the topic of adversarial machine learning and study how the trolls can leverage prior knowledge of the models in order to avoid detection. At last, we will reflect upon our findings and discuss different pitfalls that one may encounter during the process of identifying spam-campaign participators.

Limitations

In this section, we will try to limit the scope of the project and identify other limitations which may have an effect on the results. As previously stated, the goal of the project is to study application of machine learning for identification of russian trolls participating in astroturfing during the 2016 US election. It is not within the scope of the project to ensure the integrity of the troll dataset. We assume that the troll data produced by researchers from Clemson University is valid and the spam-campaign indeed is a state-sponsored attempt to meddle with US politics. Documenting the influence and impact of the spam-campaign is covered in other literature[40][41] and is also out of scope of this project. Causal analysis of the data is also out of scope. We are not particularly interested in the cause of the observed effects in our data, but merely to identify significant correlations we can use for detection of trolls.

By choosing an existing dataset of inauthentic accounts, we automatically confine the detection model to the features which can be extracted from this data. Since the inauthentic accounts have already been removed by Twitter, we do not have the ability to query additional information on these accounts, thus limiting our analysis to the features described in the data section. Also, the format of the troll dataset is quite unique, making it hard to find pre-existing baseline data of authentic accounts for comparison. As a consequence of this, effort has been put into sampling and processing the baseline dataset of authentic accounts from scratch. Therefore, the results of this project has to be viewed in the light of the dataset creation by two different entities. Nevertheless, much care has been taken to avoid bad sampling and to ensure the compatibility of the troll and the baseline data.

Background

Throughout this project, we will be investigating astroturfing within US politics. In this section we will provide a short definition of some of the concepts that we are going to study, in order to get the reader acquainted to the domain.

Twitter

Twitter is a social media platform founded in 2006. With its 192 million users it is one of the biggest social media platforms in 2021[23]. Its primary purpose is for sharing short messages, a.k.a. "Tweets", of at most 280 characters. It is possible to include other content like links to videos and images as well. The networking on the platform is done through the concept of followers and following. A user can follow a number of other users in order to add more content to their feed. Twitter is widely used by influencers and other public figures to provide relevant updates to their followers.

Disinformation vs misinformation

The terms disinformation and misinformation seems to be used interchangeably in some of the literature. However, we will distinguish between these two during this project. We will use the definition from [24]:

- Misinformation: "false information that is spread, regardless of intent to mislead."
- Disinformation: "false information, as about a country's military strength or plans, disseminated by a government or intelligence agency in a hostile act of tactical political subversion."

The main difference between these two definitions is the intent behind the spread of false information. Since we are studying state-sponsored political astroturfing, by the above definition it is clear that we are dealing with disinformation rather than misinformation.

Spam campaign

Authors of [5] describe the activities of malicious russian "trolls" participating in "industrialized political warfare". In the context of our work, we are studying the collaboration of these inauthentic users which have common goal of polarizing the opinions on US politics through the use of Twitter. The term "spam campaign" will loosely refer to these definitions throughout the report.

Troll accounts

Traditionally, internet trolls are associated with anonymous internet users posting content on platforms in order to get attention and provoke negative reactions from others. Their motivation is often personal amusement, but some also have specific agendas. The creators of the "russian troll dataset", describe the inauthentic users participating in the spam campaign as "russian trolls". The trolls spread divisive and upsetting disinformation on US politics in order to provoke reactions from authentic users. The motivation seems to be polarization of opinions in the political landscape as discussed in [5] and [6].

Well-known Twitter accounts used for experiments

During the experiment phase, we will use some well-known US Twitter accounts for benchmarking and examples. Some prior knowledge on these accounts might be useful in order to interpret the results. The accounts are:

- Mitch McConell: The leader of the republican party in the US senate
- Sarah Palin: Former republican governor of Alaska. 2008 US election vice-president candidate.
- Elon Musk: Inventor and engineer. Director of SpaceX and Tesla Motors. Very active on Twitter.
- Hillary Clinton: Former New York senator and Minister of Foreign Affairs for the democratic party. Nominee for president of the US in the 2016 election.
- Nancy Pelosi: The speaker of the US house of representatives.
- Donald Trump: Media celebrity and owner of the Trump Organization. Republican presidential candidate in the US 2016 election. The 45th president of the United States.

Data description

In this section, we will examine the data which will be used throughout this project. The section is divided into 3 parts: description of the chosen dataset of the inauthentic accounts, construction of a baseline dataset with authentic accounts and at last a short discussion on enhancing the dataset with features which we can use in our final detection model.

Russian Troll Tweets dataset

The primary data used in this project is the Russian Troll Tweets dataset published by fivethirtyeight[1]. The dataset contains over 3.000.000 tweets of Twitter accounts identified to be associated with IRA[5]. The troll accounts are identified and collected by researchers from Clemson University, Darren Linvill and Patrick Warren. The data is of great value to this project and to the research community in general, as information on labeled inauthentic accounts is hard to come by. Furthermore, the dataset is a full-archive snapshot of the accounts tweet activity from 2012 to 2017, meaning we can perform analysis on the long-term activity of the accounts and not just single tweets. In this project, we will be looking at activities in 2016, which is the US election year. In addition to the US presidential election, this year has a few other interesting political events, such as increased focus on the BLM movement, the Podesta's email leak and Hillary Clinton's collapse at the 9/11 memorial. Happenings as these can be of importance as the trolls tend to target their activities around specific world events[6]. The following list summarizes the features which are part of the russian troll dataset:

- **author:** a unique identifier of the tweets author.
  Example: '333147649'
- **content:** the contents of the tweets. If it is a retweet, the contents are a copy of the referenced tweet. If it is a quote, the contents are the actual quote.
  Example: 'Consistency Remains Key on Social Media'
- **publish_date:** ISO 8601 UTC timestamp of the tweet.
  Example: '2016-12-10 10:04:11+00:00'
- **post_type:** Can be blank, indicating a normal post, RETWEET or QUOTE.
- **retweet:** a boolean flag indicating if the tweet is a retweet.
- **tweet_id:** a unique identifier of the tweet.
  Example: '808371381901529088'
- **followers:** the number of accounts following the author at the time of the tweet.
- **following:** the number of accounts that the author is following at the time of the tweet.
- **account_category:** one of five categories specified by the creators of the troll dataset.

As shown in figure 1, the accounts in the dataset are divided into 5 categories based on the type of information that they post. 'LeftTroll' and 'RightTroll' users mostly tweet far-left and far-right political messages and point of views. The authors of [5] emphasize that one of the trolls agendas is to increase the polarization of the public opinions, thus moving the political spectrum away from the middle and more into far-left and far-right. That is why the troll dataset contains percieved supporters of both the republican and the democratic party. The 'NewsFeed' category are accounts which are trying to act as news outlets in order to reach a broader group of people. 'Fearmonger' accounts generally try to post alarming misinformation content with the intent to spread fear. Users labeled 'HashtagGamer' mostly post benign tweets encouraging people to participate in so called "hashtag games"[35]. These observations suggest that account categories 'LeftTroll' and 'RightTroll' are more politically active than accounts from other categories. We will therefore refer to 'LeftTroll', 'RightTroll' as

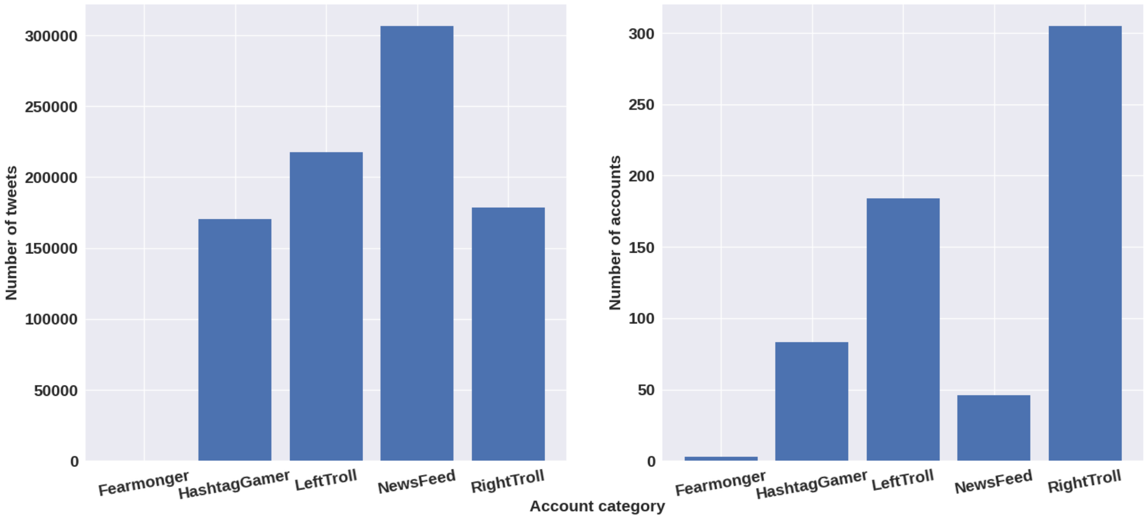

Figure 1: Account category metrics

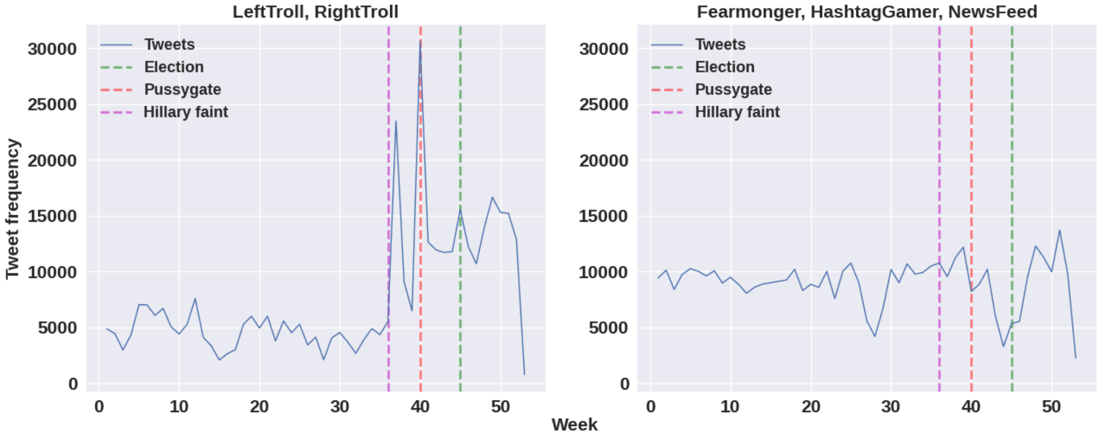

Figure 2: Tweet frequencies of different accounts categories by week (year 2016)

political accounts, and ’Fearmonger’, ’HashtagGamer’, ’NewsFeed’ as non-political.

Figure 2 shows the tweet frequency of troll users, grouped by political and non-political accounts, together with labelings of political events which got a lot of media attention in 2016. For the political active accounts, there is a clear spike in tweet activity around the marked political events. At the week of the presidential election, labeled ’Election’, the tweet frequency rises from 12.000 to over 15.000 tweets per week. Even larger spikes are observed around the week labeled ’Pussygate’, when the infamous Donald Trump quote ”grab them by the pussy” reached the media[9], and also the week labeled ’Hillary faint’, which marks the event of Hillary Clinton fainting at stage at a 9/11 memorial[8]. For the non-political categories, we do not observe any significant tweet activity spikes during those weeks. This further indicates that the ’RightTroll’ and ’LeftTroll’ accounts are more aware and active around political events than the other categories. As the main scope of this project is to study the russian interference in US politics, we choose to discard the categories ’Fearmonger’, ’HashtagGamer’ and ’NewsFeed’ from the dataset and focus solely on the ’LeftTroll’ and ’RightTroll’ accounts.

Creating the baseline dataset

In order to create and evaluate a detection model for troll accounts, we need some labeled baseline data of benign Twitter users. There exists plenty of Twitter datasets in the wild, but many of them are not compatible with the troll account dataset due to its specific nature. The reason being, that the troll account dataset is a full-archive collection of specific accounts tweets while most datasets available online are simply collections of random sampled tweets, without any user-specific information. In order to create a proper user profile for model evaluation, it is ideal to have the entire tweet history of the users. A part of this project has therefore been dedicated to creating a baseline dataset, containing a full-archive of tweets of random sampled users, which will represent an authentic user in our detection model.

Sampling baseline account tweets

This section is dedicated to describing the sampling of baseline users. [37] suggests starting the sampling process by clearly defining the sample population. Since our goal is to create a dataset of authentic Twitter accounts tweets, we want so sample from a population of verified authentic accounts. In practice, such sampling source is hard to find, as verifying account authenticity of online users is a labour-intensive task[36]. Instead, we have to settle for a reliable source of active Twitter accounts, namely the Twitter API, which exposes activities of both benign and malicious users. Nevertheless, since Twitter is continuously removing malicious accounts from their platform[10], we will assume that the majority of the active users on Twitter are authentic.

The Twitter API exposes a dedicated “sampled stream” endpoint for sampling of publicly available realtime tweets[38]. The endpoint delivers a 1% random sample of all live tweets on the platform. Furthermore, the tweets returned by the API can be enhanced with additional information about the user like their user id, creation date and follower count. This additional information is crucial for our purposes, as we are interested in profiling authentic users. The second and third phases of the sampling taxonomy from [37] deals with choosing a sampling frame and technique. By using the Twitter sampling API, we are implicitly defining the sample frame containing users who are active on Twitter in the period of the sample. The sampling technique used by the API is random[38]. Since the API returns single tweets, we have to extract the user id from the response and save it in order to query additional tweets for those users.

