Using usernames to predict the survival of accounts on Twitter

Bálint Gyevnár

MInf Project (Part 1) Report
Master of Informatics
School of Informatics
University of Edinburgh

2020

Abstract

Usernames are short strings that serve as unique and persistent account identifiers in social media. Previous research has shown that the content and structure of usernames may encode important information about many different account characteristics, such as whether the account is a bot. The goal of the thesis is to establish to what extent the usernames of Twitter accounts help predict the survivability class (alive, suspended or deleted) of the associated accounts. Survivability has been comparatively neglected in the literature, but it is a useful feature for Twitter analysis. If an account is likely to be suspended or deleted soon, it may be more likely to be a bot, or malicious.

The analysis proceeds in three steps: username segmentation, dimensionality reduction of the resulting sparse data set, and classification.

In the segmentation step, N-gram, random split, informed segmentation, the Morfessor algorithm, and a combinations of informed segmentation and Morfessor are compared.

Classification is based on four different feature sets: uncompressed username segmentations, compressed segmentations, uncompressed metadata, and uncompressed username segmentations plus meta data.

In order to reduce the dimensionality of the segmented user name data, we first apply Latent Semantic Analysis (LSA), followed by a Principal Component Analysis of LSA results.

The classification algorithms used are Support Vector Machines (SVM) and Random Forests (RF), which are well suited for dealing with sparse data.

Classifiers that are based only on username segmentations outperform the baseline classifier, which predicts the majority label. However, the best performance, which is comparable to the state of the art, is achieved using only uncompressed metadata. Since username representations are relatively sparse, they introduced large variances across features, the results degraded.

We conclude that while username features show promise, classifiers that incorporate them will need to be trained on a large dataset.

Acknowledgements

I would like to extend my gratitude to my supervisor Maria Wolters for greatly helping with my thesis even during the unpredictable times of the COVID-19 disease and social isolation.

Table of Contents

Chapter 1

Introduction

1.1 Twitter

Online Social Networks (OSNs) serve as a major channel for communication between people and provide an indispensable source of news for many. In particular, Twitter has gained a huge popularity since its founding in 2006, as a simple and direct microblogging platform, where users can issue short, maximum 280 (previously 140) character long messages called *tweets*. According to Twitter, it had around 330 million registered users in the first quarter of 2019, of which approximately 145 million were active daily[1]. Furthermore, alexa.com reports[2] that the website ranks 39th in global user engagement as of 27 March 2020, making it an ideal testbed for studies in a number of fields.

Twitter supports inter-user connections much like any other OSNs, which can give rise to complex networks of relations and information propagation [28]. Within the Twitter ecosystem, accounts followed by a given user are referred to as *friends*, and accounts following that user are called *followers*. The follower-friendship relationships however need not be mutual. Furthermore, whenever a friend makes a tweet, that tweet appears on the news feed of every follower, who can then comment on it, or *retweet* (i.e. repost) it. Users can directly refer to each other in tweets by tagging others using the @ symbol, and relevant topics or thoughts are often indicated using hashtags with the # symbol. Users may also create a collection of accounts called *lists*, that can be followed similarly to individual accounts and provide easy access to tweets from all members of the list.

The popularity and size of Twitter are very often used for commercial or political purposes, that have been transforming the platform from microblog to an elaborate influencing system [42]. Additionally, the easily accessible and very loosely moderated channels of Twitter also facilitate the wide spread of rumours and fake news. Incentives such as these and an easily accessible API have fostered the existence of automated

[1] `https://investor.twitterinc.com/financial-information/quarterly-results/`
[2] `https://www.alexa.com/siteinfo/twitter.com`

robots (bots) on the platform. Even though the Twitter rules[3] explicitly prohibit any type of platform manipulation and spam activities, both bots of benign (marketing, news alerts, weather, etc.) and malicious (spam, astroturf [50], smoke screening [1], etc.) nature have been detected on the platform [43].

1.2 Motivation

It became clear, that malicious bots can have a negative effect on society and individuals alike, as seen during events such as the 2016 US presidential election, the Ukraine crisis, Brexit, or more recently the waves of misinformation around the COVID-19 outbreak, therefore any attempt to counter the spread of ill-intent bots are a step closer to restraining their effects.

In response to the prevalence of malicious accounts, there is great research interest into the detection, analysis and classification of bots on Twitter. Early bots, also referred to as traditional bots, performed repetitive or predictable tasks, therefore initial methods could use standard machine learning techniques (SVM, logistic regression, etc.) to detect them based on huge corpora scraped from Twitter primarily using linguistic information contained within tweets [48, 49]. On the other hand, with the improvement of detection evolved a new generation of bots called social bots, that try to imitate the behaviour of humans to improve their chances of being unnoticed by existing methods and to seem more believable to genuine users. These bots are much harder to detect, therefore several studies were conducted to improve classification accuracy [7, 26, 14].

Twitter has put some automated measures in place, in part built on users manually reporting malicious behaviour, to combat the spread of ill-intent accounts, however the exact methods are not open to the public. As a direct consequence of these measures many bots or fake accounts are *suspended*. These algorithms are not perfectly accurate, but there is a clearly observable correlation between the survival of an account and its level of automation as indicated by Figure 1.1 which plots the survivability of accounts against their ages. Traditional bots as described earlier are simpler so they are more easily detected and suspended by Twitter. On the other hand, social bots have a much better chance of avoiding suspension due to their ability to imitate humans. Many automated accounts are also *deleted*, which hints that these bots are 'cyborgs', meaning they are partially operated by humans and are not fully autonomous, as only humans can request the deletion of accounts. Naturally, only very few genuine users were suspended and most of the now inaccessible accounts were instead deleted. In general, real users behave heterogeneously in terms of survivability. A fourth type of frequently suspended accounts are fake followers which are used to boost follower counts. Several users can be suspended in bulk (as shown by the blue stripes on the figure) because their behaviour follows very simple patterns including predictable naming conventions and very similar dates of creation. Therefore, relating account features to survival could provide useful information for later works trying improve bot detection.

Most research on survival prediction or bot detection uses primarily linguistic and temporal information related to tweets to perform their analysis, however they do not

[3] `https://help.twitter.com/en/rules-and-policies/twitter-rules`

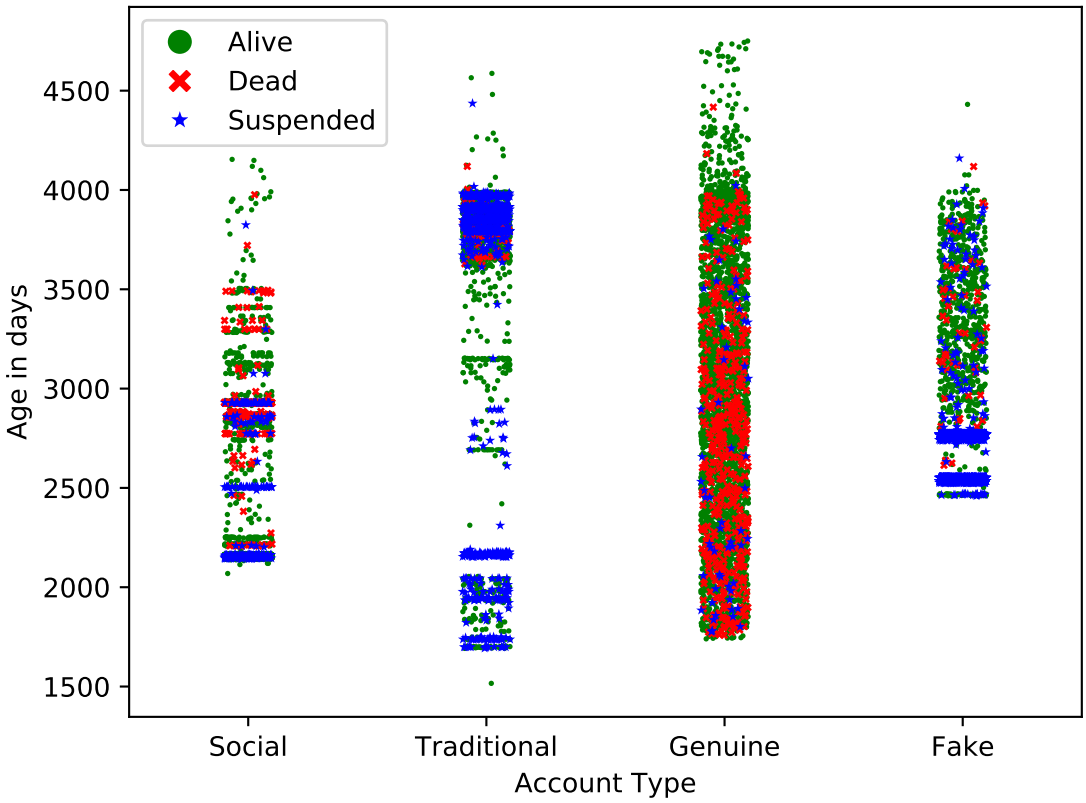

Figure 1.1: Survival of account types against their age. The blue stripes correspond to groups of accounts that were created around the same time and were suspended at the same time.

deeply consider account related information, most importantly *usernames*. These features are intrinsic to the account itself, therefore they are very difficult or even impossible to change, which makes them easy to track and very *robust* against the changing tactics of bots. Furthermore, usernames and account metadata is very *easy to retrieve* using the Twitter API, and the survival state (alive, dead or suspended) of accounts is also easy to check. Using quickly accessible information is crucial as annotating large corpora of tweets is time consuming and lacks the *adaptability* necessary to accommodate the evolving strategies of bots. Simple features derived from usernames such as character length distributions can also already hint at a relationship between survivability and usernames, as suspended accounts largely prefer 15 character long names while alive and dead accounts are more centred around 12 character lengths. The usernames of accounts considered in this analysis often also follow prominent and sufficiently general *naming patterns*, which are later exploited to design the informed segmentation method. Combining these observation with the fact that automated accounts are more correlated with the suspended or dead survival state, it is promising to use usernames for survivability prediction.

Finally, [47] argue that much of the research related to the analysis of OSNs rely on large scraped datasets drawn from public social networks, that are by nature dynamic and so access to account information may be removed or restricted over time. This hinders the reproduciblity of works. It would therefore be beneficial for researchers to know in advance whether their data would persist over time so that they could build on data that allows the results to be reproduced.

1.3 Contributions

The contributions of the thesis is two-fold:

1. Analysis of the **syntax** and **semantics** of usernames in the context of account types and survivability. The discovered information is used to test several **segmentation methods** for extracting latent information. The methods are: N-gram, random, informed, Morfessor, minimising combined and standardising combined. The random, informed, and combined methods are novel approaches presented in this thesis.

2. Prediction of **account survival** based on username **segmentation data** encoded using a bag-of-words (BoW) model, and also with the addition of **profile and tweet metadata**. The SVM and random forest methods are tested to establish a connection between survivability and accounts, and to the assess the effect of including segmentations.

As far as the author knows, no research has been published on the syntactic and semantic analysis and decomposition of usernames, therefore an approach for their systematic analysis is presented and used to predict survivability. These segmentation methods are general (except for the informed method which however can be tailored to data specific observations) and could be used in further works to analyse usernames.

The detailed outline of the thesis is as follows:

In Chapter 2, a literature review of related research outlines the current standing of account and survivability analysis on Twitter and places this thesis into context. This is followed by an elaboration on the *background* of already established methods (such as Morfessor, PCA, SVM, etc.), that are used for the analysis in this work.