Sampling bias is a known problem in literature[37][39], and has to be considered in this case as well. By sampling users from a live stream, we are effectively excluding users which are not currently active due to working hours, time zones or abandoned accounts. This kind of problem is referred to as convenience sampling, where “the participants are selected in order of appearance according to their convenient accessibility”[39]. To tackle this issue, we have to expand our sampling frequency, and spread the sampling efforts during a longer period of time in order to make our sample time zone invariant. Unfortunately, we have no way of addressing the issue of abandoned accounts and must accept that these will not be represented in the sample of authentic accounts. Another limitation of the sampling process, is the requirement that the sampled users have tweet activity in the period of the spam-campaign. It is important that the activity of trolls and baseline users are from the same period as the usage patterns on Twitter can change over time due to updates introduced on the platform, new behavioural trends in the community and significant historical events[42]. This means that we have to disregard users without any activity in 2016, thereby discarding accounts that are created less than 5 years ago. We highlight that this additional age requirement might further skew the baseline sample but will not put further effort into investigating how and why due to time constraints. At last, in order to perform correlation analysis based on the contents of the tweets, we will make use of various techniques from the Natural Language Processing (NLP) domain. In order to compare tweet statistics derived by NLP methods, we require that the language of both account categories is the same. Since the language of the trolls is given (English), we will need to use purposive sampling[37] and disregard non-english speaking users.

Based on the above discussion we have derived the following requirements for the baseline dataset:

1. The users are active during the period of the spam campaign
2. The sampling must be conducted during a longer period of time to avoid convenience sampling
3. The features are comparable, ie. tweets are in English and the schema of the baseline dataset matches the trolls dataset

In order to meet the above requirements we will use the following sampling strategy:

1. **Distributed sampling**: The accounts are sampled continuously during a period of one week (05/03/2021 - 12/03/2021). Every hour of the day, 300 accounts are collected from the realtime streaming API. This approach increases the probability of the sample being time zone invariant. The result is a list of approx. 50.000 user id's.
2. **Remove duplicates**: It can happen that highly active users are sampled multiple times. We therefore remove duplicates from the user id list.
3. **Randomization of user list**: The list of users is randomly shuffled to remove any ordering related information.
4. **Querying tweets**: We consume user id's from the list and query the twitter API for all the tweets of the users in 2016. If the user have no tweets in that period, they are discarded from the sample. We repeat this query until we reach the quota on the number of tweets from the API which is 10.000.000.
5. **Checking language**: For each user, we check the language of their tweets. If more than 80% of the tweets are in english (determined by Python's 'langdetect' package), we keep the user in the dataset. Otherwise we discard the user.

After discarding the non-english users, the above approach yields a full archive tweet collection of 5.888.533 tweets dated from 01/01/2016 to 31/12/2016 for 1198 unique baseline accounts[59].

In order to fulfill requirement 3, we need the troll and the baseline schemas to match completely. Recall from the data description section, that the schema of the troll dataset also contained variables like the retweet flag, tweet publish date, number of follower and number of following. We want to enrich the baseline dataset with the same information to enable comparison of these features during analysis. Fetching the publish date and retweet flag is straightforward and can be done by a single query to the API for each tweet. The follower and following count in the troll dataset is historical, meaning that that the number is dated to the time of the tweet. Having the historical information, allows us to track the follower/following progression for troll users over time. In order to have matching data for the baseline accounts, we need to enhance each row (tweet) of the baseline dataset with the users follower and following count dated to the time when the tweet was posted. Since the Twitter API does not expose the historical number of a users follower and following count, we need another way to infer it. We can estimate the history of these number by computing an empirical average of users follower and following counts as a function of their age. Then, by knowing the age of a baseline user at the time of the tweet, we can then estimate their historical follower and following count by looking their age up in our function. In order to create such a lookup function, we query an additional sample of 10.000.000 users from Twitter together with information of their age and their follower and following count. Grouping the sampled users follower and following count by week and taking the average of each bin, will yield a lookup function which we can use to get an empirical value of followers and following given the users age. Such function is show in figure 3.

We can do simple lookup operations on the functions in figure 3 to get the historical follower and following counts. More precisely, having the publish_date of the tweet and the age of the baseline account we estimate the followers and following counts by the process in listing 1.

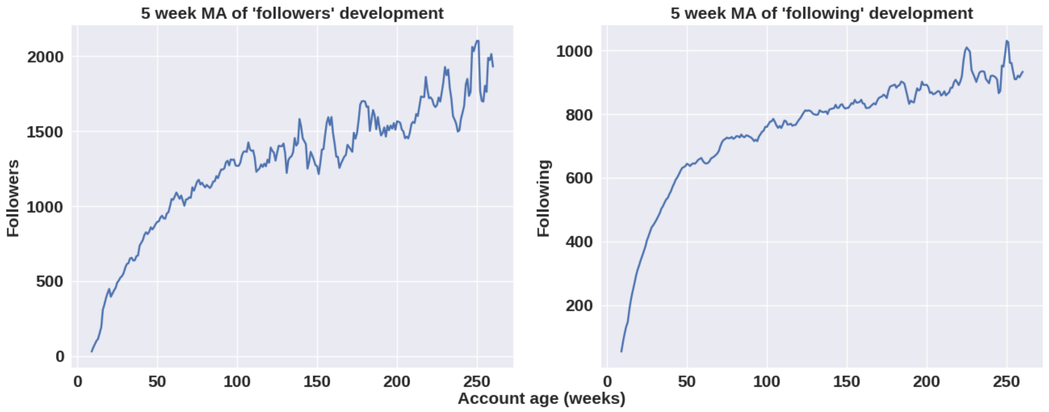

Figure 3: Development of followers and following of random Twitter accounts over time (n=10.000.000)

Listing 1: Inferring follower and following count

```
account_age_at_tweet = publish_date − account_creation_date
follower_count = empirical_followers_function(account_age_at_tweet)
following_count = empirical_following_function(account_age_at_tweet)
```

Such an operation is equivalent to as simple table lookup and can be applied to every row in the baseline dataset in linear time.

Having done this, we now have a baseline dataset with the same features as the troll dataset. The resulting baseline dataset is sampled almost randomly, with the exception that the sampled accounts are active, english-speaking and are at least 5 years old. Some of these exceptions fit well to the potential use-case for the detection model, as we can imagine that troll detection will be mostly applied to active accounts. The fact that the accounts are at least 5 years old and still active increase the chance of the accounts being legitimate, as Twitter already has fake account detection methods in place which remove inauthentic accounts over time. For instance, over 70 million fake Twitter accounts were removed during May and June 2018. In fact, the rate of account suspension on Twitter has more than doubled since the discovery of the IRA troll farm and Russia's alleged attempt to interfere with the US 2016 elections[10]. Consequently, all of the accounts from the troll dataset have also been removed from Twitter since the release of the data. This can in fact pose some extra challenges in the analysis phase, as we will see shortly.

Additional data features

In general, data gathered from a social network can be grouped in two categories; profile or content. Profile data includes information about the users profile, like their gender, age or status. On the other hand, content data is not directly describing the user. In our case content data are the tweets. In fact, many account classification models employ both content and profile data[11][12]. For instance, features like existence of a profile picture, text color of the profile and background color of the profile have been identified by [12] to be amongst top 5 most significant indicators of a fake account. As we do not have the profile information of the users and are not able to retrieve it for the deleted troll accounts, we will resort to NLP techniques in order to gather additional data features from the content.

It is a common practice to infer additional statistics from the existing data to strengthen the detection model. In the next section we will take a better look at the data and try to squeeze additional information on the Twitter accounts through correlation analysis and NLP. More precisely, we will be adding the following features to the troll and the baseline datasets:

- **top_30_troll_hashtag_hitrate**: The hitrate of users hashtags against the 30 most frequent hashtags used by troll accounts. The number is between 0 and 1. A value of 0.3 means that 30% of the hashtags used by the account are amongst the 30 most popular troll hashtags.

- **top_30_baseline_hashtag_hitrate**: The hitrate of users hashtags against the 30 most frequent hashtags used by baseline accounts. The number is between 0 and 1. A value of 0.3 means that 30% of the hashtags used by the account are amongst the 30 most popular baseline hashtags.

- **top_30_troll_word_hitrate**: The hitrate of users words (non-hashtag terms) against the 30 most frequent words used by troll accounts. The number is between 0 and 1. A value of 0.3 means that 30% of the words used by the account are amongst the 30 most popular troll words.

- **top_30_baseline_word_hitrate**: The hitrate of users words (non-hashtag terms) against the 30 most frequent words used by baseline accounts. The number is between 0 and 1. A value of 0.3 means that 30% of the words used by the account are amongst the 30 most popular baseline words.

- **Tweet sentiment**: The sentiment score is a value ranging -1 to 1 indicating how positive or negative a tweet is. Values closer to -1 indicate negative sentiment while values closer to 1 indicate positive sentiment. A sentiment score of 0 means that the text is neutral.

- **Activity variance**: The tweet activity variance is the standard deviation of tweet frequency per week. This number will be used to reason about account behaviour.

- **Retweet rate**: The retweet rate is the average of the boolean 'retweet' flags across all tweets. During the analysis, we will group this number by account type and by user.

- **Account lifespan**: The lifespan of a user is the number of days passed from the first to the last tweet in 2016.

- **Tweet subjectivity**: Tweet subjectivity is a number ranging from 0 to 1 indicating the subjectivity of the tweet text. In this context, subjectivity tells us something about the level of bias in the text due to personal opinions.

- **Follower/Following rate**: We compute the metric by simply dividing number of follower by number of following for each user.

If there is a correlation between account authenticity and any of the above features, we will likely improve our detection model by including these. Recall from the limitation section, that we are interested in correlation rather than causation as the typical use of machine learning models is not to explain causal effects of the data, but merely to label new data. Therefore, we will include features in the detection model based on simple correlation analysis and hypothesis testing.

Analysis

In this section, we will aim to identify features of the data which we can use in our detection model. The analysis will primarily focus on the content of the tweets and the behaviour of the users over time. The content analysis will give us insights into what topics were trending amongst fake and authentic users throughout 2016. The behaviour analysis will tell us something about the behaviour of the users, like their tweeting frequency and tweet timings. During the next sections, we will be testing the following hypotheses from which we will try to derive indicators of inauthentic and authentic accounts:

1. **”The tweets of troll users are more homogeneous than the tweets of baseline users”** - This hypothesis is based on the assumption that the malicious users are collaborating and therefore might post similar content. It has been shown in [5], that the users in the troll dataset are participators of the same spam campaign hosted by the ”Internet Research Agency” located in Saint Petersburg which reassures us that the assumption of collaboration holds.

2. **”The sentiment of users tweets participating in a political astroturfing campaign is more negative than for baseline users”** - This hypothesis is also supported by some of the findings in [6]. ’LeftTroll’ and ’RightTroll’ account categories aim to support the far-left and far-right wing movements in US. In order to support one side of the political spectrum, the accounts tend to write a lot of negative tweets about the opposition, thus producing more negative sentiment on average.

3. **”The activity of inauthentic users varies more than the activity of authentic users”** - It is fair to assume that political spam campaign participators have a specific agenda and thus may adjust their activity in order to fit that agenda. The assumption is that the activity of users participating in political astroturfing changes significantly during periods of important political events. We have already seen indications of this in the data description section. We will therefore be investigating the activity of the users and how it may change over time.

4. **”Spam campaign participators located in the same area have a similar hourly activity schedule”** - It is established in [5] that the trolls are employed by the IRA located in Sant Petersburg. Such a setup might imply that the trolls have a more unified activity throughout the day than baseline users.

5. **”Inauthentic users will have a higher retweet rate than authentic baseline users”** - It is assumed that the troll accounts will try to appear as normal users and try to share quality content in order to get more followers and expand their attack surface. By using retweets, a user can get quality content on their profile for ”free”, thereby potentially get easier access to more followers.

6. **”The lifespan of inauthentic accounts is shorter than the lifespan authentic accounts”** - Since the users are inauthentic and are driven by a specific goal or agenda, one can imagine that once that goal has been fulfilled, the accounts will not longer be used, resulting in a lower lifespan.

7. **”The tweets of troll accounts are more subjective than the tweets of the baseline accounts”** - It has been suggested in [6] that the tweets of the troll accounts are more subjective than authentic users.

8. **”The rate of followers and following is different between trolls and baseline accounts”** - It has been found by [6] that trolls try to nudge people to follow them and that they are more effective in obtaining followers than the average user. It can therefore be of interest to further examine the follower and following counts of trolls and baseline accounts.

When testing for correlations between numerical and binary variables, it is suggested to use Point-Biserial correlation score[25]. We will therefore use this metric to enhance the analysis with a more formal hypothesis testing approach. It is common heuristic that a p-value of less than 0.01 is highly significant and gives us enough confidence to reject the null-hypothesis[26]. We will therefore use a significance level $\alpha = 0.01$ to reject the null hypothesis, which assumes that there is no correlation between a data variable and the account type.

Hypothesis 1: Tweet similarities

In this section we will analyze the contents of the tweets and try to identify any interesting differences between trolls and baseline accounts. We calculate a word frequency dictionary of the tweets to get a quick insight into the contents. Figure 4 shows top 20 most frequent words for the troll and the authentic accounts. In the context of Twitter, it is common to use the hashtags symbol to link the tweet to other similar tweets containing the same hashtag. This helps to identify common topics (or campaigns) across multiple tweets. It is therefore also interesting to look at hashtag frequencies, which are shown in figure 5.

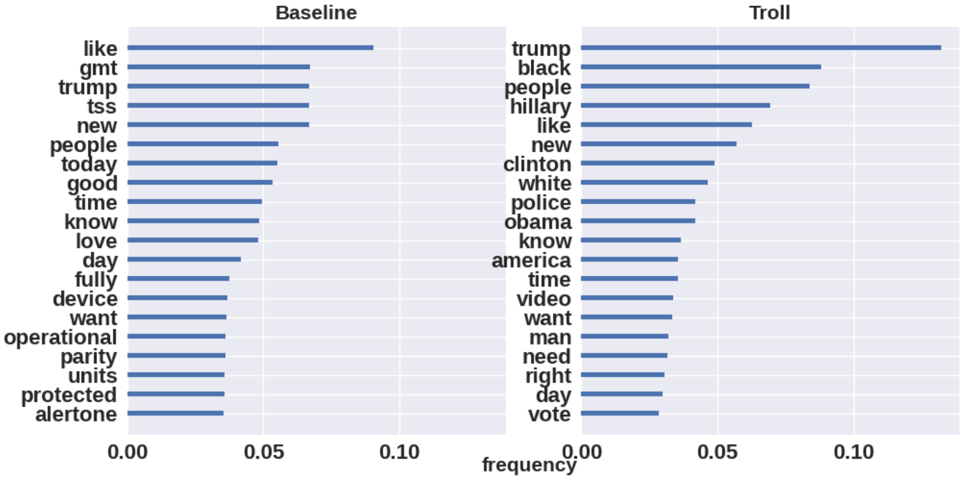

Figure 4: Top 20 word frequencies of baseline and troll accounts

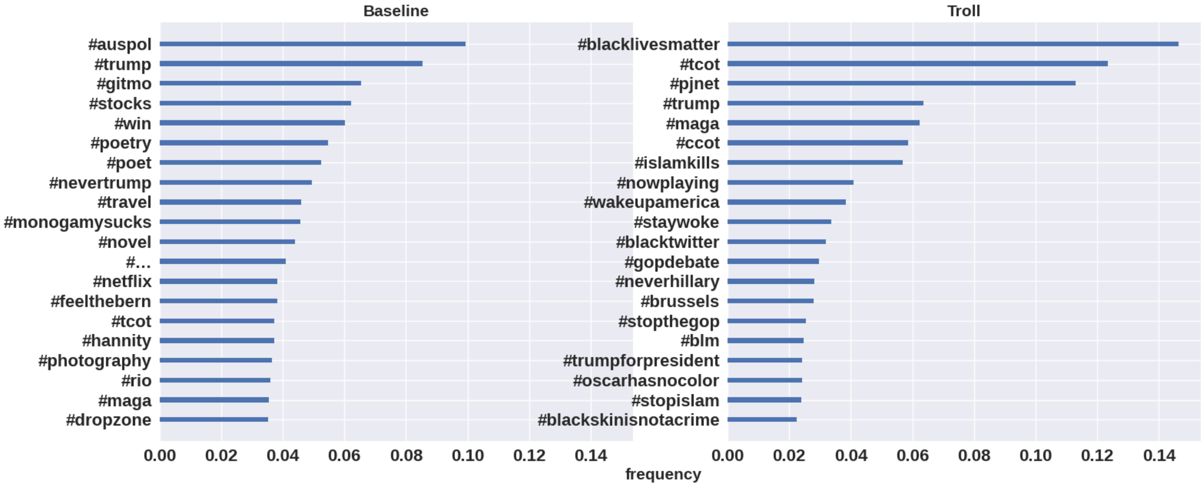

Figure 5: Top 20 hashtag frequencies of baseline and troll accounts

The tweets are normalized and prepossessed before calculating the dictionaries. The preprocessing pipeline is:

1. **Lowercasing** - The semantics of words rarely change by removing the casing. Although in some special cases, we can only tell the difference between the words when casing is included, like US (country) and us (pronoun). Looking closer at the tweet word and hashtag vocabularies, we find that only few of these cases exist.

2. **Stopword removal** - It is reasonable to remove words like ”it”, ”they” or ”the” as these words rarely contain any meaningful semantics for our analysis. The stopwords typically have a much higher frequency, thus cluttering our dictionaries of words and hashtags. The stopword removal is done by using the gensim stopword list.

3. **Punctuation removal** - Punctuation can often confuse our dictionary as it will separate instances like ”word,” and ”word” in the dictionary. To avoid this, we chose to remove punctuations. Special care has to be taken when doing so, as in some cases, we need the punctuations to preserve important semantics. In our particular case, we want to keep the hashtag symbol ”#”, as it enables us to keep information about hashtags and perform further analysis. The ’@’ is another important symbol on Twitter, as it is typically used to reply or quote a user by prepending ’@’ in front of the username. However, as we will not be looking into user-related tags, we will not preserve this symbol.

A lot more can be done to further process the tweets. Some additional techniques frequently used in NLP are stemming, which is the process of reducing the words to its ”stem” or lemmatization which is the process of mapping words of similar meaning to one single word[45]. Both techniques can help to achieve a more normalized dictionary. Stop-word removal can also be performed based on word frequency cut-offs (removing most popular and least popular words). At last we can translate all words into pre-trained word embeddings, achieving a numerical vector representation of each word. This allows us to do numerical computations on words like distance between words or retrieving top n similar words, thus doing vector-space computations on the word semantics[13]. We chose to avoid any additional prepossessing techniques due to time constraints and scope of the project.

Looking at figures 4 and 5, we can get an insight into most popular tweet trends for troll and authentic accounts. The dictionaries are separated into words and hashtags to distinguish between common language terms and user annotated hashtags. Looking at figure 4, we see that the words used by troll accounts are more political in nature than for authentic accounts, which seem to have a more generalized vocabulary. It may also seem that the popular troll account words have a slightly more negative and serious sentiment, when looking at examples like , ’police’, ’white’ and ’black’ while the authentic account words seem to have a more positive sentiment with words like ’like’, ’thanks’, ’good’ and ’love’. It is hard to determine this without a formal sentiment analysis, but the idea that trolls might have a more negative sentiment fits well with the findings from [5]. We will be looking further into sentiment analysis later.

As described in the previous section, we calculate the hashtag hit-rate against the top-30 most popular hashtags. This metric can tell us something about hashtag similarities between a particular user and the categories of troll and baseline accounts. We suspect that spam campaign collaborators will have a tendency to share the same hashtags, while authentic users will be more heterogeneous in their posted content. Table 1 shows the hashtag hit-rate for trolls and authentic users against top-30 dictionaries of trolls and authentic users. The table also reports the word hit-rate for both account categories against the top-30 word dictionaries. We observe that the average hit-rate between troll account hashtags and the top-30 troll hashtag dictionary is 0.38, meaning that on average, 38% of all hashtags used by trolls are amongst the top-30 most popular troll hashtags. At the same time, the hashtag hit-rate of baseline accounts against the top-30 baseline hashtag dictionary is 0.02, meaning that only 2% of the hashtags used by the baseline accounts are amongst the top-30 most popular baseline hashtags. This supports our hypothesis that the troll accounts are posting more heterogeneous content than the baseline, if the content similarity is measured by the hashtags. We do not observe any significant differences in the word hit-rates for the account categories and the top-30 word dictionaries. This means that on average, trolls are not more homogeneous in their tweets than baseline users, if the similarity is measured by words only. These findings indicate that the troll collaboration is evident from the hashtags that they tweet. At last we observe that the troll and baseline hashtag hit-rates against the top-30 troll hashtag dictionary are 38% and 1% respectively, a more significant difference compared to the other hit-rates.

Dictionary Account type	Troll hashtag	Baseline hashtag	Troll word	Baseline word
Troll	0.38	0.04	0.08	0.06
Baseline	0.01	0.02	0.05	0.07

Table 1: Mean hit-rates for words and hashtags against the ”top-30” word and hashtag dictionaries grouped by account types
(Purple marks the strongest indicator of a troll/authentic account)

The Point-biserial correlation score for the hashtag-hitrate of trolls and authentic accounts against the top-30 troll hashtag dictionary is 0.67 with a p-value < 0.01. We conclude that there is a highly significant correlation between the hashtag-hitrate and account type. We will therefore reject the null hypothesis, and use the hit-rate against the top-30 troll dictionary as an indicator of collaboration in our detection model.

Hypothesis 2: Tweet sentiment

Sentiment analysis has been shown valuable in opinion mining and analysis of unstructured textual data[43]. It can help us get further insights into the tweet contents of trolls and baseline users. While the sentiment score is not part of the original data, it is not hard to compute the sentiment of the tweets, given many of the pre-trained classifiers that exist today. One of the classifiers publicly available, VADER (Valence Aware Dictionary and sEntiment Reasoner)[14], is made and published by NLTK. It is trained and tuned on data from social media, making in particularly usable for our purposes. VADER is a rule-based classifier which applies a set of rules to a human-generated sentiment lexicon in order to derive sentiment scores for pieces of text. This means that the classifier doesn’t have to be trained or use an excessive amount of computing power to predict sentiment scores. The prediction engine requires the existence of a sentiment lexicon, which can be easily downloaded through the VADER python library.

We use VADER to label each tweet with a sentiment score. A sentiment score is a number ranging from -1 to 1, and indicates the sentiment of a piece of text using a numerical value. Values closer to -1 indicate a negative sentiment, while numbers close to 1 indicate a positive sentiment. A value of exactly 0 indicates a neutral sentiment. The average sentiment scores of troll and baseline accounts are -0.004 and 0.074 respectively, a difference of 0.078. Furthermore, looking at how the sentiment develops week-by-week (figure 6), we observe a much higher variability amongst the trolls than the baseline, with the standard deviations being 0.034 and 0.009 for trolls and baseline accounts respectively. The standard deviation is computed on aggregated weekly data.

We see some similarities between the sentiment developments of trolls and authentic accounts; most remarkably week 28 where the average sentiment drops significantly for both categories. This drop is most likely caused by the Nice truck attack which happened July 17th, resulting in death of over 80 people and over 400 injured[44]. The attack was very largely covered, especially by the western media. This indicates that the baseline accounts indeed react upon world events, but not in the same manner as the troll accounts, as the variability is much higher for trolls than for baseline users. The higher variability of the sentiment scores for trolls is also supported by qualitative analysis of the Tweets. We see that trolls mainly use 2 strategies in their tweets; pushing agenda of their supported party which results in a positive sentiment or speaking ill about the opposite party, resulting in a more negative sentiment. Some negative examples are:

- *’drapermark37 not really. Just one more brainless Trump supporter’*
- *’Hillary couldnt prevent nor can she solve her own problems, how is she going to solve Americas?? #hillarysemail’*
- *’Refugees are not welcome, that my view! #IslamKills #StopIslam’*

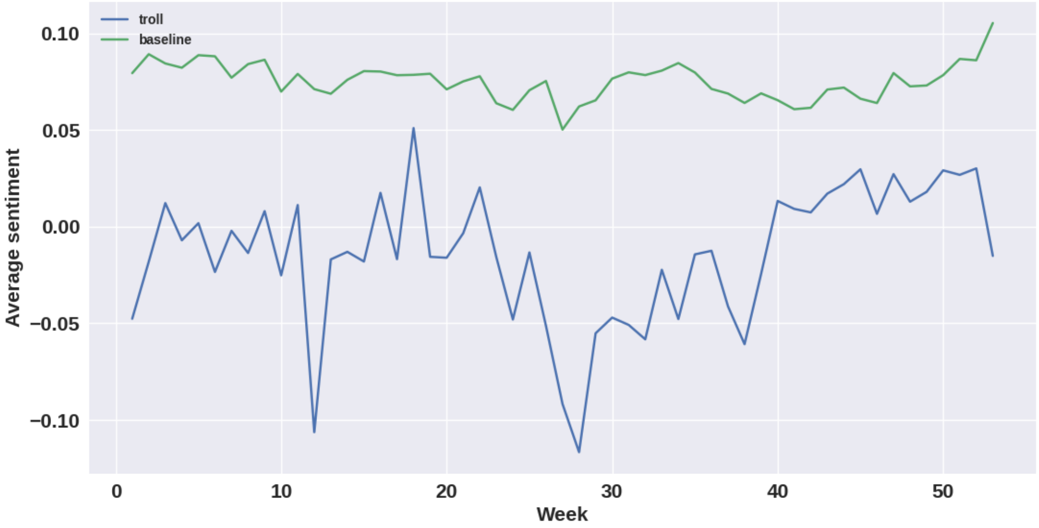

Figure 6: Sentiment development over time for trolls and authentic accounts

while some of the positives are:

- *’According to Politico, Caroline Giuliani, the daughter of former NY Mayor Rudy Giuliani, supports #HillaryClinton this election’*
- *’Patriots Quarterback, Tom Brady, was a real-life Patriot today when he endorsed Donald Trump. #MAGA #ImVotingBecause #TrumpForPresident’*
- *’Praise black women, because it must have taken so much courage and strength to stay calm and record that video. #BlackSkinIsNotACrime’*

The findings about the differences in the sentiment scores of trolls and baseline accounts are also supported by [6]. The Point-biserial correlation score for the sentiment of trolls and authentic accounts is -0.31 with a p-value < 0.01, which means there is a highly significant correlation between the sentiment score and the account type. We therefore chose to reject the null and conclude that the sentiment scores of users participating in political astroturfing is on average lower than for baseline users.