Chapter 3 introduces the dataset used for the analysis. It then provides a detailed exploration of topics briefly covered in Section 1.2 that motivate the writing of the thesis. First, a preliminary semantic and syntactic analysis of usernames is given with particular focus on the relationship between usernames and survivability. This is followed by an exploration of the connection between account types and survival states. Finally, the tweeting habits of accounts are examined in the context of account types and survivability.

In Chapter 4 the description of tested username segmentation *methods* and an evaluation of the Morfessor based methods is presented. Then the methods for the three steps of the analysis are detailed. First the application of segmentation methods to usernames is explained. Then the use and need of dimensionality reduction is presented with focus on the issue of data sparsity. Finally, the classification methods are explained, in particular the various metadata and segmentation based features and the employed models.

Chapter 5 presents the *results* of predicting survivability using username segmentations and additional features with conclusions on the performance of the examined systems.

Finally, Chapter 6 gives a summary of the topics discussed in the thesis with open questions for further research.

The results of the thesis show, that usernames provide significantly improved performance in predicting the survivability of accounts over the baseline, which predicts the majority label of all instances. However, usernames are by design very unique and concise, therefore data sparsity is a prominent issue. To solve this, dimensionality reduction is used which is however a lossy transformation. As segmentations are also extremely varied, including them in a classification based only on metadata information actually deteriorates the performance of the classifier due to the large increase in feature variance. This could be counteracted using more data, however that was not available during the writing of this thesis.

Chapter 2

Background

This chapter first presents an overview of relevant research and related work in the field of OSNs, bot and account analysis with particular focus on survivability and usernames. This is followed by an exploration of the background on methods used for the analysis of usernames and survivability. In particular, the Morfessor algorithm is presented which provides the basis for three of the tested segmentation methods. The actual segmentation methods are presented in Section 4.1. The dimensionality reduction methods, Latent Semantic Analysis (LSA) and Principal Component Analysis (PCA) are described briefly which are followed by an overview of Support Vector Machines (SVM) and Random Forests (RF).

2.1 Survivability

As introduced in Section 1.2, an important concept related to Twitter accounts is *survivability*. All users on Twitter can be categorised into one of three classes based on survival state: *alive*, *suspended* and *dead*. This state could encode useful knowledge about the account type. Traditional bots and fake followers are more susceptible to suspension, while social bots or genuine accounts are more likely to be deleted [13]. In fact, [45] take the simplifying assumption in their work, that all suspended accounts are spammers and validate this hypothesis using 1.1 million tweets. They also find, that 77% of spammers are suspended within one day of their initial tweet. However, they do not provide an automated approach to predicting whether a given account would be suspended or not.

In addition, [2] used a dataset of 1.6 million tweets by 292,000 users to examine patterns in deleted tweets and found that the client (mobile, web, etc.) and the spatial information was highly discriminative of deleted tweets. Several other works have been published on spam tweet deletion [34, 39, 40] based on different linguistic features, however account survival remains a largely understudied field.

Importantly however, the thesis builds on the work of Volkova and Bell [47] who use log-linear models and recurrent neural networks on shallow and deep linguistics features, embeddings of tweets and account related metadata to perform a classification of

approximately 300,000 Twitter accounts into alive, suspended and deleted categories, achieving current state of the art performance. Additionally, they propose to use LSA as a tool for dimensionality reduction of tweets. However, their work does not use information embedded into usernames to create additional features. They also note, that most related works focus on the detection or analysis of malicious behaviour and not on the prediction of the future survival of accounts. Finally, their prediction results are used here as basis for comparing classification performance of the models presented in this thesis.

2.2 Usernames

An important, and crucially, persistent feature of every account is the unique, maximum 15 character long *username* referred to as screen name on Twitter. These may be built from any lowercase or uppercase character of the English alphabet, any base-10 numeric digit and the underscore.

Despite the apparent lack of expressibility, studies have shown before, that important information may be inferred from simply these strings, including sex and language [27]. In addition, usernames allow people to establish several online personas, that allows them to be recognisable on OSNs [9]. This method of self-tagging may allow users to reflect individual beliefs in a particular choice of username [24]. Considering the rich information embedded in the usernames of *genuine accounts*, it is reasonable to hypothesise, that automated accounts would lack such content, which forms one basis for analysing their behaviour. This assumption is also supported by the observation, that most bots have very similar or predictable usernames, as they often stem from the same bot farm. A consequence of this, is that such accounts are suspended in bulk which determines their survival state.

Because of the previously mentioned reasons, username are analysed in depth for this thesis. As a result, this work relies heavily on the segmentation of usernames during analysis. Prior work on this topic has been done in [27], who used N-gram splits and the Morfessor algorithm [46] to split usernames into compounds, while [24] gave a description of internal username structures, that help establish segmentation rules.

2.3 Bots

The analysis of bots on Twitter is a well researched topic, that has important connections to social and political fields with the recent surge of misleading or fake information [32, 25]. It is noted here, that due to the evolving nature of bots several different terminologies are in use in the literature. For example, Ferrara, et al. [19] use social bots as a general term for automated robots on OSNs, while Cresci, et al. [13] use the narrower definition, meaning bots acting as humans. For the purposes of this analysis the latter meaning is used in conjunction with the definitions laid down by Stieglitz, et al. [43], that are summarised in Table 2.1. The two main categories defined for bots are *intent* and *imitation of human behvaiour*.

		Intent		
		Malicious	Neutral	Benign
Imitation of humans	High	Astroturfing bots, influencer bots, infiltration, etc.	Humouristic bots	Chat bots
	Low	Spam bots, Fake accounts, etc.	Nonsense bots	News feed bots, recruitment, public announcement, etc.

Table 2.1: Categorisation of bots on OSNs by [43]

Many efforts of bot detection involve the collection of large datasets, that usually utilise the Twitter Streaming API and the `statuses/filter` query, which provide a real time filtered feed (using topic keywords, hashtags and user IDs) of 1% of all tweets generated at a moment [49]. Several studies utilise manual labelling, sometimes using crowdsourcing, to annotate the scraped data and to create a ground-truth [8]. Classification is performed using a set of carefully considered features that are used with some type of supervised machine learning algorithm [8, 16]. These approaches have proven to be useful in detecting early generation Twitter bots or *traditional bots*, but they require a lot of manual data collection and annotation, while survivability information can be quickly retrieved for accounts, that is important to sustain good performance in the quickly changing Twitter ecosystem. In addition, with the evolution of human-imitating bots the need for more rapidly adapting approaches has increased.

In response to the success of *social bots* [13], several unsupervised approaches have been proposed, that take the temporal patterns [7], retweets [35] or social fingerprints [11] of bots into account to drastically improve detection accuracy. The major advantages of these methods are that they do not require annotated data to run and are capable of some level of adaptation to bots as they do not rely on textual information but rather detect behavioural patterns over time.

A third type of bot-like entities on Twitter are *fake followers*. These accounts are used to increase follower counts of other users to alter their public perceptions. Several websites offer services online to inflate user followings, however fake followers risk distorting the definitions of popularity and influence on OSNs [12]. Unlike bots of other kind, these accounts have a short lifespan and get suspended in large numbers by Twitter.

2.4 Morfessor

Morfessor is used in this thesis as the basis for three of the tested segmentation algorithms. The detailed workings of all segmentation techniques are given in Section 4.1. Here only the motivation and a mathematical overview is given for Morfessor as this is a complex method and other segmentation algorithms rely on straightforward techniques.

Morfessor is a set of unsupervised probabilistic learning algorithms, that find morphological segmentations of words for a given natural language [46]. A morpheme in linguistics is defined as the ”minimal linguistic unit with lexical or grammatical meaning” [5]. It was shown, that Morfessor performs well on complex morphological segmentation tasks [15], which makes it an ideal candidate for username splitting. It is noted, that usernames are often made up of individual words and are not a single conjugated expression, so it is important that Morfessor is completely agnostic to whether it is analysing a list of terms or a list characters.

The Morfessor segmentation algorithm is based on two parts, a *lexicon* which describes the properties of morphemes (equivalently, the distribution of the data), and a *grammar* which encodes the rules of combining morphemes into word forms (i.e. the distribution of the model). Training is based on minimising the negative-log MAP-estimate of this system, where the prior of the lexicon is based on the Minimum Description Length (MDL) principle and the likelihood of the grammar uses a conditional independence assumption given the parameters of the grammar. The grammar and the lexicon formulate two competing goals, as the probability of the former is maximised with longer morphemes and the probability of the latter with shorter morphemes. The trade-off between the two can be adjusted with a hyperparameter $0 < \alpha \leq 1$ called 'corpus weight'.

As this cost function is computationally intractable, Morfessor uses a recursive iterative approach for finding optimal segmentations. This algorithm also supports semi-supervised learning with annotated data, where the weight of the annotated data can be adjusted using a hyperparameter $\beta > 0$ called 'annotation weight'. Segmentation after training can be performed using the Viterbi algorithm to find the most probable sequence of morphemes.

2.5 Dimensionality Reduction

The classification approach for the username segmentations are based on a bag-of-words (BoW) term-frequency model, which means that the counts of every distinct term among the username segmentations are encoded with their own feature. This model is simple to interpret, however due to the uniqueness of usernames the feature space generated is very sparse (i.e. zero for most features). To alleviate this problem dimensionality reduction must be performed. In this thesis two methods were used namely latent semantic analysis (LSA) and principal component analysis (PCA). The former is used to reduce only the segmentation data and the latter is used to decrease the number of dimensions of the combined data (metadata + segmentations). The following section gives a brief overview of these two methods.

2.5.1 Principal Component Analysis

Principal component analysis is a standard variance maximising dimensionality reduction technique. It finds linearly independent vectors called *principal components* onto which if the data is orthogonally projected the largest possible amount of variance is retained. It has a very well established history of use across many fields and is often the

state-of-the-art dimensionality reduction method in many applications. It was applied successfully on Twitter data for mapping emotions to detect mental health issues [29] and spam detection [37].

This thesis uses the implementation of PCA from the Python package scikit-learn[1] [38]. The algorithm is based on calculating the singular value decomposition (SVD) of the input matrix of data. In general, if the data is stored in a n x m real-valued zero-mean matrix $\mathbf{X}$, then the singular value decomposition of $\mathbf{X}$ is given by the eigen-decomposition of the covariance matrix $\frac{1}{n-1}\mathbf{X}^T\mathbf{X}$. If this is given as $\mathbf{U}\Sigma\mathbf{U}^T$, then the columns of $\mathbf{U}$ correspond to the principal components and Σ is a diagonal matrix of eigenvalues whose magnitude are proportional to the total variance encoded by the corresponding eigenvector. One can choose $k < n$ of the largest eigenvalues and use their eigenvectors as the basis for the orthogonal projection of the original data. As PCA is sensitive to the scaling of the original variables it is important to standardise the input data before running the algorithm.

2.5.2 Latent Semantic Analysis

Latent semantic analysis is a linear dimensionality reduction method proposed by [18] and was used here following the work of [47] on usernames. LSA is a variant of PCA, meaning it also uses a singular value decomposition (SVD). Unlike PCA however, LSA was specifically designed to work in natural language processing settings with often sparse term-document matrices of corpora modelled with BoW. This thesis uses a particular implementation of LSA called truncated SVD, which is part of the Python package scikit-learn[2].

The dimensionality reduction process consists of four phases. The first phases is simply setting up the term-document matrix. The second phase transforms the matrix elements with a function, that helps the SVD-based algorithm perform better. This is usually chosen to be $\log(freq_{i,j} + 1)$ for the i, j element of the matrix. The choice of function is a heuristic based on empirical evidence [18]. The third phase is calculating the SVD decomposition on the transformed matrix. The fourth phase involves selecting the desired number of latent dimensions and projecting the original data onto this subspace.