Hypothesis 3: Change in behaviour

As demonstrated in data desciption section, troll accounts tend to react upon important political events, resulting in big spikes of tweet activity. We also saw during sentiment analysis, how the sentiment of trolls is more variable than for baseline users. This motivates a further investigation into the variability of activity for trolls and baseline accounts. Before we start looking at changes in tweet frequencies over time, we have to address a general issue observed with the average tweets per user. There appears to be a skew in the way that the tweets are distributed amongst users. A recent study on the differences between Democratic and Republican Twitter users in the U.S. posted by Pew Research Center, found that 92% of all tweets since November 2019 were produced by only 10% of the users[15]. This indicates that it is the minority of the Twitter users that account for majority of the tweets. A similar skew in distribution of tweets is also observed in our baseline dataset. Figure 7 shows the CDF of tweets posted by user. In our sample, we see that 70% of all tweets are posted by 10% of all users from the dataset. At the same time, we see the effects of this skew when plotting the average and the median tweet frequency per user over time. By using the average, we get that a user tweets around 130 times a week vs 30 tweets per week for the median. Based on this observation, we acknowledge the existence of extreme outliers in our dataset. Since the median is a more robust metric in the presence of outliers, we will use the median instead of the average in the following analysis.

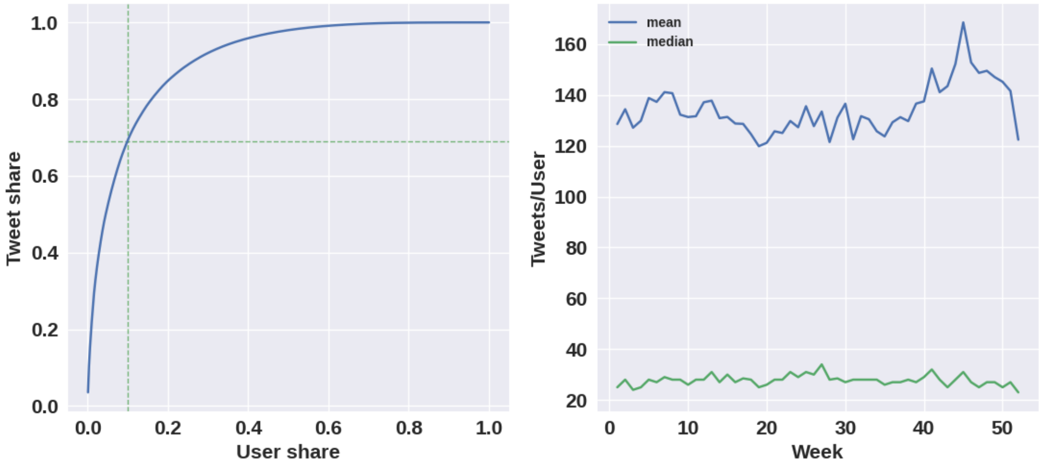

Figure 7: Illustrating the skew in tweet distribution amongst baseline users

We have already seen that the variability of the sentiment score is higher for troll users. This change in behaviour can be partly attributed to the polarized nature of the tweets that the trolls post, examples of which were provided in the previous section. We can hypothesize that not only does the sentiment of the trolls change more over time, but also the frequency of their activity. The main reason being that spam campaign collaborators will ultimately concentrate their activity around specific events of interest, thus deviating from the baseline. Figure 8 shows a significant change in user activity in the months prior to the US presidential election, which was in November 2016. Also a significant rise in activity is seen already in October, where Donald Trump experienced his infamous 'Pussygate' scandal, the Podesta emails got leaked and the US presidential debate was hosted. A smaller rise in activity is also observed for baseline accounts during the same time period.

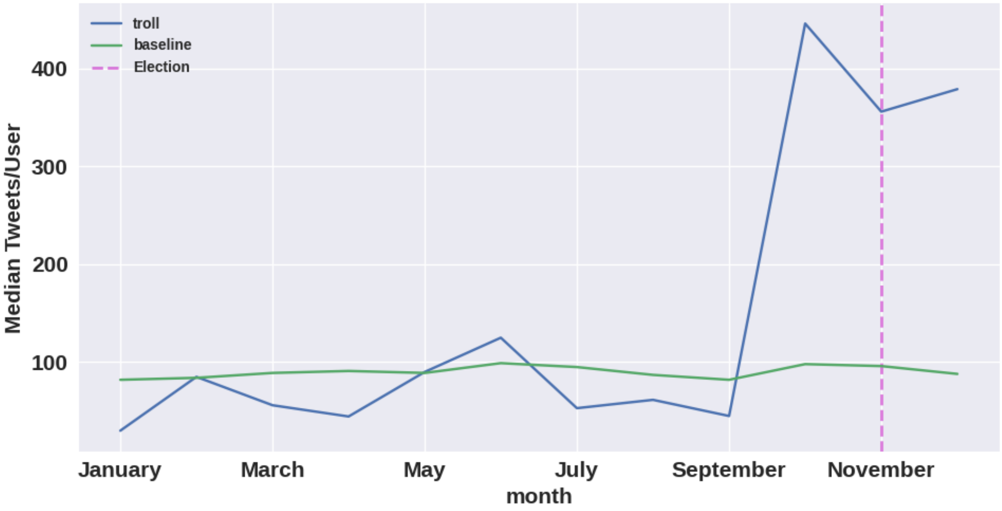

Figure 8: Median tweets per user over time

As with the sentiment scores, we see a more aggressive change in activity during important political events. The standard deviation of the median tweet frequencies per week is 38.0 for troll accounts and 2.8 for the baseline. The number of users in the datasets is 489 and 1198 for trolls and baseline accounts respectively. We realize that a smaller sample size will have a tendency for greater fluctuations. Nevertheless, if we downsample the baseline dataset to 489 users, we still observe a standard deviation of 3.9 for the median tweet frequencies per user per week, which is still much smaller than for the troll dataset. This increased variability in the activity of troll accounts is also an indicator that the trolls are more selective about their use of Twitter, having a greater activity on ”important” days, while the baseline accounts are less selective about the specific timing in which the chose to engage in Twitter. At last, it is important to consider that the troll accounts are sampled from a distribution of known political active accounts, identified to be trolls, while the baseline dataset is random sampled. This can also explain why the baseline is more homogeneous in their tweet activity. Nevertheless, we still assume that the average politically engaged authentic user will have a more heterogeneous tweet activity than the average troll account. Also, we assume that a potential troll detection model will be applied to the entire population of users and not only those who are politically active.

Unfortunately, when digging deeper into the analysis we discover certain limitations in including this variable in the model. When grouping the tweet frequency by user, we find that there are not enough activity data-points in order to calculate a proper variability score. For instance, 21% of the troll users have an activity lifetime of only 1 week, resulting in a 0 standard deviation in the tweet frequency per week for these users. Because many of the trolls have too few activity data-points when aggregated on user-level, this metric is too unstable to include in our model. Even if evidence might suggest that we can reject the null, we chose to not use the tweet frequency variability in our particular case because of the short lifespan of the trolls. Nevertheless, this metric can be of interest in cases where the user activity has more data-points.

Hypothesis 4: Hourly activity similarities

In our particular case, the spam-campaign participators are located in the same time-zone as the spam-campaign is an ”industrialized” operation where most of the spammers are employees of the same organization. With this type of spam-campaign, one might hypothesize that the hourly activity of troll accounts throughout the day will be similar. However, it has been found difficult to test this hypothesis, due to lack of geolocation information in the Twitter API. The biggest problem is that geolocation data is only available if the user chooses to disclose it on their account. Another part of the problem is, that geolocation is user-provided and can therefore not be completely trusted. Generally, whenever the data is provided by the user, one must consider data poisoning attacks, which we will look at later in the project. The lack of geolocation data makes it hard to estimate time-zones of users, as all of the tweets publish_date are normalized to UTC. Having no time-zones, we cannot perform sensible correlation analysis on the hourly activities of the users and can therefore cannot reject the null hypothesis.

Hypothesis 5: Retweet rate

It has been suggested in literature, that troll accounts employ various strategies in order to gain popularity in order to make their disinformation efforts more efficient. An interesting behaviour noted in [6], is that troll users are actively nudging their audience in order to gain more followers and thereby a bigger attack surface. Getting a follower on Twitter implies that the follower wishes to add the content of a particular account to their feed. In order to increase the likelihood of getting followers, the trolls must therefore produce higher quality content, thereby catching the interests of more users. One ”cheap” way of producing high-quality content on Twitter, is simply to retweet high-quality content of other users. One can therefore hypothesise, that a high-retweet frequency can be an addition indicator for a troll account. The average retweet of troll accounts is 40% while it is 37% for the baseline. The Point-biserial correlation score for the retweet count of trolls and authentic accounts is 0.03 with a p-value 0.11. There is only a small and statistically insignificant correlation between the retweet rate and the account type. One can argue that the baseline accounts likewise have a desire to increase their follower count in order to reach a broader audience, thus using re-tweeting as one of the strategies.

Therefore, we cannot reject the null and choose to conclude that the retweet rate of users participating in political astroturfing is on average the same for both authentic and inauthentic accounts.

Hypothesis 6: Account lifespan

We have discussed earlier, that the typical troll account has a specific agenda. In this case, the agenda is to spread political disinformation and polarize the public opinion on U.S. politics. Once the troll has fulfilled its purpose, we can imagine that the account will be discarded by the operator. In some cases it has also been discovered that the accounts in fact switch operators completely and that the behaviour and content of the account changes significantly[6]. Figure 9 (left) show the CDF over the lifespan of the accounts measured in days. The account lifespan is calculated as the difference between the first and the last recorded tweet activity measured in days. The figure tells us that a large majority of the troll accounts have a short lifespan throughout 2016, while the majority of baseline accounts are active most of the year. We see that 60% of all baseline accounts are active more than 350 days of the year, close to a full-year activity while only 20% of all troll accounts have the same lifespan. We see also that 50% of the troll accounts have a lifespan shorter than 100 days throughout the year while the same holds for only 10% of the baseline accounts. The lifespan of accounts seems to be a significant separator for troll and baseline users. These differences support our hypothesis that troll accounts in general are short-lived, compared to normal accounts which use Twitter in a stable and regular way. The fact that trolls account are not operated by legitimate users makes it more likely that the accounts eventually will be abandoned. Even though research has shown that these type of troll accounts try to appear legitimate by having real humans operators, we still see that their lifespan is irregular, indicating that even the human operators don't stick to the account for as long as authentic users. Also, because the trolls have a specific political agenda on Twitter, one can imagine that some of the accounts are created only for the purposes of specific political events and will be discarded later. This is all in line with our previous finding and also the tendencies observed in [5] and [6] regarding the selective activity patterns of the trolls.

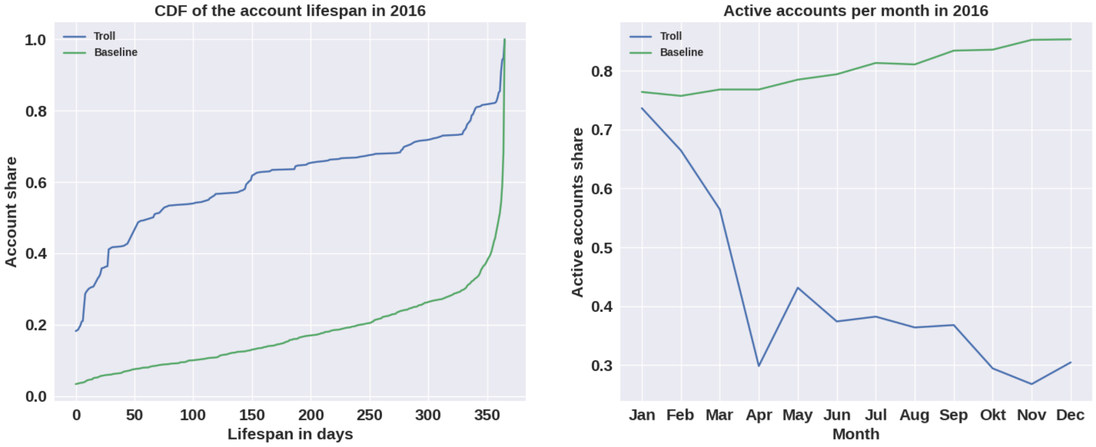

Figure 9: Activity and lifespan of troll and baseline accounts during 2016

Figure 9 (right) shows the fraction of simultaneously active accounts per months for trolls and the baseline. This plot shows the same tendency of the account activity. The troll activity is not as stable throughout the year as the baseline, which is also a tendency we saw during analysis of hypothesis 3. Here we observe that on average, 80% of the baseline users are active all months of the year, while the same holds for only 40% of the trolls. Interestingly enough, we see that the number of active accounts is all-time low during the months around the US election. This does not necessarily mean that that the trolls are downsizing their efforts in meddling with the US politics, but rather that their attempt to influence public opinion is distributed throughout the whole year. The small share of users active during the election months can also be a sign of some coordinated attempt to involve only certain users, specially suited for the purpose. But without a more formal analysis, which we will avoid in this project, we cannot say anything more concrete about the small user share during the election months.