The main purpose of LSA is to find a dense representation of the term-document matrix, that efficiently encodes the semantic information between terms and can extract relations among words based on context [18]. It was shown to work successfully in a number of settings [17, 23, 20] including usernames [47].

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html`

[2] `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.TruncatedSVD.html`

2.6 Models

2.6.1 Support Vector Machine

Support vector machines (SVM) [10] are standard supervised machine learning algorithms, that are widely used and produce state-of-the-art results in many domains. They are used extensively with data from Twitter for goals such as detecting social spammers [30, 4], emerging epidemics such as the flu [3] or sentiment analysis [22].

SVMs are a family of non-probabilistic binary linear classifiers, that are also called maximum margin classifiers. Their goal is to find a hyperplane that separates the data into two disjoint subsets, while maximising the distance (i.e. the margin) from the hyperplane to the nearest datapoints called support vectors. If a normal vector to this hyperplane is denoted with $\mathbf{w}$, then for a given dataset $D = \{(\mathbf{x}_i \in \mathbb{R}^m, y_i \in \{-1,1\})\}, i \in \{1,\ldots,n\}$, this problem formulation is equivalent to solving a quadratic optimisation problem. Furthermore, SVMs are also able to fit non-linearly separable data through the use of slack variables $\zeta_i \geq 0$. In this case, the model is penalised for violating the linearity assumptions, so the quadratic program becomes:

$$\begin{aligned} &\min_{\mathbf{w},w_0,\zeta_i} \|\mathbf{w}^2\| + C\sum_i \zeta_i^k \quad \text{s.t.} \\ &y_i(\mathbf{w}^T x_i + w_0) \geq 1 - \zeta_i \end{aligned} \tag{2.1}$$

Here C is a hyperparameter of the model, that adjusts the trade-off between correct classification of the training instances and the size of the margin. The larger this value is the less important a wide margin becomes, which effectively results in a more complex decision boundary. For this reason C can be viewed as a regularisation hyperparameter. The exponent k is set to 1 in most implementations of SVMs and is a fixed value.

The solution of the quadratic program of Equation 2.1 is a linearly weighted sum of the original data where every datapoint $\mathbf{x}_i$ with $\alpha_i \neq 0$ is a support vector: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$. An illustration in two dimensions for the result of running SVM is given in Figure 2.1. Using this result predictions can be made according to the function:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w}^T\mathbf{x} + w_0) = \text{sgn}(\sum_i \alpha_i y_i (\mathbf{x}_i^T \mathbf{x}) + w_0) \tag{2.2}$$

The last part of Equation 2.2 is crucial as the dot product $\mathbf{x}_i^T\mathbf{x}$ may be replaced by any function $\kappa(\mathbf{x},\mathbf{x}_i) = \phi(\mathbf{x})^T\phi(\mathbf{x}_i)$ called *kernel*, that is defined as the dot product of the inputs in a transformed feature space given by ϕ. The kernel function κ gives a closed form formula for the computation of this dot product. This allows the SVM to work in a transformed feature space different from the original input data space and enables it to produce highly non-linear decision boundaries in the input space without needing to compute the dot-product explicitly. One such important kernel, that is also used in this thesis is the Gaussian radial basis function (RBF) defined as $\kappa(\mathbf{x},\mathbf{x}_i) = \exp(-\gamma\|\mathbf{x}-\mathbf{x}_i\|^2)$ for some $\gamma > 0$. The higher γ is the less effect a single

[3] https://en.wikipedia.org/wiki/Support-vector_machine

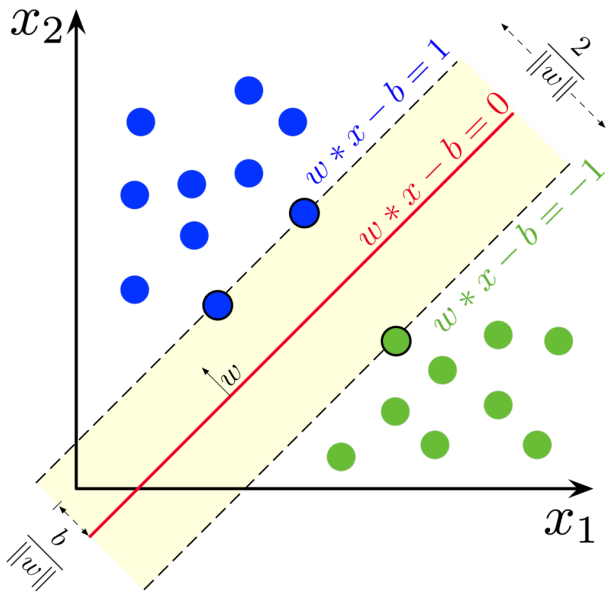

Figure 2.1: SVM maximum margin classifier. Image from Wikipedia[3].

training instance will have on the resulting decision boundary. This kernel in fact corresponds to an infinite dimensional transformed space, as the exponential function can be interpreted as an infinite sum of polynomials as given by its Taylor expansion.

SVMs with the kernel-trick are able to produce very good results with complex decision boundaries and can also utilise high dimensional data, however they produce black-box models, that are not easily interpreted. Furthermore, SVMs are only capable of binary classification therefore a one-vs-one approach is used during multiclass classification.

2.6.2 Random Forests

Random forests (RFs) [6] are a versatile method that create easily interpretable and quickly deployable models with very high accuracy. They were applied successfully to many domains and are capable of state-of-the-art performance. Random forests were used amongst others to classify accounts on Twitter to bots, cyborg or human [8, 41] and to identify content polluters on Facebook and Twitter [31].

Random forests are an ensemble method, that train several decision trees on random samples of the dataset and take a majority vote over all trained classifiers. Decision trees are models, that recursively split the dataset S into a set of disjoint subsets $\{S_j\} \in 2^S$, $j \in \{1, \ldots, m\}$ based on rules derived from the features. These rules are selected so that each split S_j encodes as much information about the data as possible. In order to measure the encoded information of a split, several functions can be used including the Gini impurity and information gain. The former is used in this thesis and is defined as the sum of variances over Bernoulli variables X_i with corresponding parameter p_i being the fraction of items labeled as class i in the subset S_j:

$$H(S_j) = \sum_i p_i(1 - p_i) \tag{2.3}$$

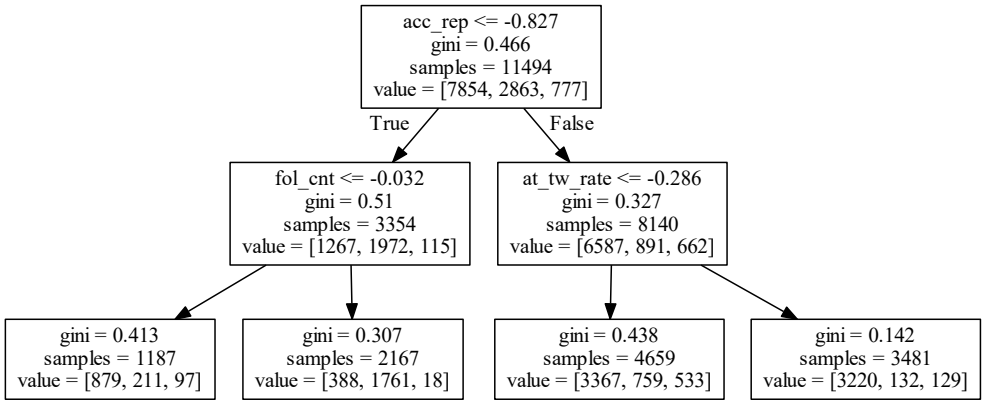

Figure 2.2: An example decision tree with maximum depth limited to 2. First it splits the data on account reputation then either on follower count or on mention count to tweet count ratio.

Each split S_j forms a leaf in the decision tree, that can be further divided until only a single element remains in S_j. This means that decision trees are able to achieve perfect accuracy on the training data, however at cost of severe overfitting. To avoid this problem the depth of the tree can be limited. Decision trees can use a simple majority vote at leaves to determine the predicted class so they are intrinsically capable of multiclass classification and they can also ignore missing data making it a very robust method. An example of a decision tree for the dataset of the thesis is given in Figure 2.2.

Chapter 3

Preliminary Analysis

The first section gives a detailed description of the dataset used in the thesis.

This is followed by an exploratory analysis of usernames with regards to naming patterns. This shows that most usernames follow four distinct naming patterns with some additional variations that are especially prevalent with genuine accounts, which try to express some level of personality with these variations. Then a syntactical analysis of usernames in the context of survivability and account types is given. The results of this section confirm, that usernames have a wide variety of information encoded in features, such as number of characters, number of terms of segmentation and types of characters used. These could allow for better discrimination both among survival states and among account types. In particular, suspended accounts show large differences in syntax compared to alive and dead accounts. The latter two follow similar distributions due to the similarity between genuine accounts and social bots.

The next section discusses the link between survivability and account types. It shows, that social bots and genuine accounts are mostly deleted and very rarely suspended and the number of alive accounts, though decreasing naturally overtime, was almost identical between the two account types. On the other hand, Twitter manages to suspend many more fake accounts and traditional bots. The figures as compared to the numbers of the original study on this dataset [13] changed over time in a way that pronounced this distinction between account types. In general, the differences between the features derived from usernames or tweets are much more pronounced for account types than survivability states, so it is important that there are directly observable connections between the two.

The initial look at usernames is followed by the results of an analysis carried out to investigate the tweeting patterns of users on Twitter in the context of survivability and account types. This section shows, that there are important differences in the way bot accounts and genuine accounts tweet both in the frequency of tweets and their temporal distributions. This result also appears when accounts are grouped according to survivability with differences between the behaviour of suspended accounts and the rest. This provides more evidence that survival states could be used to detect automated accounts.

3.1 Dataset

The basis of this analysis is the dataset published in 2017 by Cresci, et al. [13]. The data consists of a collection of four different types of users, that differ both in intent and imitation of human behaviour. Additionally, the authors have also included sets of tweets scraped in junction with the user accounts, whose counts are indicated below in parentheses. It is noted, that the original paper further categorised social and traditional spambots according to how they were attained, however here each are treated as one category as this simplifies the analysis but added no extra information regarding survivability:

1. Social spambots [*sb*]: 4918 accounts of more complex bots acting similarly to humans to influence genuine human users. (3,457,344)

2. Traditional spambots [*tb*]: 2661 accounts, that perform more easily detectable and simpler spamming activities. (6,014,982)

3. Fake followers [*ff*]: 3351 fake accounts, that the authors of [13] have purchased online and then aggregated. (196,027)

4. Genuine accounts [*ga*]: 3474 accounts, that had been confirmed to be run by real humans through a simple questionnaire in natural language. (8,377,522)

All of the data can be fully described using 37 features, such as user ID, username, account description, time of creation, etc. The original dataset was expanded with an extra feature called *state*, bringing the total number of columns to 38. This represents the survival state of the accounts as of March 7, 2020. These were identified by probing Twitter through its API and then sorting the accounts into *alive*, *suspended* or *dead* categories, that are defined as follows:

1. Alive: accounts that are still available online and are viewable publicly.

2. Suspended: accounts that have been suspended by Twitter for some reason, usually spamming, security risks or abuse.

3. Dead: accounts, that have been manually deactivated by the user and then subsequently deleted by Twitter.

For dead and suspended categories, probing the API simply returns a 404 error code, in which case simple scraping techniques were used to detect the actual state of the account. It is noted, that 1.6 million tweets among social bots are in Italian. The analysis however does not rely on features particular to only a specific language, therefore these tweets are also included in the research.

During classification the dataset was randomly shuffled and split into an 80%-20% training-testing split and for the exploration of hyperparameters 5-fold cross-validation was used on the training set. The best models were finally evaluated on the testing set.

3.2 Username Analysis

This section presents the results of the analysis carried out to better understand the semantic and syntactic structure and patterns of usernames in the context of survival states and account types.

Twitter limits the number of characters in usernames to 15 characters, which provides a hard cap on the maximum length. Furthermore, only characters matching the following regular expression are admissible in a username: `[A-Za-Z0-9_]`. As it is shown however, rich information can arise even within these constraints.

3.2.1 Semantic Information

Users regardless of account types, almost always follow some kind of naming convention that are shared across many usernames. An initial hypothesis would be that accounts choose their usernames independently of each other and these naming patterns are merely the coincidental result of the limited number of admissible characters. A closer look at the dataset shows, that this is in fact the case for genuine users, however most suspended accounts follow very systematic patterns, that imply the existence of some hidden automated naming process. An inspection of usernames in the dataset reveals, that most align with four distinct patterns. These occur across all survivability states although the frequencies differ from class to class, and with the exception of the fourth rule which occur mostly with genuine accounts. A made up example is given in parentheses for each pattern:

1. Concatenated strings with initial uppercase letters. This is the most frequently recurring scheme, that most often corresponds to the adjoining of several proper nouns or names. (JohnSmith)

2. Consecutive string of numeric digits anywhere within the username. Often this could be a birth year or a number of covert meaning. (e.g. 1337, 101, 420, etc.). (JohnSmith1969)

3. Underscores. Many usernames contain underscores, that serve as a natural separator between terms much like uppercase letters. (john_smith)

4. 'Leet speak', which is an Internet slang term, for when some letters are replaced by numbers that resemble them. For example, o = 0, t = 7, i/l = 1. (J0hnSm17h)

As mentioned, many usernames are built from proper nouns or names, or they describe some action, often related to business activities. Additionally, several encode some kind of affection such as love, hate or 'dig' (used as synonym for liking). It is finally worth noting, that most usernames of genuine accounts show some level of deviation from these patterns. Examples include adding extra letters to prolong the perceived pronunciation of a username, appending or prepending letters to signify a mood or a personality, or adding simple emojis to convey feelings such as arousal (UwU), laughter (xD) or surprise (O_O).

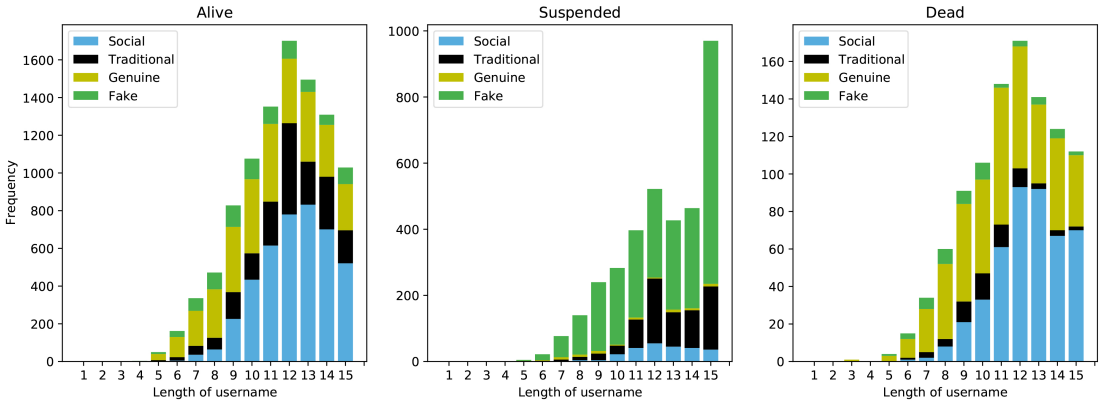

Figure 3.1: The character length of usernames for each survival state with further division into account types. Suspended account are more much more likely to have 15 character long names, while dead and alive accounts generally group around 12 characters.

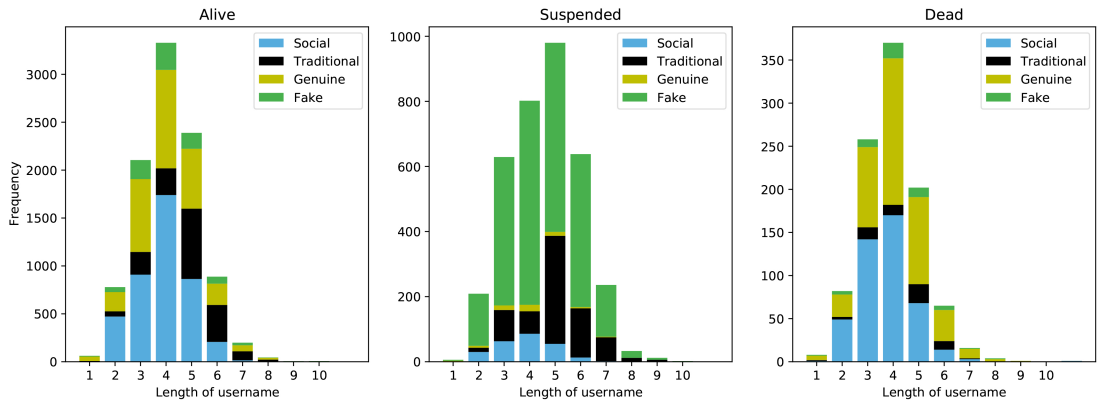

Figure 3.2: The term counts of usernames for each survival state with further division into account types. Suspended accounts are more likely to have 5 terms, while dead and alive accounts generally group around 4 terms.

3.2.2 Syntactic Information

Username Lengths. First, an analysis on the lengths of usernames is performed both with respect to individual characters and term counts of segmentations. For the latter, the standardising segmentation technique was used to split the username into terms. The complete distributions of the username lengths for every survivability type with further grouping by account types is displayed in Figure 3.1 for character lengths and in Figure 3.2 for the segmentation term counts.

The most striking difference is that suspended accounts largely prefer to use the maximum length of 15 characters. These accounts predominantly correspond to fake accounts and an inspection of the dataset reveals that most of these account have usernames which are the concatenation of a first name, a last name, and a few seemingly random letters. This pattern is also visible when looking at term counts, as the majority of suspended accounts consist of 5 terms.

Alive and dead accounts are very similar in the distribution of both character lengths and term counts. The reason for this is that social spambots and genuine accounts constitute the majority of these survival states and these accounts follow very similar distributions of username lengths while their survivability also matches. This reinforces the prediction of Section 2.3, that social bots are designed to act like humans. This also means that their survival patterns would line up with genuine accounts. A noticeable difference between dead and alive accounts is that the subset of traditional bots preferred 12 character long usernames by a significant margin. The reason for this is again similar naming patterns. For example a portion of accounts were named according to the regex: `tmj_[a-zA-Z]{3}_mgmt` which are link sharing spam accounts. It is very unlikely for any bot account to have a username shorter than 9 characters (sb: 2.5%, tb: 6%, ff: 12.6%), while genuine users more often (19.6%) name their accounts with 7 or 8 characters. In fact, this discrepancy is statistically significant with $p < 0.01$ according to a Mann-Whitney U test. Therefore, username lengths could be considered as a good additional feature in predicting survivability and subsequently detecting bots.

Finally, all Morfessor based username segmentation methods were compared to calculate the distribution of term counts in usernames. They were shown to be able to retain the distributional properties of usernames as specified by the character lengths, which provides evidence that they are able to split usernames following the correct underlying distribution.

Character Types. Another interesting property of usernames to observe is the distribution of different character types for a given survivability class. For this analysis lowercase, uppercase, numeric and non-alphanumeric (i.e. underscore) characters were considered as dictated by the username restrictions of Twitter. The results are displayed in Figure 3.3 with further divisions according to account types.

Across all survival states, both social bots and fake accounts preferred to use exactly two uppercase letters in their usernames, which is the result of the first naming pattern of Section 3.2.1. The distribution of lowercase characters mostly match the shape of the distribution of the length of usernames for each survivability type shifted left with

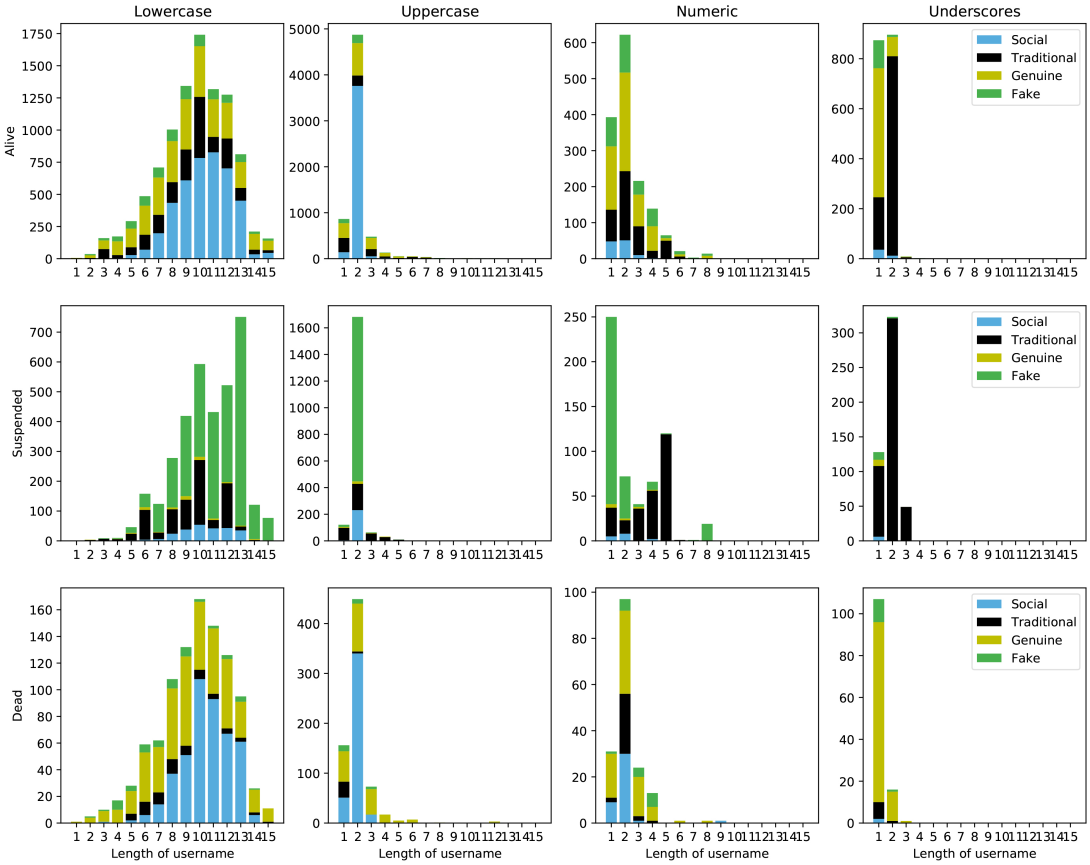

Figure 3.3: The distribution of different character types for every survivability type with further grouping into user types.