At last we have to address that the lifespan variable limits our detection model to inactive users. We cannot calculate the lifespan of users which are still alive, as we do not know when they will become inactive. This is a rather significant limitation, as inactive users are not posting any new content on Twitter. However, their content does still exist, and can also propagate "post-mortem" through quotes and retweets. Thus, removing inactive users is still a priority, even though it will have a lesser impact on reducing the disinformation campaign.

Based on the analysis in this section, we see that there is a difference between the troll and baseline account activity and lifespan. The Point-biserial correlation score for the lifespan of trolls and authentic accounts is -0.39 with a p-value < 0.01. Therefore, there is a statistically significant correlation between the lifespan of an account and the account type. We reject the null hypothesis and will use the lifespan variable in the detection model.

Hypothesis 7: Tweet subjectivity

It has been suggested in [6], that the tweets of troll accounts are more subjective than the baseline used in the paper. A subjectivity score is an indicator of how biased a piece of information is towards personal opinions. It is reasonable to assume that troll accounts who try to push a certain political agenda, will also have a high subjectivity score. Like with sentiment analysis, we can find pre-trained classifiers that given a piece of text will estimate a subjectivity score. The classifier chosen for this particular purpose is 'TextBlob' library for Python. We apply the classifier to each tweet in the dataset, yielding a score in the range $[0, 1]$, where values closer to 0 means less subjectivity, while values closer to 1 means more subjective content. Figure 10 shows the distribution of subjectivity scores for troll and baseline accounts. We see that the distributions are closely aligned, but that the mean of the troll subjectivity scores is actually lower than for authentic accounts. The mean subjectivity score of troll accounts is 0.28 while it is 0.32 for baseline accounts, suggests a small difference across the account categories. The Point-biserial correlation for the subjectivity score of trolls and authentic accounts is -0.16 with a p-value < 0.01. Therefore, there is a statistically significant small correlation between the subjectivity score and the account type. These findings support our assumption that the subjectivity scores across account categories are different, but contradicts the hypothesis which states that the subjectivity is higher for trolls. Nevertheless, we can still reject the null and use the subjectivity score in our model.

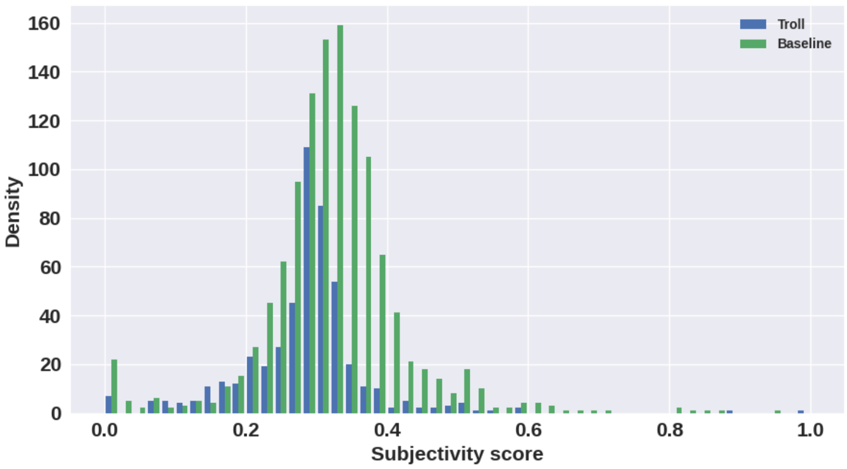

Figure 10: Distributions of subjectivity scores for troll and baseline accounts

Hypothesis 8: Follower/following rate

The way social networks grow on Twitter is through users following each other. Increasing the amount of followers is a great way to broaden the reach of the tweets. This number can be hard to increase outside of the pre-existing circle of friends, as it requires that people wish to see the tweets of a particular account, without having a personal relationship to the user. Therefore, accounts of celebrities, big companies or influencers will typically have a higher follower/following ratio's than average users. Increasing the "following" count is more straight-forward, as this only requires the user to follow a lot of accounts, without them having to follow in return. A user having a high follower/following ratio implies that a lot of people are interested in the content that the users posts, thus indicating that it is an account with a high level of quality. We are aware that the definition of "quality content" is subjective and governed by the users, but we assume that an account having a lot of followers is on average more likely to be a quality account, than an account having few followers.

With this in mind, we will now take a look at the follower and following rates of trolls and baseline users. Figure 11 shows the distributions of the follower/following ratios for trolls and baseline accounts together with a benchmark of their means. We observe that the average troll has fewer followers than a baseline account and that the following count of trolls is bigger than the baseline. This suggests that baseline accounts are more popular and might post higher quality content than trolls. Furthermore, the distributions of the follower/following ratios for trolls and baseline accounts are almost non-overlapping, indicating that this variable might help the separation of account classes in our detection model. It has been suggested in [6], that one of the strategies that the trolls use to gain more followers, is to follow other accounts in the hope of getting a "follow" back. In order for the strategy to succeed, it requires to follow a lot of accounts first, thus resulting in a high "following" count and thereby decreasing the follower/following ratio. It is therefore not surprising that this ratio is lower for troll accounts than the baseline.

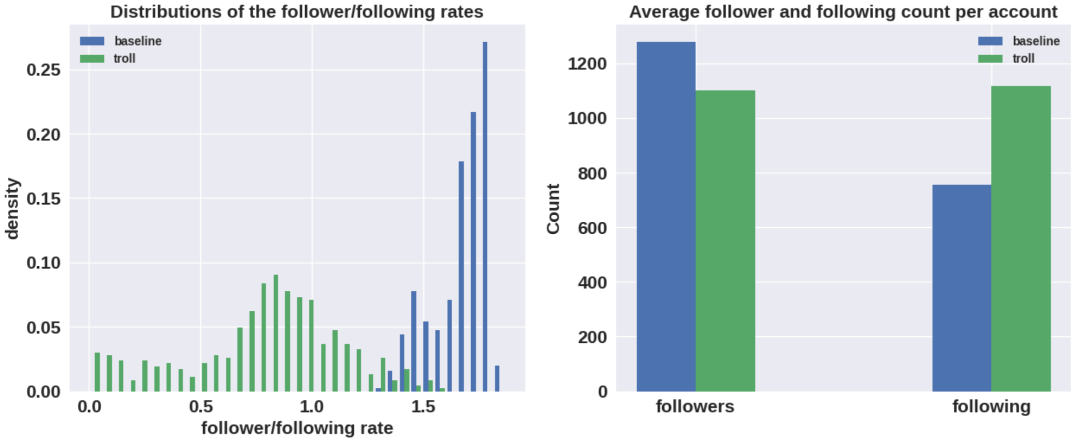

Figure 11: Differences in followers and following between account categories

At last, we have to address the origin of the follower and following counts of baseline users. Recall that in the first section, we estimated the follower and following counts from another random sample of approx. 10 million users. Based on this sample, we calculated functions over average numbers of users age and their followers and following count. At last we applied the function to the "time of tweet" variable of the baseline users to infer the follower/following count at the time of each tweet. While this approach provides us with some reasonable estimates, we have to keep in mind that these are not the real follower and following counts for the baseline, but an estimated average from another sample. This approach can have an impact on the distributions of the follower/following ratios of baseline accounts, as we observed that the distribution for baseline accounts is more narrow than the distribution of the ratios for trolls. Having the actual follower and following counts for the baseline would most likely yield a distribution with more variance. Due to different sampling approaches, the difference in the follower/following ratios between trolls and authentic users is excessively large. Consequently, for the sake of correctness, we will not use this variable in the model. Instead we choose to highlight that this variable can be important in other cases where the follower and following counts are sampled from exactly the same distribution for both account categories.

Hypothesis summary

During the last few sections, we have been looking into 8 different hypothesis about the trolls and the baseline accounts. During the analysis, we have been deriving interesting indicators of inauthentic accounts. We were able to reject the null for 4 out of 8 hypothesis. This means that we have 4 variables we will use in our model. We provide a short summary of the tested hypothesis:

1. **Hashtag similarity:** Null rejected. Comparing hashtag-hitrates against the top-30 troll hashtags, there is a significant diffference in the hitrate for trolls and baseline accounts.

2. **Sentiment difference:** Null rejected. The average sentiment score of the troll users was shown to be lower than the baseline.

3. **Activity variance:** Could not test due to missing data points when aggregating activity on the account-level.

4. **Hourly activity similarities:** Could not test due to missing geo-location data on accounts, making it hard to derive a timezone and make a proper aggregation of tweet activity per hour of day.

5. **Retweet rate:** Could not reject the null.

6. **Account lifespan**: Null rejected. The troll users was shown to have a much shorter lifespan on average than the baseline.

7. **Subjectivity:** Null rejected. There was a small significant correlation between the subjectivity score and the account type.

8. **Follower/following rate:** Could not test properly due to sampling issues. The follower and following counts were shown to be very different between trolls and authentic accounts. This was mainly contributed to the different ways that the variable has been computed for trolls and baseline accounts. For the sake of correctness, we choose not to include the variable in the detection model.

Model training and evaluation

During this section, we will use our results from the analysis to construct a machine learning model capable of automatically distinguishing between trolls and authentic accounts. There is plenty of research showing that a successful detection model is highly reliant on the quality of the input data[46][47][48]. That is why we have dedicated time and effort to find good quality features which we can extract from our dataset and feed to our detection model. Following the analysis, we calculate the hashtag-hitrate, mean sentiment score, account lifespan and subjectivity scores for each user in the dataset. This process yields an aggregated dataset of 1687 users, of which there are 1198 baseline users and 489 trolls, each having the 4 features derived from the analysis[60]. This is the data we will use for model training and evaluation.

Before training the models, we need to prepare the data and make some considerations about the training and evaluation process in general. We will divide our training and evaluation into following steps:

1. Split the data
2. Apply feature scaling
3. Select models for training
4. Create a grid of parameters for parameter-tuning
5. Train the models with cross-validation
6. Evaluate the models

We will describe and motivate each step in the following sections.

Data splitting

The typical ML process starts by splitting up the data in training and test sets[50]. This is done to avoid any bias during model evaluation. Naturally, evaluating the model against data which was used during training will yield an artificially better performance, and the results will not be representative of the performance we can expect once the model is deployed in the real world. Evaluating the model against a hold-out test set, allows us to get an unbiased performance estimate of the classifier. There exists many heuristic and best practices on how to split the data, but unfortunately there is no general approach that always works best[51][52]. We choose to do a simple 50/50 randomized split on the troll and the baseline dataset. This results in a training set of 843 accounts, of which there are 599 and 244 baseline and troll accounts respectively. The test set contains 844 accounts, of which there are 599 and 245 baseline and troll accounts respectively. Since we will be evaluating multiple ML models, we need separate splits for model benchmarking and for estimating the performance of the final model. Therefore, we apply a randomized 50/50 split on the test data, partitioning it into a test and a validation set. The validation set will be used to benchmark the different classifiers and the test set will be used to estimate the performance of the classifier with the best benchmark results. Note that since we are splitting the troll and the baseline data separately, we are effectively performing a stratified split, which ensures that each class of accounts are represented in all of the splits.

Feature scaling

Feature scaling is a common discipline in ML and can in fact improve the performance of some ML models significantly[49]. Reason being that some models are sensitive to the magnitude of the data values. For instance, any model which uses a distance metric between vectors, will be affected if any feature in the vector is scaled. If one feature has a significantly greater range than the rest, it will tend to govern the distance value and thereby also the position of the "separation line" in the model. Therefore it is important to apply feature scaling, when the ranges of the features are significantly different. Calculating the ranges of our features in the training set we get that:

$$h\_hitrate{\in}[0,1] \qquad sentiment{\in}[-0.68,0.87] \qquad lifespan{\in}[0,365] \qquad subjectivity{\in}[0,0.88]$$

We observe that the lifespan has a much greater range than the other features and will therefore dominate the detection model. We therefore choose to apply feature scaling. There are a number of ways to scale the features. We chose to use feature standardization which scales the features to 0 mean and unit variance. This makes them comparable and normalizes the ranges, thus making it easier for the model to obtain a good classification during training. As a small experiment, we try running the training with and without feature scaling. The resulting training scores have a difference of about 5-10 percentage points across different performance metrics, illustrating the significance of feature scaling. We apply the scaling to the training set only. Then we use the weights obtained from the scaling of the training set and apply these weights to the test set, such that these are scaled according to the ranges of the training data. After the feature standardization, the ranges become:

$$h\_hitrate{\in}[-0.51,3,49] \qquad sentiment{\in}[-4.84,5.04] \qquad lifespan{\in}[-1.82,0.76] \qquad subjectivity{\in}[-3.30,6.08]$$

After this standardization, any significant differences in the resulting ranges of the features, could be explained by a big number of outliers in some of the variables. This doesn't seem to be the case with our variables. Having the standardized data and more comparable ranges, we are ready to move on to model selection.

Model selection

Choosing a ML model to use is not a trivial task, as most models behave differently depending on the setting. We also have to distinguish between supervised and unsupervised ML models. Ultimately, we want to obtain a a yes/no answer to the question "Is this account a troll?". Or more formally: having a known subset of spam-campaign collaborators, can we use machine learning to find the unknown spammers. Thus we are dealing with a binary classification problem within the class of supervised learning, and are limited to ML models which are capable of solving this task. Luckily for us, there are plenty to choose from. Rather than using formal model selection methods or brute-force techniques, we will employ a more heuristic approach, by simply choosing 3 models which seem to be prevalent in literature. The models are also sufficiently different from one another, both by the type of the domain in which they are typically used but also in their learning algorithms. The 3 classifiers we will use are scikit-learn implementations of K-nearest-neighbor, Support Vector Machine and Random Forest Classifier.