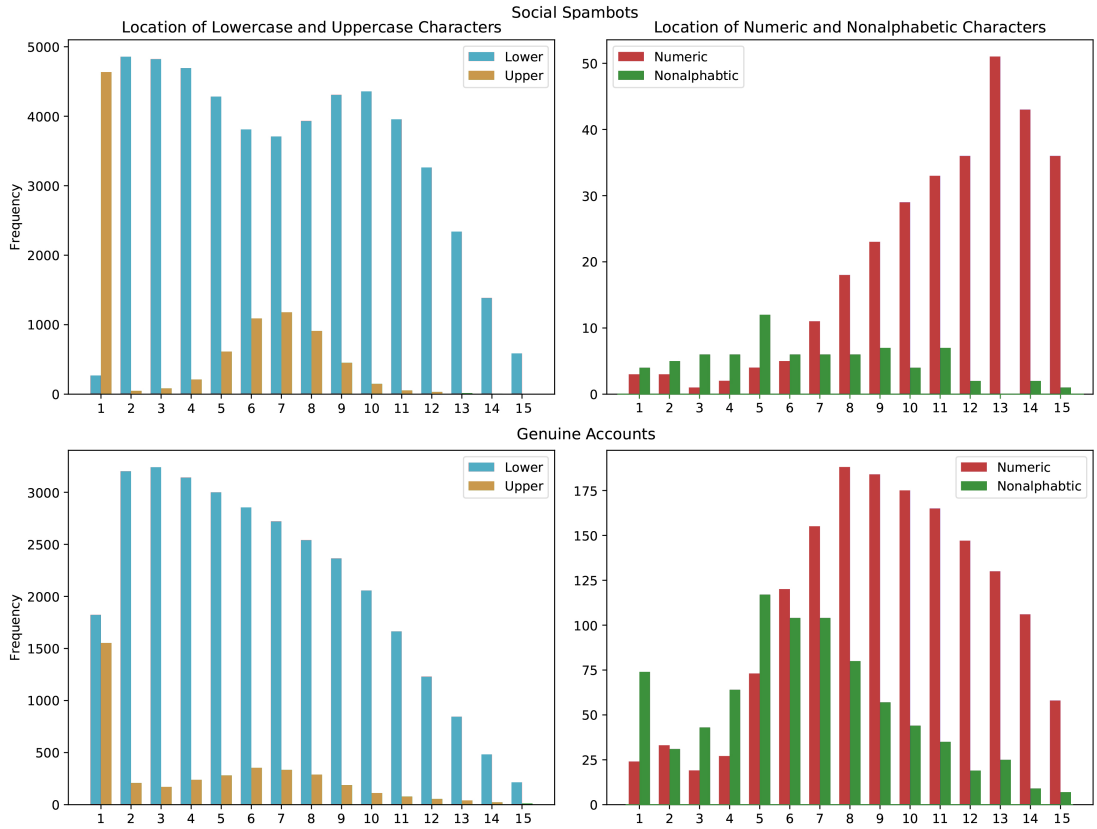

Figure 3.4: Within word location distributions of different character types for genuine accounts and social spambots

two units, which implies that the bulk of usernames are made up of lowercase letters with around two other outstanding characters. The similarities between alive and dead accounts for genuine accounts and social bots are also apparent here for all character types except underscores, which implies that the underscore could be an indicator of survival state. In addition, almost no traditional bots were deleted that used underscores but almost all suspended accounts that used underscores were traditional bots, strengthening the belief that underscores could be a differentiating factor for survivability. The proportion of deleted accounts that used underscores were interestingly highest for genuine accounts. Suspended traditional bots used more numbers in their usernames than other accounts.

Character Locations. As explained before, alive and dead accounts are largely similar as most of them are made up of genuine accounts and social bots. To find differentiating factors between these two survivability classes, it is worth comparing the locations of occurrences with regards to character types for social bots and genuine accounts only, as depicted on Figure 3.4. The difference between choices of initial letters is striking. While social spambots predominantly prefer uppercase characters as the first character with very few exceptions, genuine accounts display a much more diverse range of choices by showing an almost equal mix between upper and lowercase letters and sometimes even numeric or non-alphabetic characters. Additionally, there is an increased preference again for uppercase characters around the seventh character (and

	Total	Alive	Dead	Suspended
Social Bots	4912	4215 (85.81%)	448 (9.12%)	249 (5.07%)
Traditional Bots	2631	1814 (68.95%)	63 (2.39%)	754 (28.66%)
Genuine Accounts	3474	2968 (85.43%)	446 (12.84%)	60 (1.73%)
Fake Followers	3351	817 (24.38%)	50 (1.49%)	2484 (74.13%)

Table 3.1: Summary survival statistics for account types.

a complementary decrease in lowercase characters), which is statistically significantly more pronounced for social bots (Mann-Whitney U test with $p \approx 0$). On the other hand, genuine accounts use more non-letter characters. In general, there is a decreasing trend in number of lowercase characters for both types.

3.3 Survivability and Account Types

The previous section showed that usernames could provide important information to predict survival states and account types. This section investigates the direct relationship between survivability and accounts types to provide further evidence that survival prediction can be linked with account types. The results of this section is continuously contrasted with the previous results of [13]. As this was published in 2017, it is important and insightful to observe the changes to the original dataset and the derived statistics.

The distribution of alive, dead and suspended accounts for every account type is presented in Table 3.1. As expected, both social bots and genuine accounts have largely succeeded in persisting over time. Furthermore, the relative survival rates between the two have remained mostly unchanged, indicating that social bots managed to evade detection as effectively as genuine accounts (which ideally should not be 'detected' at all). This may indicate, as [13] have remarked previously, that the methods used by Twitter are still ineffective against advanced bots. On the other hand, Twitter has mostly managed to suspend fake follower accounts, that as a result show the lowest survival rate of all account types.

The relative differences between survivability classes compared to genuine accounts is displayed in Table 3.2. The difference between alive social bots and genuine accounts is insignificant according to a Mann-Whitney U test, and this provides more evidence, that the two classes are hard to distinguish for Twitter's methods. In contrast, the number of alive, simple malicious bots are significantly lower and the suspension rates are much higher as compared to the genuine accounts.

The relative change in survivability as compared to the results of [13] is displayed in Table 3.3. Except for fake followers, the number of live accounts has drastically decreased, which signals a positive trend for the elimination of bot accounts. Additionally, more genuine users deleted their accounts, than social bots have been removed. In fact, the significant number of deletions of social bot accounts indicate, that many of these profiles are at least partly human-run, unlike traditional bots, which have seen

	Alive	Dead	Suspended
Social bots	+0.38%	-3.72%	+3.34%
Traditional bots	-16.48%	-10.45%	+26.93%
Fake followers	-61.05%	-11.35%	+72.40%

Table 3.2: Relative survival state rates of bot accounts compared to genuine accounts

	Alive	Dead	Suspended
Social Bots	-10.55%	+9.02%	+1.4%
Traditional Bots	-19.85%	+3.45%	+15.27%
Genuine Accounts	-11.07%	+9.54%	+1.63%
Fake Followers	-1.02%	+0.39%	+0.63%

Table 3.3: Change in relative account state counts compared to the figures of [13] from 2017.

a sharp rise in account suspension but not deletion. In addition, fake followers have seen a comparatively negligible change, which is intuitive considering almost 75% had already been suspended by the time of the publication of [13].

3.4 Tweet Distributions

The tweeting patterns of an account are a huge a source of information for bot detection which has been used in studies [7, 35] to detect automated accounts with high accuracy. Therefore, building on the results of the previous section, it is worth using tweeting patterns for survivability prediction. For the purposes of this thesis a less involved approach was used to analyse the tweeting behaviour of accounts. The relative temporal differences of adjacent tweets were discretised into 22 bins. The detailed methods of discretising tweeting intervals is given in Section 4.4.1.

To visually observe the emergent patterns in tweet distributions as grouped by the different account types, the histograms of the mean temporal differences are pictured on Figure 3.5. The x-axis is on a logarithmic scale with the last column corresponding to the number of zero-distance tweets. The figure shows surprisingly that in terms of tweeting patterns, traditional bots better match the behaviour of genuine accounts than social bots. Looking at the frequency counts (y-axis) shows however that neither traditional bots nor fake followers made many posts in contrast to social bots and genuine accounts which have around ten times as many tweets. Social bots have exceedingly preferred to tweet with a frequency of about 7.5 which is equivalent to around half an hour. This behaviour is significantly different from other types according to the Wilcoxon signed-ranked test with $p < 0.001$ after normalising the data to attribute for the differences in absolute frequencies. The distribution of fake followers show a dip centered around 8.91 and local peaks at 3.26 (around 30 seconds) and 12.44 (around 3 days).

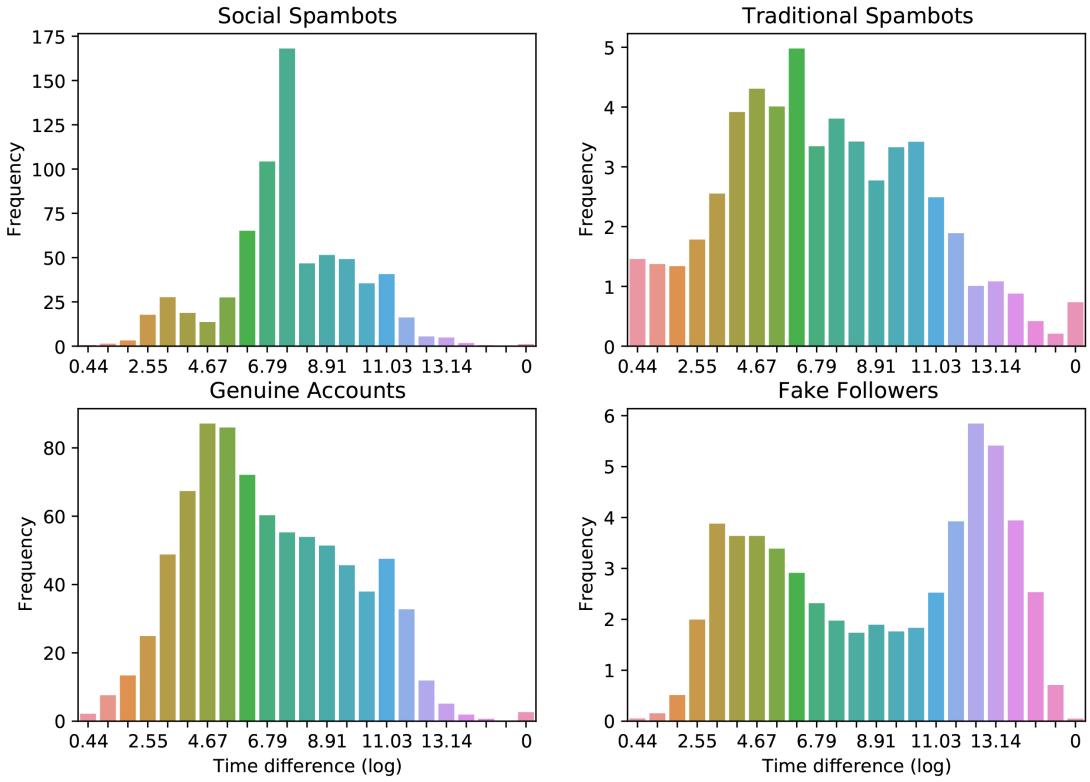

Figure 3.5: Temporal tweet distribution grouped by account type. The last column corresponds to zero-length differences. x-axis uses logarithmic scale.

On the other hand, the differences between tweeting intervals with regards to survivability is much more subtle as shown in Figure 3.6. The timing distributions of alive and dead accounts are similar with a noticeable peak at 7.5, which is also confirmed by a Wilcoxon signed-rank test with $p \approx 0.29$ after normalisation, which implies that these account behave in similar ways. However, there is a very significant distributional discrepancy between suspended accounts and other survival states (Wilcoxon $p < 0.001$), which implies that the Twitter suspension algorithm uses specific tactics related to tweeting behaviour to determine whether an account should be suspended or not.

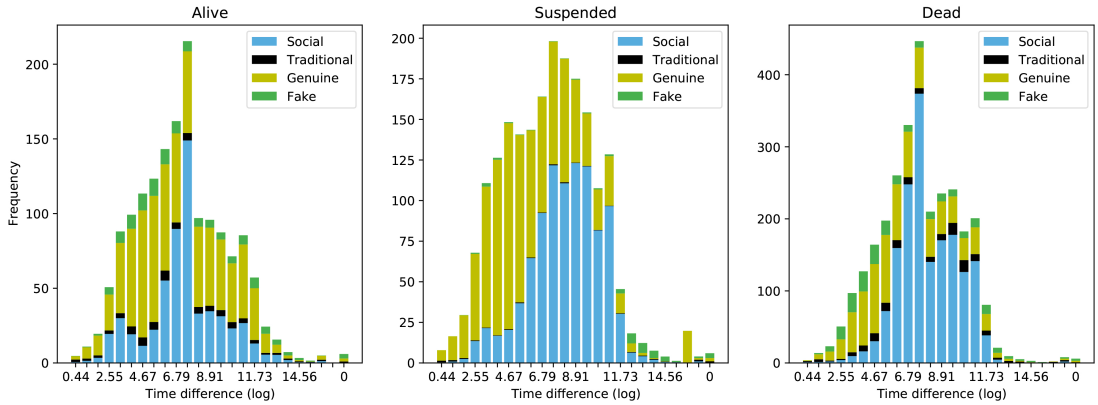

Figure 3.6: Temporal distribution of tweets grouped by survivability state. The last column corresponds to zero-different tweets. x-axis on log scale.

Chapter 4

Methods

This chapter presents the methods used to carry out the analysis of usernames and survivability including the different proposed segmentation algorithms. Then the three step approach to survivability prediction is presented. First, a detailed look at the use of segmentations for the dataset of Section 3.1 is given. Second, the application of dimensionality reduction to alleviate sparsity is explained. Third, the choice of features for classification are shown with the method of establishing a baseline and the advanced approaches used to predict the different survivability classes.

4.1 Segmentation Methods

In order to discover latent information contained in a username, it is necessary to divide it into sub-word fragments. These units encode meaningful information, that represent not only semantic content but also intra-word relationships between each sub-element. Furthermore, this step is crucial to building a classifier with a suitable complexity as the space of whole usernames is too sparse, or on the other hand, individual letters would be dense but too inexpressive.

For comparison, six different segmentation methods are considered here. As [27] argue, it is often the case that a change in letter case encodes a segment boundary (for example 'JohnSmith' $\rightarrow$ 'John Smith'), however including uppercase characters significantly increases the vocabulary size, therefore it is beneficial to encode such boundaries using a '$' sign and then turn everything lowercase. Nonetheless, for username segmentation both original and lowercased usernames are examined for comparison.

4.1.1 N-gram

The simplest approach is to split a given username into N-grams or N character long segments. This is a very simple approach proposed by [27] for usernames, that could serve as a suitable baseline during classification. In order to encode more semantic information, overlapping N-grams can be implemented with an overlap of $0 \leq M < N$ characters. For this thesis, N-grams with $N \in \{3,4\}$ were used in addition with an overlap of $M = 2$. These choices were found to produce a denser feature space

with more encoded semantic information due to the overlaps. An example of such segmentation for $N = 4$ and $M = 2$ for the fictitious username 'JohnSmith' is shown below:

```
JohnSmith: [John, hnSm, Smit]
john$smith: [john, hn$s, $smi, mith]
```

4.1.2 Random

Another basic but novel approach would be to split a username of length L into N consecutive, non-overlapping segments of random length. A sub-string is split from the start with a length calculated as follows:

$$l = \text{clamp}(n, 1, L - N + 1), \text{ where } n \sim \mathcal{N}(\mu, \sigma^2) \tag{4.1}$$

where $\mu = \sigma^2 = \left\lceil \frac{L}{N} \right\rceil$ is chosen for this thesis. That is the length of random splits are distributed normally with mean μ and variance σ^2 and they are clamped to the range $[1, L - N + 1]$. Finally, the remaining part of the string is further divided recursively until N splits in total have been performed.

The splitting distribution ensures that there will be N distinct non-empty substrings of various but not necessarily unique lengths by the end of the splitting process. The clamping ensures, that the splits are at least one character long but not so long, that they leave not enough characters for the remaining $N - 1$ splits. Overlaps are redundant to consider for random splits, as one could simply run the algorithm multiple times to retrieve different segmentations for the same username, effectively creating several overlaps. An example case for this approach with $N = 4$:

```
JohnSmith: [Joh, nS, mit, h]
john$smith: [joh, n$sm, i, th]
```

4.1.3 Informed

This method is based on the syntactic and semantic information inherent in usernames, that was discovered during their analysis presented in Section 3.2.1. The most important three components contributing to the design of the informed segmentation method are:

1. Usernames tend to be built out of names, that are delimited by a change from lowercase to uppercase characters.
2. Usernames usually contain numbers that are often relevant or significant to the holder of the account.
3. Username parts are regularly split into sections with underscores.

As this algorithm relies on information specific to the dataset of Section 3.1 it cannot technically be considered a general approach, however it is straightforward to tailor the rules to observations from other datasets, as it only relies on regular expressions. Using the previously listed patterns, a given string is split according to the following regular expressions:

1. Split at transition from lowercase to uppercase letter:
   (a) Without lowercasing: `([A-Z][a-z0-9_]+)`
   (b) With lowercasing: `(\$[a-z0-9_]+)`
2. Split at transition from letter to number: `(\d+)`
3. Split at underscore: `(_)`

Advantages of using informed segmentation are its simplicity, explainability and efficiency. A shortcoming however, is that it cannot split all lowercase/uppercase usernames into sections. For instance, it fails for examples such as `iloveyou` and `WINMONEYNOW`. For this reason a fourth algorithm, Morfessor is considered. An example using informed segmentation is presented here:

```
JohnSmith: [John, Smith]
john$smith: [john, $smith]
```

4.1.4 Morfessor

To overcome the limitations of the informed method morphological segmentation was used on usernames with Morfessor as described in Section 2.4.

Traning. First, a pre-trained semi-supervised model by [46] of the English language was trained on 384,903 expressions. This gives a basis model, that has a fundamental knowledge of English word constructions. Using the basis model with transfer learning a username-specialised model was trained for both mixed case and lowercased data. A set of 14,365 usernames was extracted from the original dataset. Transfer learning was chosen as the number of available usernames was relatively low for Morfessor to perform well. The original training-testing split was retained, however the testing set was further split as shown below. The number of usernames in each dataset split is indicated in parentheses:

- Unannotated training set: 80% of the original data used for unsupervised training. (11,494)
- Annotated training set: 10% of the original data manually annotated to be used for semi-supervised training. (1,437)
- Validation set: 5% of the original data manually annotated and used to fine-tune the corpus weight α. (718)
- Test set: 5% of the original data manually annotated used for testing the final models. This set was also used to evaluate other segmentation methods. (719)

Annotation was done by hand using the observed patterns within usernames as described in Section 4.1.3, the English grammar and the user-specified profile descriptions of the corresponding Twitter account to guide splitting in cases where usernames included fictional words. Semi-supervised training was particularly beneficial in this case, as the number of available usernames was relatively low compared to the base

	Basis	Mixed	Mixed Validation	Lowercase	L.-case Validation
F_1-Score	0.333	0.571	0.641	0.537	**0.667**
Precision	0.458	0.413	0.652	0.414	**0.662**
Recall	0.263	**0.928**	0.630	0.763	0.673
Vocab Size	17,978	**12,925**	14,854	14,077	14,838

Table 4.1: Initial test results for Morfessor models: basis, mixed case with no validation, mixed case with validation, lowercase only and lowercase only with validation

model expression count, therefore annotated data could provide convergence to better segmentation models.

Evaluation. To determine the models best suited for classification, an evaluation based on the F1-score and the generated bag-of-words (BoW) vocabulary size was carried out using the testing set. Importantly, achieving well on these measures does not directly translate to better classification results later in the analysis, therefore several good models are saved for later comparison. This is due to the large variability within the username segmentations.

The used metrics are defined as follows. The F_1-score is based on the notion of precision and recall which are calculated on the morpheme boundaries (i.e. the locations of splits) within a segmented word according to the definitions of [46]:

$$P = \text{precision} = \frac{\text{\# correct boundaries}}{\text{\# all boundaries}} \tag{4.2}$$

$$R = \text{recall} = \frac{\text{\# correct boundaries}}{\text{\# all correct boundaries}} \tag{4.3}$$

$$F_1 = \frac{2PR}{P+R} \tag{4.4}$$

In this use case, precision and recall have an intuitive mutual interpretation. Namely, if $P > R$ then the words are likely undersegmented and if $R > P$ they are likely oversegmented. Therefore, optimising the corpus weight with a validation set such that $P \approx R$ aims to balance segmentation complexity. Finally, it is crucial to have a small enough vocabulary to avoid the data becoming too sparse, therefore the models are also evaluated on the generated BoW vocabulary size using all usernames in the dataset. This does not introduce bias as no hyperparameters are tuned based on vocabulary size. It is used simply as a model selection criterion after the parameters have been fixed.

Evaluation was first performed for both mixed case and lowercase variations of Morfessor with and without validation to match P and R. Initial results presented in Table 4.1 showed, that mixed case produces very large vocabulary sizes without achieving the highest F_1-score, therefore they will be excluded from analysis further in the thesis. Furthermore, the lowercase validation method achieving the highest F_1-score produced a massive vocabulary. To find a denser model, a grid search was performed over the corpus weight α and annotation weight β as described in 2.4 with

	Base	Minimise	Standardise
F_1-score	**0.536**	0.449	0.47
Precision	**0.391**	0.293	0.316
Recall	0.852	**0.959**	0.917
Vocab Size	7,816	**7,542**	9,795

Table 4.2: Test results of the best performing model ($\alpha = 0.3, \beta = 25$) on lowercase data obtained with a combination of grid search and applying a modified informed split.

$\alpha \in \{0.1, 0.2, \ldots, 1.0\}$ and $\beta \in \{5, 25, 50, 250, 500\}$ covering a huge range of combinations for the parameters. This search gave the best combination of $\alpha = 0.3$ and $\beta = 25$ with $F_1 = 0.536$ and test results shown in the first column of Table 4.2.

Morfessor is able to split usernames, where informed segmentation fails, however it takes careful calibration for it to perform well. On the other hand informed split works very fast without much testing. Contrived examples for the effectiveness of Morfessor are presented here:

1. Heavily conjugated expression: `pretendering27: [pre, tender, ing, 2, 7]`
2. Lacking syntactical information: `WINMONEYNOW: [win, money, now]`

4.1.5 Combining Segmentation Methods

As each segmentation method has both advantages and drawbacks, it is useful to consider different combined approaches. For basic splitting methods a combination of N-grams/random and informed segmentation are not expected to produce considerable improvement over the individual methods as the N-grams and random splits largely remove the syntactic information that the informed method would rely on. The same argument also holds for a combination of N-gram/random and Morfessor.