K-nearest-neighbor classifier

One of the models we chose is from the class of clustering algorithms. While many of the clustering algorithms are unsupervised and are better suited for data mining purposes, some of them can be used for supervised classification. When we train the KNN classifier in a supervised setting, it simply "remembers" all of the training data points and their corresponding labels. During the prediction phase, it simply calculates the distance between the new point x and all of the "remembered" points. The algorithm chooses the k points nearest to x and takes the majority vote amongst the labelings of those k points. In scikit-learns's implementation, the model uses euclidean distance, but other distance measures can be used as well. One of the parameters in the KNN model are the weights of the k majority voters. Weights can be applied such that the points with the smallest distance to x are given more weight than the points which are further away. Another parameter in this model is the k number of points.

Since the training phase differs somewhat from the traditional models, it can be discussed if this type of model, when used in a supervised setting, is in fact learning anything. Nevertheless, it has proven to have good results in literature[19][53] and since we want to try out different types of models, KNN classifier seems to be a suitable choice.

Support vector machine

SVM is a completely different model, utilizing support vectors to find a suitable separator for the data. At its core, it is a linear separator, which creates a line to separate the data into classes. However the technique it uses to separate the data ensures that the line is a best possible separator, by maximizing the distance from the separator line to the closest points between each class (the support vectors). As a result of this optimal separation, the prediction score is often much better than that of a simple perceptron. SVM's can also employ kernels to transform the data, which allows them to work in non-linear space. The parameters subject to tuning in SVM's is the regularization parameter C, the kernel coefficient parameter γ and the kernel type parameter.

Random forest classifier

The random forest classifier is yet another type of model which uses randomized decision trees in order to yield predictions. Each decision tree operates on a random subset of the data and also a subset of the data features. To make a prediction, the model takes the majority vote of all the trained classifiers. The randomization helps us to avoid over-fitting and achieve a better generalization. The random forest classifier also differs from the usual perceptron by the way it creates separation in the data. Since the classification is done by an ensemble of trees, our data plane will be fragmented by multiple hyper-planes, thus yielding a non-linear separation. An ensemble classifier has numerous parameters. For simplicity, we will use the default configurable parameter given by the scikit-learn implementation. The only parameter we will consider is the number of estimators (decision trees) that we want to include in the ensemble.

Parameter tuning and scoring

Parameter tuning is a common technique to use during training. During parameter tuning, we continuously tweak the parameters of the model in order to find the best results. It is also sometimes called "model selection", to emphasize the fact that changing the parameters results in a new distinct model, providing different results. A typical way to do parameter tuning is the grid-search approach[54]. It works by creating a grid of size $\prod_{p\in m} v(p)$ where m is the set of parameters of the model subject to training and $v(p)$ is the number of values that parameter p has in the grid. In our case, we work with 3 different grids, one for each model. There exist a number of heuristics we can follow in order to chose a suitable grid. For the sake of brevity, we will use parameter values loosely inspired by [55]. The parameter grids are reported in table 3. As we can imagine, using parameter grids during training takes a toll on the speed performance. The same is the case when trying out different models. More specifically, if the time it takes to train a model m is t_m, then the running time of the whole training is:

KNN	
neighbors	weights
5	Uniform
10	Distance
30	
50	
100	
200	

RF		
estimators	depth	features
25	10	auto
50	20	sqrt
100	40	
200	80	
400	None	

SVM		
C	γ	kernel
0.1	0.001	rbf
1	0.01	poly
10	0.1	sigmoid
100	1	

Table 2: Grids used during parameter tuning

$$\sum_{m\in M} t_m \cdot \prod_{p\in m} v(p)$$

Where M is the set of all models we want to test, p denotes a parameter of model m and $v(p)$ is the number of different values that the parameter p has in the grid. This number is quite large and grows exponentially with addition of extra tuning parameters. By contrast, taking a single model and a pre-defined set of parameters yields a training time of merely t_m. However, in some cases, including ours, the number of data points and the feature set are small enough, such that parameter tuning is feasible.

At last we have to decide what performance metric we want to use for parameter tuning. Ultimately, the process of parameter-tuning will yield the model and the parameters with the best performance. However, the term “performance” is rather vague, and can in fact change meaning depending on the context. It is discussed in [16] that the performance evaluation of machine learning models lacks formal heuristics and consensus, which the authors try to remediate by grouping the performance metrics into classes containing correlated metrics. The reduction of performance metrics into fewer classes indeed gives some more clarity over the landscape of different metrics, but does not provide us with a method of choosing a suitable metric for evaluation of our model. Generally, the most commonly used metrics are calculated based on 4 numbers:

- **True positive count:** The number positive predictions, which are in fact positive.
- **True negative count:** The number of negative predictions, which are in fact negative.
- **False positive count:** The number of positive predictions, which are in fact negative.
- **False negative count:** The number of negative predictions, which are in fact positive.

With these numbers, we can calculate most of the popular metrics used in model evaluation. In fact, most of the performance metrics create a trade-off between these 4 numbers and this trade-off is the key in choosing the appropriate metric. For instance, in an aircraft security mechanism, the tolerance of having a false negative is very low, because the failure of detecting an engine fault can have catastrophic consequences. For an e-mail spam detector, a false negative is not as critical, because the result is merely some spam mail in our inbox folder. For e-mail spam detection, we are more concerned with minimizing the number of false positives, as we don’t want potentially important e-mail’s to end up in the spam folder.

Therefore, considerations about these trade-offs must govern the selection of the performance metric. Our end goal is to identify participators of a political spam-campaign on social media. We assume that ML models used for this purpose will flag potential spammers, while social media employees will need to verify the account manually before taking any action. In fact, a recent report on “coordinated efforts to manipulate or corrupt public debate for a strategic goal” published by Facebook, proposes automated detection combined with expert investigations as one of the mitigation techniques[56]. The manual verification step is rather expensive, from the point of view of the social media network, as it requires time and resources to investigate each flagged account. In order to focus the efforts on accounts which are most likely to be spammers, we therefore want to maximize the number of true positives, while also minimizing the number of false positives. By doing so, we have to take the trade-off that the number of false-negatives might increase. This means that our model might not capture all spam-campaign participators, but the ones that it flags are very likely to be spammers. Some of the most popular metrics used in model evaluation, which are discussed in [16][17], are precision, recall, accuracy and f1 score. The mathematical formulas for computing each metric are:

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4}$$

Where TP is the true positive count, FP is the false positive count, TN is the true negative count and FN is the false negative count. Accuracy tends to be the default evaluation metric, and also for a good reason. It is easy to interpret and understand as it is simply the ratio between the number of correct predictions and the number of total predictions. But it doesn't tell us anything about the FP-FN trade-off. We see that the precision is maximized whenever the number of false positives is low and that recall is maximized whenever the number of false negatives is low. So precision and recall are suitable metrics to use whenever we want to measure the trade-off between FP and FN count. F1 metrics is mostly used when we want to seek a good balance between precision and recall. Since we have already identified that our objective is to minimize the number of false positives in order to reduce the burden of manual verification for the human operators, we will prioritize the precision as our metric score during parameter tuning and evaluation. We will also display the other scores for comparison.

Training and Evaluation

Selecting model parameters based on evaluation against the training data will yield biased results, as the model is already fitted to the data. To avoid this bias, 5-fold cross can be used for validation to estimate model performance during training[57]. This technique splits the training data into 5 equally sized folds, trains the model on 4 of the folds and then evaluates the results on the fifth. This is done for all possible splits of the 5 folds, resulting in 5 training and validation sessions. The reported 5 scores are then averaged to get a training performance estimate. 5-fold cross validation is also used to avoid over-fitting, which happens when the model fits too well to the training data, thus having a bad generalization resulting in poor performance on new samples.

During the training phase, we apply parameter-tuning with grids defined in the previous section. We select the final parameters of each model based on the best 5-fold cross validation precision score. The models and their parameters are:

- **KNN**: The K-Nearest-Neighbour classifier which achieved the highest precision score during 5-fold cross validation uses uniform weights and $k = 200$ neighbours
- **SVM**; The Support Vector Machine which achieved the highest precision score during 5-fold cross validation uses $C = 0.1$, $\gamma = 0.01$ and $kernel = sigmoid$
- **RF**: The Random Forest classifier which achieved the highest precision score during 5-fold cross validation uses 50 estimators, square root number of features and a max depth of 80.

Having found the models with the best parameters, we are now ready to benchmark them against each other. Recall that we have dedicated a hold-out validation set for this purpose to avoid any bias from the training. Table 3 shows the validation scores of all 3 models and their ability to distinguish between trolls and the baseline.

Score Model - class	Precision	Recall	Accuracy	F1-score
SVM - troll	0.97	0.24	0.78	0.38
SVM - baseline	0.76	1.00	0.78	0.86
RF - troll	0.97	0.85	0.95	0.90
RF - baseline	0.94	0.99	0.95	0.96
KNN - troll	0.95	0.48	0.84	0.64
KNN - baseline	0.82	0.99	0.84	0.90

Table 3: Performance measured by each account class (validation set)

The scores of the models can be interpreted in two ways, depending on which class is selected to be "relevant". The choice of relevant class makes a difference for the interpretation of TP, TN, FP, FN and thereby also affects the performance metrics. Since our objective is to identify trolls, we will mostly look at the performance metrics where "troll" is the relevant class. Note that the accuracy is not affected by the choice of relevant class due to the way it is calculated.

Observing the performance scores of the validation set, we see that the models differ a lot, each having their strengths and weaknesses. Despite of our desire to maximize the precision, we cannot simply disregard the other metrics. We see that the SVM classifier has a precision of 0.97, meaning that 97% of the accounts identified by the model to be trolls, are in fact trolls. The SVM classifier is very likely to produce only relevant results for the human operators, thus lessening the burden of manual verification. However, the recall score of this model is only 0.24, meaning that only 24% of all the trolls from the validation set are identified. This number is much lower than the recall of the RF classifier. This is a good example of why we should look at multiple scores and choose a model which achieves a proper balance between them. In our case, we want our model to be precise, but not at any cost. In general we see that the RF classifier has good scores across all metrics, and even though it has a similar precision to both SVM and KNN, it also has a recall of 0.85 and therefore identifies a lot more trolls than the other models. The accuracy of 0.95, which is often used as the default metric, is also much higher than for the other models. Based on these scores, we will use the RF classifier and test it on the final 25% of the data. The test results are shown in table 3.

Score Model - class	Precision	Recall	Accuracy	F1-score
RF - troll	0.93	0.87	0.94	0.90
RF - baseline	0.95	0.97	0.94	0.96

Table 4: Final scores for Random Forest classifier (test set)

The final test score will often be lower than previous training and validation scores, as it is an unbiased performance estimate of the model. In our case, we also see a small drop in most of the metrics. The reported precision of the model is 0.93 for trolls, meaning that 7% of the accounts marked as trolls will be authentic, a fairly small share of additional irrelevant accounts to verify for the human operators. At the same time we observe a recall of 0.87, which means that our model identifies 87% of the trolls in the dataset, another important performance metric. The model has a good balance between precision and recall, and a fairly high accuracy against the test set.

Evaluation against well-known Twitter accounts

Estimating the model performance against an independent test set is important to get an unbiased estimate of the model performance. The unbiased score is an indicator of how the detection model will perform in real life. To make the testing more interesting, we will in this section provide a real-world example of applying the RF classifier to a set of well-known American Twitter accounts. We use the Twitter API to obtain all tweets of Elon Musk, Mitch McConell, Sarah Palin, Hillary Clinton and Nancy Pelosi from 2016. Furthermore, we have obtained tweets of Donald Trump from [27], as his account was suspended by Twitter at 8 January 2021 and the tweets are therefore no longer available through the API[28]. All the tweets are from the US election year of 2016, similar to the baseline dataset. For further details on the selected well-known accounts, we refer to the background section. Table 5 shows the computed variables used in the detection model for each selected account. We see that both right-wing republicans, Donald Trump and Sarah Palin, have a higher hashtag hitrate against the troll top-30 dictionary than the other accounts. The rest of the variables are somewhat similar, except the sentiment score of 0.06 of the republican opposition leader Mitch McConell, which is quite lower than for other accounts. It is not a surprise that the lifespan of these accounts is very high, considering that they are active famous figures on Twitter. Table 6 shows the detection results of applying the random forest classifier obtained from the previous section to the well-known accounts. Note that the names of the accounts are reduced to their initials. By using all 4 variables for detection, every account is correctly classified as authentic. For the sake of the example, we have also applied the detection model on the account data, where the lifespan variable

Variable Account	Hashtag hitrate	Sentiment	Subjectivity	Lifespan
@HillaryClinton	0.07	0.12	0.33	360
@SarahPalinUSA	0.27	0.20	0.40	363
@SpeakerPelosi	0.00	0.13	0.33	360
@elonmusk	0.00	0.17	0.32	365
@leadermcconell	0.00	0.06	0.30	364
@realDonaldTrump	0.23	0.17	0.40	365

Table 5: Variables used for detection of well-known accounts

Account Variables	HC	SP	NP	EM	MM	DT
All	0	0	0	0	0	0
w/o lifespan	0	1	0	0	0	1

Table 6: Classification of well-known accounts. 1:troll, 0:benign

is removed. Interestingly enough, this causes the model to misclassify both Donald Trump and Sarah Palin as troll accounts. The misclassification can primarily be attributed to the hashtag hitrate variable.

This small example illustrates that one should be cautious when using machine learning models to draw conclusions. The models are probabilistic in nature and can be fooled in the presence of outliers, poor data and low number of features. The performance of ML models is not only governed by the underlying learning algorithm, but also by the condition of the data input. We have tried to achieve such quality data by deriving significant variables which can help identify troll accounts. Having seen how a ML model can misclassify accounts due to issues with data, in this case too few features, we will now use the next section to look into the area of adversarial machine learning. In this domain, there is a malicious intent to fool the model. It can be valuable to study how such machine learning models can be tricked in the presence of an adversary in order to improve the integrity of the results in future applications.

Attacking the detection model

Adversarial machine learning has to do with deceiving the model in a way that makes it misbehave. Depending on the context, this can mean various things, but ultimately, the attacker wants to control the output of the model to their advantage. This can be achieved in a number of ways depending on the attackers capabilities. It is hard to argue about possible attack and defense vectors, without considering what an attacker is capable of. In the IT-security literature, this is often referred to as a threat model[58]. [19] presents a 4-phased taxonomy for adversarial ML domain, where the first phase is to establish the threat model. In adversarial ML domain, the capabilities of an attacker can include their knowledge about the model, like the learning algorithm being used, the exact weights of the model or the chosen hyper-parameters. Having knowledge of these internal parameters enables the attacker to perform queries against the model in order to infer something about the training data through its output. Having unrestricted access to the model can also help the attacker discover which malicious input vectors can avoid detection. This can be compared to learning the ruleset of a standard IDS. Furthermore, the attacker might have control of parts of the training data or the input data. Having control of the training data enables the attacker to perform data poisoning attacks, while having control of the input data provides capabilities to perform evasion attacks[21]. We will briefly describe these type of attacks in the following sections and try to apply them in the context of the studied spam-campaign.

Evasion

Evasion attacks are those where the adversary can evade detection by modifying the input to the model. This requires the capabilities of input modification, which is not always feasible. For example, if a ML model receives feedback from a pacemaker, it can be quite hard to access the pacemaker in order to modify the data it sends out. Figure 12 (right) shows an abstract representation of a ML detection model life cycle and highlights the point of compromise of the model in an evasion attack. The left side of the figure shows the effect of a poisoning attack, which we will cover in next section. Mutating the input data in a correct way can flip the output label and thus evade detection.

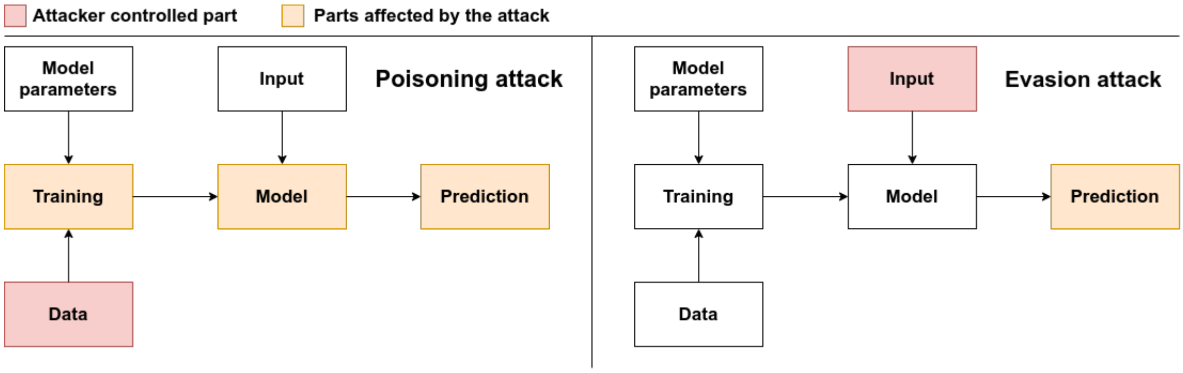

Figure 12: Poisoning and evasion attack points of compromise and affected parts

The evasion attack only affects the final prediction of the model, whereas data poisoning affects multiple parts of the model life cycle and can therefore be considered more invasive. It may seem like the evasion attack is easier to execute, due to its stealthiness and the lesser capability requirements, but it has been shown that the domain of cyber-security, including that of adversarial machine learning, evasion can in fact be very difficult. This is because modifying the input of the data will also modify the "payload", which is the malicious component of the attack. A modified payload, although undetected, may not achieve the malicious purpose it was designed for. For instance, modifying a just a few bytes in a malware binary can render it useless. Executing evasion attacks in other domains, like image recognition is easier, as it is possible to change the prediction by introducing small modifications to pixel images which are not detected by the naked eye[19].

In the context of the spam-campaign, the users are in full control of the tweets they post, and therefore have the capabilities to control the input to the detection model. In order for the evasion to be successful, the troll has to post tweets which resemble those of an authentic user, while still preserving the malicious component, namely the disinformation. Simply posting benign content will not work, as it will not have the desired effect of disinformation. Recall that the input features to our detection model are based on aggregations of all the tweets of a user. This opens the possibility of modifying only parts of the input to resemble benign user activity, while preserving some of the malicious tweets to keep the disinformation effect. To simulate this attack, we will add random tweets of authentic accounts to every troll user in the test data. The random tweets are selected from the test set of baseline accounts. After this, the number of troll tweets are doubled, being 50% benign and 50% malicious.

Score Model - class	Precision	Recall	Accuracy	F1-score
RF - troll (no evasion)	0.93	0.87	0.94	0.90
RF - troll (evasion)	0.85	0.59	0.85	0.70

Table 7: Comparing test scores of evasion vs no evasion

Table 7 shows the test score of the RF model when run against the modified troll input, together with the score from our previous test, where the input is not modified. We observe that the precision of the model has decreased from 0.85 to 0.93, meaning that amongst the users identified to be trolls, there will be 8 percentage points less trolls, if evasion is used. The recall score has decreased from 0.89 to 0.59 due to evasion, thus decreasing the total number of trolls which the model identifies. This is a significant degradation, meaning that a lot more spammers can survive detection if they all agreed to use this evasion technique. The results are not too surprising, as we have been adding benign tweet activity to the troll accounts, resulting in a higher false negative rate and therefore a lower recall score. In practice, this evasion technique should be fairly easy to automate. It only requires some software to periodically sample a number of random tweets and post them on behalf of the troll accounts.

Even though the attack seems successful, one still has to consider the impact such evasion technique can have on the disinformation effect of the campaign. The attack lets the spam-campaign keep an addition 30% of the spammers. But the cost of the evasion is that the Twitter feed of the spammer will be filled with benign Tweets which may dilute the impact of the disinformation. Furthermore, if the benign tweets are selected randomly, they may have content which directly contradicts the disinformation of the campaign, thus discrediting the troll even more. This can be avoided by manually crafting or selecting the Tweets, but yet again, the price will be paid in increased amount of labour for the spammers. Considering all this, even though the evasion seems successful, it comes at a price of degrading the effect of the spam-campaign.

Data poisoning

Data poisoning attacks are those which target the training set of the model prior to training, in order to change the model performance against new samples. Figure 12 (left) shows the point of compromise for the typical poisoning attack and the affected parts of the model life cycle. As mentioned earlier, data poisoning is a more involved attack, mostly because it requires greater capabilities from the attacker, but also because it affects the model directly and might be easier to detect. In order to avoid detection of data poisoning during evaluation phase of the model, attackers can install backdoors which only causes the model to misbehave post-evaluation or at a time selected by the attacker[22]. In the following experiment we will assume, that the attacker has unrestricted access to the training data and can modify the records and labels in an arbitrary way. [20] proposes two different algorithms for data poisoning; label flipping and adding new entries to the training set. The advantage of label flipping is the simplicity of the approach. By simply flipping the labels of authentic and inauthentic accounts randomly, we can degrade the models training performance. In order to add new entries to the training set, an examination of the data has to be performed to figure out how to craft records that would degrade the score. We will therefore try the simple approach of label flipping. The algorithm in listing 2 is applied to the training set. The parameter p is the poisoning rate, which is a probability that decides if we flip a label or not.

Listing 2: Simple label flipping algorithm

```
p = probability_of_flipping
for x,y in training_data:
  if rand() < p:
    y = flip(y)
```

Since the poisoning is applied during training, we need to run model selection and parameter tuning again, to see if the poisoning will affect our choice of model. For the sake of brevity, we will use only the accuracy score during the following benchmark. We will use the previously selected validation set to measure model performance. Figure 13 shows how the accuracy of the models develops as we adjust the poisoning rate.

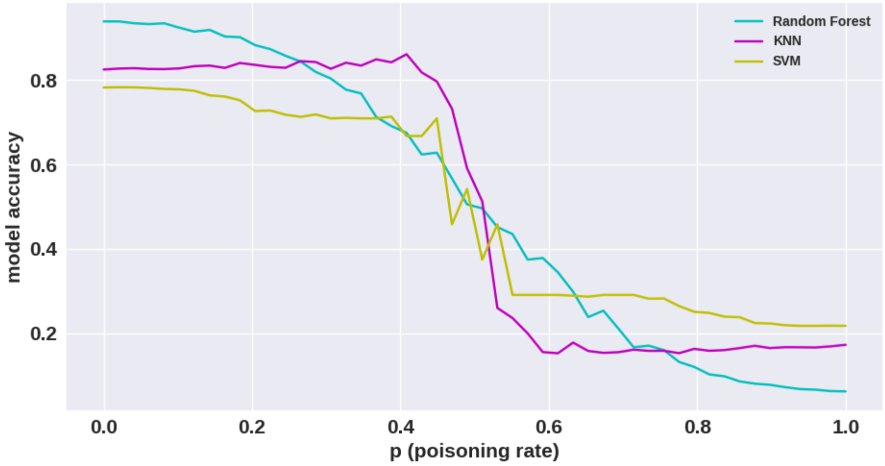

Figure 13: Empirical accuracy as a function of poisoning rate (avg. over 10 runs)

We see that each classifier reacts to the poisoning attack differently. The empirical accuracy for the random forest classifier suggests a linear relationship between the poisoning rate and the model performance, while the empirical accuracy of SVM and KNN resembles an inverse sigmoid curve. Since random forests are a collection of decision trees which makes a lot of splits in the data plane, it makes sense that randomly flipping points in that plane results in a linear decline in model performance. The chosen KNN model classifies the points by taking the majority vote of 200 nearest neighbors. If the data are well separated in 2 clusters from the start, it will take on average around 50% of data pollution, before the majority vote starts to change, which explains why we see a big decrease in accuracy at around $p = 0.5$. A similar behaviour holds for SVM with a sigmoid kernel. These results suggests that the adversarial setting has to be considered during the choice of classifier. For instance, for a poisoning rate of $p = 0.4$, the KNN model performs with an accuracy of 83%, which is a significant improvement from SVM and RF models with accuracies around 71% and 65% respectively. This example highlights that prior knowledge of p can be a deciding factor in model selection. If we assume a value of p before the training phase, we can use it as an additional parameter during parameter tuning, and thus select a better model for the adversarial setting that we are expecting.

Using the RF-classifier, we compute the models ability to identify trolls for different values of p. Table 8 shows the performance metrics for the RF-classifier against the test set. It is clear that both the precision and recall takes a toll when the data is polluted. At $p = 0.25$, we still have a relatively high recall of 0.81, meaning that the model is detecting 81% of all trolls in the dataset, but this comes at a cost of many false positives, as

Score p	Precision	Recall	Accuracy	F1-score
0.00	0.93	0.87	0.94	0.90
0.10	0.85	0.81	0.91	0.83
0.25	0.67	0.81	0.83	0.73
0.50	0.24	0.37	0.47	0.29

Table 8: RF performance for the troll class at different values of p

the precision is only 67%. At $p = 0.50$, the accuracy is 47% suggesting that if half of the data is poisoned, our model does not perform better than randomly guessing the labels. Furthermore, at $p = 0.50$, only 25% of the accounts flagged to be trolls are in fact trolls.

These results indicate that when dealing with an adversarial setting, it can be fruitful to consider the threat model in which we deploy the classifier. Having a proper threat picture might help us defend against evasion and poisoning attacks by adapting suitable countermeasures during the training or the detection process. For example, the results suggests that KNN classifiers are more robust against poisoning attacks for certain values of p. These results are also in line with the findings in [20], where a KNN classifier is used to adjust the poisoned dataset prior to training. While the poisoning attack seems more powerful, it can be discussed if it is feasible to execute in practice. Gaining access to the training set might be hard in different settings. In our case, the training set is collected from the Twitter API, so an attacker would need access to the Twitter database, the communication link or our machine in order to pollute the data. Furthermore, poisoning the model requires additional capabilities of installing backdoors, such that the attack is undetected during model evaluation. In the context of the russian spam-campaign, it seems like the demonstrated evasion attack can be performed with relative ease.