Two novel variations of the informed segmentation method and Morfessor was also tested. In the first instance, informed segmentation was used to *minimise* the vocabulary size by splitting all non-letter characters. This created a vocabulary containing all base-10 digits, the underscore, the $ sign and words that use the English alphabet. In the second instance, informed segmentation *standardised* the generated Morfessor vocabulary by moving all dangling $ signs to the beginning of the following term and merging all consecutive numbers. Standardisation produced a vocabulary that follows the rules and patterns outlined in Section 4.1.3 for informed segmentation. These methods gave a consistent BoW vocabulary, however as Table 4.2 shows, they performed worse on the test set as compared to the unmodified approach. Nonetheless, this consistency in the vocabulary is highly desirable therefore they are retained for use during classification.

4.2 Username Segmentations

The first step in predicting the survivability of accounts. The usernames of each account is split using the different segmentation methods of Section 4.1. First, a bag-of-

Segmentation	Original	LSA
N-grams ($N = 3, M = 2$)	15,888	3,624
N-grams ($N = 4, M = 2$)	*30,941*	*8,619*
Random ($N = 3$)	*25,530*	*10,010*
Random ($N = 4$)	23,264	8,217
Informed	18,674	9,804
Morfessor ($\alpha = 0.3, \beta = 25$)	7,816	2,624
Combined (minimise)	7,542	2,372
Combined (standardise)	9,795	3,399

Table 4.3: Size of generated BoW vocabulary and the corresponding LSA space dimensions, that accounts for at least 90% of all variance in the data. Models discarded from further analysis are highlighted.

words vocabulary is calculated over the whole dataset of lowercased usernames and for each segmentation the *term frequencies* within that username is recorded in the corresponding column for the vocabulary. 8 models as described in Table 4.3, were tested to generate different vocabularies of which 6 were retained for further classification and evaluation.

Due to large vocabulary sizes, the sparsity of the BoW features may become a limiting problem during the training of a classifier, such as SVM, therefore linear dimensionality reduction was also performed. Similarly to [47], latent semantic analysis (LSA) [18] was used through the implementation of truncated singular value decomposition (SVD) of the Python package scikit-learn[1] [38]. LSA as described in Section 2.5.2 is a specialised variant of the more general principal component analysis (PCA) and was specifically designed to be used in natural language processing settings to find low-dimensional latent representations of sparse feature spaces such as the term-document matrix, which is the vocabulary of the tweets in this case, and to encode interdependence between terms. It uses SVD to find a good projection onto a low-dimensional space.

The number of components for the projections were chosen, so that they cumulatively account for at least 90% of the total variance within the data. Unfortunately, the original data is so sparse due to the uniqueness of usernames, that even with LSA a large number of dimensions remain. The original BoW vocabulary dimensions and the chosen LSA dimensions are shown in Table 4.3. The highlighted methods are excluded from further analysis. When using LSA-reduced data during classification the original segmentation data was not used, only the LSA reduced feature space.

N-grams with $N = 4, M = 2$ and random splits with $N = 3$ were dropped from further investigations due to the massive vocabulary size and the resulting sparsity. It is interesting to note, that in the case of informed splitting LSA needed almost 50% of all original variables to achieve a 90% variance retention and around one third of all vari-

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.TruncatedSVD.html`

N-gram ($N = 3, M = 2$)	892
Random ($N = 4$)	76
Informed	35
Morfessor ($\alpha = 0.3, \beta = 25$)	79
Combined (minimise)	69
Combined (standardise)	79

Table 4.4: Final dimensions of the data before classification as a result of PCA dimensionality reduction.

ables for other datasets, which shows that these methods encode usernames in a highly varied ways. Finally, in order to quantify the representative power of the username segmentations, classifications based purely on the BoW models (both LSA reduced and original) were also carried out.

4.3 Dimensionality Reduction

As it was shown, sparsity of the data is a recurring issue with the use of username segmentations, which makes the second step of the analysis, dimensionality reduction, necessary. For the encoding of usernames as mentioned in Section 4.2, LSA was used.

Concatenating the LSA reduced data with the rest of the features still resulted in a large dimensional dataset. However, these features are no longer sparse, so PCA could be used to bring the data to more manageable sizes. The final dimensions of the data were chosen so that PCA would retain at least 95% of the total variance in the data. Using this method results in vastly smaller datasets. A drawback of dimensionality reduction however is that these features are not explainable, therefore it is necessary to perform classification on the original data as well. The exact chosen dimensions are shown in Table 4.4.

Looking at this table it is worth noting, that most methods, except for N-grams had seen an extremely large reduction in feature dimensions, which suggests that the original data indeed lies on a very low dimensional manifold. Moreover, the number of chosen dimensions are similar between each method (except for N-gram), which also implies that the segmentation methods had originally encoded the username data in a similar manner. However, the N-gram method still retained a large portion of its original size implying that this method had significantly larger variances across its features. This is not a surprising result considering, that the N-gram method splits all usernames irrespective of semantics so each username would likely correspond to a set of unique features.

On the other hand, all advanced methods (i.e. informed, Morfessor and their combinations) are using similar rules for segmentation, which explains the similar low-dimensional size. Interestingly, PCA for the random method produced a close low-dimensional manifold to advanced methods. As this segmentation is based on normally distributed splits, it suggests that the more informed methods would also be

splitting according to a normal distribution. Finally, the largest reduction in feature space (around 99.7%) was seen with informed split which could create a dataset with less features than the 49 metadata features.

4.4 Classification

This section describes the third and final step of survivability prediction: the classification methods used to predict the survivability of a given user based on username, account metadata and tweet distribution. First, the features of the model is described in detail, then a description of the evaluation metrics is presented. Finally, a baseline and the usage of SVMs and random forests are given.

4.4.1 Features

There are a total of 49 basic attributes extracted from the dataset and a large BoW username-generated vocabulary of varying size that is based on the used segmentation algorithm. In addition, there are two target labels. The features fall into four major categories: account related metadata, username segmentations, tweet distributions and tweet related metadata.

Many features were adopted from previous works on Twitter account classification in particular from [47] and [16]. The novel features presented here are username segmentations and discretised tweet distributions. A complete list of all used features are given in Table 4.5.

Profile Metadata. This category is used to represent basic attributes of the given user account, that is readily retrieved with the Twitter API. These features include obvious measures such as number of friends, followers, favourites, the number of appearances in lists, username length, etc. Account reputation for a given account a was proposed by [8] and stands for the ratio between the number of followers to the total number of relationships. Intuitively, the closer this number is to 1 the more followers the account has and the more famous it can be said (e.g. @BillNye has $r_a \approx 1$), while $r_a \approx 0$ would mean that the account has very few friends but follows many accounts which is a characteristic of many bot accounts. Account reputation is calculated according to the following formula:

$$r_a = \frac{\#\text{followers}}{\#\text{followers} + \#\text{friends}} \tag{4.5}$$

Furthermore, the length of usernames are used as a feature both using character level division and using segmentation to maximise the available semantic information. To be able to better compare fields after classification, all attributes were standardised to mean zero and standard deviation one by subtracting the means and dividing by the standard deviations of each column. This step is also necessary later for dimensionality reduction to work effectively as PCA is sensitive to the scale of the data.

Tweet Metadata. Tweet metadata is extracted to encode basic tweeting patterns of a given account including average number of tweets with links or mentions, mean number of emoticons per words and the retweets to tweets ratio. Figures such as average

Profile Metadata (13)
#followers, #friends, #tweets, #favourites, #listed, account reputation, name length in chars, username length in chars, username length in words, bio length in chars/words, days since creation, average #tweets per hour
Tweet Metadata (13)
average tweet length in chars/words, retweet to tweet proportion, uppercase words/emojis/mentions/hashtags/URLs to words rate, punctuation to words rate, proportion of tweets with URLs/emoticons/hashtags/mentions
Tweet Distributions (23)
discretised counts of relative time differences between consecutive tweets using 22 bins, #zero distance tweets
Username Features (depends on segmentation)
vocabulary representation of usernames given a certain segmentation algorithm and using term frequencies
Labels (1)
survivability state (alive, dead, suspended)

Table 4.5: List of all features used for classification with the number of features in parentheses. (The '#' sign stands for 'number of'.)

number of URLs, mentions or retweets per tweet tend to be positively correlated with bot accounts [35]. The tweet tokenizer of the NLTK Python package[2] [33] was used to split each tweet into terms, that takes unique properties of tweets into account such as emoticons and hashtags. Similarly to profile metadata, tweet metadata fields were standardised.

Tweet Distributions. All tweets of a given user are sorted into 22 disjoint bins and then counted. Each bin represents an interval corresponding to the amount of time in seconds that has passed since the previous post for a tweet. The time intervals are represented on a logarithmic scale, as the time differences between consecutive tweets covered 9 orders of magnitude with the largest difference being 113,839,803 seconds (3.6 years). The edges of bins were calculated as the average of the edges over all accounts weighted by the the number of tweets made by the corresponding account.

Furthermore, on several occasions the time difference between two consecutive tweets was zero. As the logarithm is undefined for zero, these were filtered out, but their counts are stored in a separate variable as they could be indicative of automated behaviour. 22 bins were chosen, as this is the average, over all accounts, of the number of bins produced by the automatic histogram function of the Numpy Python package. This uses heuristics, in particular a combination of Sturges' Rule [44] and the Freedman-Diaconis estimator [21] to determine the optimal bin count[3].

[2] `https://www.nltk.org/api/nltk.tokenize.html`

[3] `https://docs.scipy.org/doc/numpy/reference/generated/numpy.histogram_bin_edges.html`

Encoding tweet frequencies this way provides a good basis of differentiation between user accounts as bots tend to tweet or retweet in rapid succession or following systematic patterns, while genuine accounts are largely erratic. Tweet distributions were standardised similarly to tweet metadata and profile metadata.

Username Segmentations. As described in Section 4.2.

Labels. The survival state (alive, suspended, dead) of each account is used as the label that needs to predicted.

4.4.2 Evaluation Metrics

For the effective comparison of models and algorithms it is important to establish evaluation metrics. The thesis reports three figures calculated based on the confusion matrix of the classification. A confusion matrix is simply a tabular display of the predictions of a given classification algorithm, where rows correspond to the ground-truth and columns represent the predictions. For the binary classification case the confusion matrix looks like Table 4.6. In this case, precision is calculated as $P = \frac{TP}{TP+FP}$ and recall as $R = \frac{TP}{TP+FN}$. The three metrics utilised in this thesis are widely used in many fields and are well studied.

- Accuracy: The ratio between the number of correctly labeled data to the total number of the data. This reports the overall performance of the system but often hides important information about the distribution of the classifications. Because of this additional metrics have to be considered.
- F_1-score: The harmonic mean between precision and recall: $F_1 = \frac{2PR}{R+P}$. This score is the same as given in Section 4.1.4, however in non-binary classification settings the *weighted-macro* F_1-score is calculated which is a weighted average of F_1-scores considering only a single label at a time with the class priors used as the weights. Weighting the score this way informs the metric about class imbalances. It also allows the comparison to the relevant previous work of [47].
- False Positive Rate (FPR): The ratio between the number of false positives and the total number of negatives. It is crucial in this domain to minimise FPR as flagging real humans as bots would be a hugely undesirable outcome. Calculated as $FPR = \frac{FP}{FP+TN}$. For non-binary classification the weighted-macro average is calculated.

In order to facilitate easy comparison an aggregate metric is also calculated based on the previous numbers, given as $S = \frac{1}{6}Acc + \frac{2}{6}F_1 - \frac{3}{6}FPR$. This measure takes into account the importance of different metrics and places more weight on a larger F_1-score and a smaller FPR. This score is used during hyperparameter optimisation to select the best performing model and ranges in $[-0.5, 0.5]$.