Reflection

Although we have succeeded in deriving a ML detection model capable of identifying the majority of spam-campaign participators, many obstacles were encountered during the process with regards to data availability and quality. As discussed in the limitations section, choosing a pre-existing dataset for such a project can automatically constrain the decision model to particular data features existing in the dataset. Furthermore, because of limitations of the Twitter API, it was not possible to query additional data attributes about the troll accounts. As a consequence of this limitation, we were restricted in the number of features we could use for the analysis. Another relevant data limitation, was the lack of access to a baseline dataset. This put an extra burden of obtaining an additional baseline dataset which was compatible with the existing one. Data availability is a common problem within ML-based projects, due to various reasons like bureaucracy, politics, privacy and moral concerns[29][18].

Much like the availability, data quality can also have big impact on the performance of ML-models. During the analysis phase, we discovered that some of the hypothesis could not be properly tested due to data quality. For example, the analysis of users tweet frequency over time showed big differences between the trolls and benign accounts, but when grouping the data on user level, we no longer had enough data points to make proper conclusions. Also, the user activity throughout the day could not be tested due to lack of reliable time-zone information. At last, the follower/following rate of trolls and baseline accounts was not comparable due to different sampling approaches between the datasets and therefore was not used in order to avoid bias in the detection model. The poor data quality had a direct consequence for the amount of features that we could include in the model, thereby potentially degrading the final results.

During our experiments, we saw the effect of removing 1 out of 4 features used for troll detection, resulting in misclassification of republicans Donald Trump and Sarah Palin as russian trolls. This is a good example of how data quality and availability are important in order to increase the number of features we can include in the model, thereby making it more reliable. In general, one has to be cautious with blindly trusting the results of ML models. Although they enable us to instantly process and classify big amounts of data, it has also been shown that the models inherit any intended or unintended bias that the underlying data may contain[30][31]. In the context of social media, misclassification can at most result in unrightful suspension legitimate account, disabling the user from accessing their content like pictures, Tweets or private messages. In other areas like law-enforcement, misclassification can have more serious consequences for the involved party, potentially resulting in false prosecutions and convictions[32]. This suggests that platforms utilizing AI for fraud detection should compliment the automatic detection with manual verification's, as also discussed in [56]. The presence of a human operator during detection of inauthentic accounts was also one of our assumptions during the project. Although human supervision of automated decision models involves an increased cost for the platform, it can be argued that keeping a human in the loop is a good idea in order to avoid consequences of unfortunate misclassifications. In the context of IT-security, there is a clear distinction between Intrusion Prevention Systems (IPS) and Intrusion Detection Systems (IDS), where the former actively blocks intrusions upon detection, while the latter merely notifies a human operator of a possible threat[33]. We argue that each system has a use-case, as the cost of misclassification can vary a lot depending on the context, suggesting that some settings require the assistance of a human, before acting upon predictions of a model.

Conclusion

Through a thorough analysis, we have derived 4 interesting indicators of troll accounts; hashtag hit-rate, sentiment score, account lifespan and subjectivity scores. All features were reported to have a significant correlation to the type of account. We applied different machine learning techniques as feature-scaling, parameter tuning and testing, in order to arrive at a well-performing model for troll account identification. During the model selection process, we have tried to consider the particular context that we are in, namely fake account identification on social media, in order to chose the best fitting model. Our results showed that a random forest classifier using 50 estimators, square root number of features and a max depth of 80, yields an accuracy of 94% against the test set. Furthermore, the model can identify 87% of all the trolls from the test set and of all accounts labeled as trolls by the model, only 7% are reported to be false positives.

During some of the experiments, we showed how the model can be fooled by a malicious party. Assuming that the attacker has prior knowledge about the model features, we showed that an evasion attack was possible and relatively easy to execute. We successfully performed an evasion attack on the random forest classifier resulting in some degradation of accuracy from 94% to 85% and a significant degradation of recall from 87% to 59%. In the context of spam-campaigns, reducing the recall rate is an important objective for the trolls in order to persist the disinformation on the platform. Assuming that the attacker has stronger capabilities, such as access to the training set, we saw that a more powerful poisoning attack can be performed, by utilizing a simple label-flipping algorithm. The poisoning attack has severe impact on model performance as the attacker can arbitrarily control the poisoning rate and therefore also the accuracy. The experiments suggested that in adversarial settings, the poisoning rate could be considered during parameter tuning and model selection, as models like KNN were observed to have a grater resilience against poisoning attacks. Despite of its effectiveness, it was argued that the poisoning attack relied on unrealistic assumptions about attacker capabilities, and was most likely unfeasible to execute in practice.

In general we experienced how data availability and quality are essential in order to obtain a good ML model. Some of the proposed hypothesis could not be tested properly due to data issues, resulting in fewer features which could be utilized by the model. Also, the direct consequence of removing 1 out of 4 data features was observed, when famous political figures like Donald Trump and Sarah Palin were misclassified as trolls due to data degradation. It is clear that a model benefits from using as many high-quality features as possible, calling for greater data collection efforts on platforms such as Twitter. In addition, the research community could benefit from easier access to this data in order to provide research which can build the foundation of future ML-driven detection systems.

Although we have succeeded in creating a model capable of identifying participators of the IRA spam campaign, continuous research is required to stay ahead of the state-sponsored astroturfing efforts. Hunting online spammers resembles the ongoing “cat and mouse” game between hackers and security experts observed in the field of IT-security. The existing detection methods will swiftly become obsolete, if not updated with new features and re-trained with fresh data on a continuous basis. Since 2016, many new spam campaigns have seen the light of day. Facebook alone has identified more than 150 new spam campaigns in the last 4 years, which violated their policy against “Inauthentic Coordinated Behaviour”[56]. This highlights the necessity of continuous research and development of ML models capable of identifying such coordinated astroturfing campaigns.

References

[1] *Russian troll tweets dataset*
https://github.com/fivethirtyeight/russian-troll-tweets

[2] *Online astroturfing: A theoretical perspective*
Zhang, J. & Carpenter, Darrell & Ko, M.

[3] *Controlling astroturfing on the internet: A survey on detection techniques and research challenges*
Mahbub, Syed & Pardede, Eric & Kayes, A. S. M. & Rahayu, Wenny

[4] *An Examination of the Impact of Astroturfing on Nationalism: A Persuasion Knowledge Perspective*
Kenneth M. Henrie, Christian Gilde

[5] *Troll Factories: The Internet Research Agency and State-Sponsored Agenda Building*
Darren L. Linvill and Patrick L. Warren.

[6] *Disinformation Warfare: Understanding State-Sponsored Trolls on Twitter and Their Influence on the Web*
Zannettou, Sirivianos, Caulfield, Stringhini, De Cristofaro, Blackburn

[7] *"Donald Trump Is My President!": The Internet Research Agency Propaganda Machine*
Bastos, Marco & Farkas, Johan

[8] *Hillary Clinton 'collapse'*
https://www.independent.co.uk/news/world/americas/hillary-clinton-collapse-faint-video-9-11-memorial-democrat-ill-overheated-latest-a7237276.html

[9] *Pussygate*
https://www.theguardian.com/us-news/2016/oct/07/donald-trump-leaked-recording-women

[10] *Twitter is sweeping out fake accounts like never before, putting user growth at risk*
https://www.washingtonpost.com/technology/2018/07/06/twitter-is-sweeping-out-fake-accounts-like-never-before-putting-user-growth-risk/

[11] *TIES: Temporal Interaction Embeddings For Enhancing Social Media Integrity At Facebook*
https://arxiv.org/abs/2002.07917

[12] *Detecting Fake Accounts on Social Media*
Khaled, El-Tazi, Mokhtar

[13] *Efficient Estimation of Word Representations in Vector Space*
https://arxiv.org/pdf/1301.3781.pdf

[14] *VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text*
C.J. Hutto and Eric Gilbert

[15] *Differences in How Democrats and Republicans Behave on Twitter*
Pew Research Center

[16] *A Strategy on Selecting Performance Metrics for Classifier Evaluation*
Liu, Zhou, Wen, Tang

[17] *Performance Evaluation of Machine Learning Algorithms in Post-operative Life Expectancy in the Lung Cancer Patients*
Kwetishe Joro Danjuma

[18] *"We Would Never Write That Down": Classifications of Unemployed and Data Challenges for AI*
Petersen, Christensen, Harper, Hildebrandt

[19] *Adversarial Machine Learning Attacks and Defense Methods in the Cyber Security Domain*
arXiv:2007.02407

[20] *Exploring Adversarial Attacks and Defences for Fake Twitter Account Detection*
Kantartopoulos, Pitropakis, Mylonas, Kylilis

[21] *A Taxonomy and Survey of Attacks Against Machine Learning*
Pitropakis, Panaousis, Giannetsos, Anastasiadis, Loukas

[22] *Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering*
arXiv:1811.03728

[23] *10 Twitter statistics every marketer should know in 2021*
https://www.oberlo.com/blog/twitter-statistics

[24] *Misinformation vs disinformation*
https://www.dictionary.com/e/misinformation-vs-disinformation-get-informed-on-the-difference/

[25] *NCSS statistical software chapter 302: Point-Biserial and Biserial correlations*
NCSS

[26] *Statistics IV: Interpreting the results of statistical tests*
Anthony McCluskey, BSc MB ChB FRCA, Abdul Ghaaliq Lalkhen, MB ChB FRCA

[27] *https://www.thetrumparchive.com/*

[28] *https://blog.twitter.com/en_us/topics/company/2020/suspension.html*

[29] *Temperaturmåling af signaturprojekterne*
KL, Danske Regioner, Digitaliseringsstyrelsen

[30] *Bias in data-driven artificial intelligence systems—An introductory survey*
Ntoutsi, Eirini & Fafalios, Pavlos & Gadiraju, Ujwal & Iosifidis, Vasileios & Nejdl, Wolfgang & Vidal, Maria-Esther & Ruggieri, Salvatore & Turini, Franco & Papadopoulos, Symeon & Krasanakis, Emmanouil & Kompatsiaris, Ioannis & Kinder-Kurlanda, Katharina & Wagner, Claudia & Karimi, Fariba & Fernandez, Miriam & Alani, Harith & Berendt, Bettina & Kruegel, Tina & Heinze, Christian & Staab, Steffen

[31] *Big Data's Disparate Impact*
Solon Barocas & Andrew D. Selbst

[32] *Automating Inequality: How High-tech Tools Profile, Police, and Punish the Poor. St. Martin's Press.*
Eubanks

[33] *NIST 800-94, Guide to Intrusion Detection and Prevention Systems*, 2007.

[34] *Økonomistyrelsens første AI system skal spare millioner på faktura-håndtering: https://www.version2.dk/artikel/oekonomistyrelsens-foerste-ai-system-skal-spare-millioner-paa-faktura-haandtering-1092672*

[35] *https://hashtagroundup.com/2015/03/15/what-is-a-hashtag-game/*

[36] *Online Learner Authentication: Verifying the Identity of Online Users*
Bailie, Jortburg

[37] *Sampling Methods in Research Methodology; How to Choose a Sampling Technique for Research*
Taherdoost

[38] *https://developer.twitter.com/en/docs/twitter-api/tweets/sampled-stream/introduction*

[39] *Sampling: how to select participants in my research study?*
Martinez-Mesa J, González-Chica DA, Duquia RP, Bonamigo RR, Bastos JL

[40] *Analyzing Twitter Users' Behavior Before and After Contact by the Internet Research Agency*
arXiv:2008.01273

[41] *Characterizing the 2016 Russian IRA Influence Campaign*
rXiv:1812.01997

[42] *The Tweets They are a-Changin': Evolution of Twitter Users and Behavior*
Liu, Kliman-Silver, Mislove

[43] *Sentiment Analysis of Twitter Data: A Survey of Techniques*
arXiv:1601.06971

[44] *https://www.bbc.com/news/world-europe-36800730*

[45] *Stemming and Lemmatization: A Comparison of Retrieval Performances*
Balakrishnan, Lloyd-Yemoh

[46] *The Effects of Data Quality on Machine Learning Algorithms*
Sessions, Valtorta

[47] *Data Collection and Quality Challenges for Deep Learning*
Whang, Lee

[48] *Data Quality Considerations for Big Data and Machine Learning: Going Beyond Data Cleaning and Transformations*
Gudivada, Apon, Ding

[49] *About Feature Scaling and Normalization and the effect of standardization for machine learning algorithms*
Sebastian Raschka

[50] *Data Splitting*
Z. Reitermanová

[51] *SPlit: An Optimal Method for Data Splitting*
Joseph, Vakayil

[52] *On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning*
Xu, Goodacre

[53] *Evaluation of the Performance of K-Nearest Neighbor Algorithm in Determining Student Learning Styles*
Musa, Elnazeerp, Galadancip, Amship, Saidup, Onyema, Isa

[54] *Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS*
arXiv:1912.06059

[55] *Tuning the hyper-parameters of an estimator*
https://scikit-learn.org/stable/modules/grid_search.html

[56] *Threat Report - The State of Influence Operations 2017-2020*
Nathaniel Gleicher, Margarita Franklin, David Agranovich, Ben Nimmo, Olga Belogolova, Mike Torrey

[57] *Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning*
Sebastian Raschka

[58] *Threat Modeling as a Basis for Security Requirements*
Myagmar, Lee, Yurcik

[59] *Tweets of random sampled accounts from 2016*
https://www.kaggle.com/sergejbogachov/tweets-of-random-sampled-accounts-from-2016

[60] *Train, validation and test set of authentic and malicious Twitter users*
https://www.kaggle.com/sergejbogachov/twitter-authentic-and-malicious-users

[61] *Project code repository*
https://github.com/Retler/TrollAccountsClassification