4.4.3 Models

For the classification of tweets several standard algorithms are considered in addition to a baseline setup. These methods are all supervised learning algorithms, that have an

		Prediction	
		P	N
Truth	P	TP	FN
	N	FP	TN

Table 4.6: Binary confusion matrix

incredibly rich literature and are established as state-of-the-art across many domains. The non-baseline algorithms used here are SVM and random forests. The former is preferred for its stronger modelling capabilities while the latter is preferred as it gives interpretable models. For the optimisation of the hyperparameters grid searches were performed. These were always conducted using the training data and 5-fold cross-validation.

During the selection of models for classification the multinomial Naive Bayes model was also tested for classifying accounts based purely on their usernames using the original segmentation data. This is a natural choice for a model using BoW-based data representations, however the multinomial Naive Bayes method produced test results worse than the baseline on all datasets, therefore its use was not pursued any further.

Baseline. To establish a very simple baseline model, the most frequent label was selected from across all the labels and assigned as the class of all instances. As the labels are the same amongst the datasets, the baseline results are the same irrespective of the chosen segmentation method.

SVM. The SVM model presented in Section 2.6.1 was used with the radial-basis function (RBF) kernel as the black-box model in this thesis due its good performance and established use in existing literature as discussed in Section 2.6.1. The hyperparameters of the model, namely γ and C were selected using a grid-search over the geometrically spaced interval $[10^{-3}, \ldots, 10^{3}]$ with quotient 10. This grid-search was performed for each dataset which gave $C = 10$ and $\gamma = 0.1$ for all datasets.

Random Forest. The random forest architecture as described in Section 2.6.2 was used due its speed, compactness, high performance and simple interpretability. A crucial advantage of this model is its explainability, so not only the PCA reduced datasets but also the original datasets were used, where the features are not obfuscated by PCA. During training of the random forest 100 decision trees were grown to produce the ensemble. The hyperparameter of this model is the maximum tree depth, which was optimised using grid search. As the number of columns differ for each dataset, the values searched over also differ. For datasets with less than 100 features every depth in $[2, \ldots, n]$ were checked where n is the number of features. For the datasets with $n > 100$ features, a linearly spaced set of numbers were tested with a difference of approximately $\frac{n}{20}$ so that at least 20 unique values were tested .

Chapter 5

Results

This chapter presents the results of classifying users to different survivability states using the three step approach of Chapter 4, with the examination of several combinations of the features that were described in Section 4.4.1 to assess the effects of username segmentations and dimensionality reduction on classification performance. This section shows, that usernames are able to provide increased performance in predicting the survival state of accounts compared to the baseline method, however due the uniqueness and sparsity of username generated feature space, combining the segmentations with metadata based information results in decreased performance, which is a result of increased variance in the features. The section also shows that dimensionality reduction is a very useful tool for the speedup of training at very little cost to classification performance. The analysis presented here is closest to the work of [47], who performed a similar survivability analysis and their results are contrasted with the figures presented here.

A summary of the tested feature combinations and the resulting modelling choices with the produced metrics are shown in Table 5.1. In total six feature combinations were tested:

- Baseline (*Baseline*): Uses only the target labels to calculate their prior probabilities, while ignoring everything else.
- Segmentation only (*Segmentation*): Uses only the segmentation of usernames with no dimensionality reduction.
- Reduced segmentations (*LSA*): Same as *Segmentation* but uses LSA to compact the data as much as possible.
- Metadata Only (*Metadata*): Uses only the 49 hand picked features described in Section 4.4.1 without dimensionality reduction.
- Non-reduced all (*All*): Combines all available features as described in Section 4.4.1, but does not apply dimensionality reduction.
- LSA-PCA all (*LSA-PCA*): Similar to *All* but applies LSA and then PCA to try to compact the data as much as possible.

	Feature Set					
	Baseline	Segmentation	LSA	Metadata	All	LSA-PCA
Segmentation	—	Min.	Min.	Min.[1]	Inf.	Inf.
ML Method	—	RF	RF	RF	RF	SVM
Hyperparams	—	1,510	1,300	28	13,106	C: 10; γ: 0.1
Accuracy	0.689	0.728	0.719	**0.878**	0.877	0.868
F_1-score	0.562	0.665	0.638	**0.862**	0.860	0.854
FPR	0.688	0.554	0.600	**0.226**	0.237	0.234
S-score	-0.042	0.065	0.032	**0.321**	0.314	0.312

Table 5.1: Summary table of the best performing models across all tested feature sets with the used algorithms, hyperparameters (for RF the maximum depth, for SVM C and γ) and the produced metrics.

These feature sets cover a wide range of combinations and allow a detailed comparison of the efficiency of modelling account survivability based on usernames.

5.1 Baseline

As the classification of Twitter accounts to survivability classes based on usernames has not yet been previously explored it is important to establish a credible baseline. The method adopted in Section 4.4.3 uses a majority label prediction. This baseline produced results as shown in Table 5.1. The majority label predicted is the *alive* state and 68.9% of all accounts in the testing data fell into this category as indicated by the accuracy score. Approximately two thirds of all accounts are false positives as indicated by the weighted macro-FPR.

5.2 Segmentations Only

As mentioned in Section 4.2 it is important to see how well usernames on their own predict the survivability of accounts. To this end, classifications are carried out based purely on the BoW features generated by segmentation methods from usernames. As a result of data sparsity, SVMs took excessively long to train which prohibited the systematic testing of these datasets, therefore only the random forest method was used. After testing all segmentation methods and running grid search, the best performing model was found to be the minimising combined model, that created a BoW vocabulary with 7,542 unique terms. Using this model gave a S-score of 0.065 and an F_1-score 0.665 using a maximum depth of 1,510. This is an improvement over the baseline, that is also statistically significant according to a Wilcoxon signed-rank test ($p < 10^{-52}$), however it is worse than most results of [47] and as shown below additional features can greatly improve the performance of the classification.

[1] Only used to count how many terms there are in a username. Metadata is otherwise independent of segmentation methods.

Additionally, it is worth investigating which features were selected first during the growing of the decision trees to establish an ordering of importance of features. The most important features were usually the most general ones with features corresponding to terms such as individual digits, the dollar sign or the words 'reward' and 'jobs'. This reinforces the expectation that individual names are highly unique therefore common features would be the most discriminating.

Finally, as mentioned in Section 4.4.3, a multinomial Naive Bayes model was also tried on solely the username segmentations, however this produced worse results than the baseline (~ 0.62 accuracy) for all datasets. Adding new features based on the Naive Bayes prediction also deteriorated performance.

5.3 LSA Reduced Segmentations

The previous section showed, that username segmentations are a viable addition to improving classification performance, however the number of features greatly restrict the choice of algorithm used to perform that classification. To solve this problem, LSA dimensionality reduction is used. In order to show, how this transformation affects the results, classifications based purely on LSA reduced data was investigated. As shown in Table 5.1, LSA reduction resulted in decreased performance due to the information lost, however it still performed better than the baseline (Wilcoxon test $p < 10^{-31}$) with an S-score of 0.032 and F_1-score of 0.638.

5.4 Metadata Only

It is important to measure the performance of using solely account and twitter metadata for classification to be able to compare with models that also use username segmentations. Both SVMs and random forests were tested on the hand selected 49 features, that encode various metadata about an account and its tweeting behaviour as presented in Section 4.4.1. Metadata information is almost identical across username segmentations, except for the feature 'length of username in words' as this field depends on the used segmentation method. This feature set gave the best results of all tested models overall using a random forest of depth 28, that gave a S-score of 0.321 and a F_1-score of 0.862. The importance of the fields followed a very similar order as shown in Table 5.2 for the *All Features* set.

5.5 All Features

Only random forests were used for training on the original data with every feature because of the previously mentioned slowness of SVMs. This feature set gave the best results of all tested combinations of features that included usernames, which is to be expected as no information is lost through dimensionality reduction. It gave a S-score of 0.314 and a F_1-score of 0.860. It used the informed username segmentation with a maximum decision tree depth of 13,106. Furthermore, this result is very close to the

Overall	Segmentation Only
Age in days, Account reputation, Tweets per hour, Statuses count, Friends count, Followers count, Hashtag tweet rate, Biography length in characters, Average tweeting frequency of 13 seconds, Punctuation rate in tweets	Dollar sign ($), Underscore (_), Letter 'q', Word 'reward', Digit 1, Letter 's', Digit 2, Letter 'j', Letter 'a', Word 'jobs'

Table 5.2: The top-10 most important fields of the best performing random forest using the *All Features* feature set and the top-10 most important username segmentation features.

best result of [47] in this category who achieved a highest weighted F_1-score of 0.87 using syntactic and stylistic features extracted from tweets.

The top ten most important features of the resulting random forest is shown in Table 5.2. This shows, that metadata fields were preferred overall before username segmentations. In particular, the age of the account and the account reputation was most influential, when choosing a survivability class. Excluding these metadata fields gave an importance ordering of only segmentation features, which follows a similar ordering to that discovered in Section 5.2.

5.6 LSA-PCA

Reducing the data using LSA and PCA made the SVM classifier possible to use, which indeed gave the best results in this category. After running grid search the best hyperparameters were found to be $C = 10$ and $\gamma = 0.1$ for every segmentation method. Contrary to expectations, the drastic reduction in the number of dimensions barely affected the performance of the model. In fact, the results of the best SVM model that uses the informed dataset is not statistically significantly different from that of the best segmentation using model (RF on all features using minimised segmentations) according to a Mann-Whitney U-test ($p > 0.1$).

Random forests were also examined for the LSA-PCA feature set and were found to produce similar accuracies and F_1-scores as their SVM counterpart, however with significantly worse false positive rates. In fact the best RF model gave $FPR = 0.245$, which is as 'good' as the worst SVM model. Furthermore, the explainability of random forests is lost when applied to PCA reduced data, therefore there is little benefit to using them for this feature set.

5.7 Results

The results of the analysis show that username segmentations are able to improve prediction accuracy over the baseline with best results obtained using an uncompressed

segmentation based on the minimising combined model. However, they are not capable of achieving acceptable results in comparison to any model relying on metadata. In fact, the false positive rate of these models are almost as large as the baseline which is clearly a huge issue.

There are several reasons for the inefficiency of usernames in the prediction of survivability. The most severe problems are related to the way segmentations are encoded for classification. Bag of words is used in the thesis, which works well when there is a large corpus of documents available, however it is not suited to encode the highly unique terms generated from usernames without the availability of a massive dataset to compensate for the sparsity and large variances across the feature space.

Another issue is that usernames only seem to be indirectly related to survivability through account types. Throughout the analysis of Chapter 3, usernames were seen to be better at discriminating among account types than among survival states. Though Section 3.3 does establish a direct link between survivability and account types which explains the improved prediction performance of username segmentations over the baseline, however the most important information extracted from usernames, such as length or naming patterns only directly relate to account types. Using BoW and segmentations amplifies this disconnect between usernames and survivability as it removes information by e.g. omitting the orderings of terms. Finally, the BoW representation of a username does not take into account any text-based context from the account, which could enrich its username's representation.

Nonetheless, usernames were able to improve classification accuracy over the baseline so it is reasonable to think that a more complex encoding, which takes into account more contextual information such as the account biography or the text of tweets, would give a richer representation and a better classification performance. Although usernames affected performance of models built on metadata in a negative way, the effect is minor and mostly due to the lack of data and the increase in variance that usernames introduce. This means that even with a simple representation such as BoW it would be possible to improve performance given more data. Additionally, developing a better representation would alleviate the negative effects of BoW so better prediction performance could be expected.

On the other hand, the thesis showed that survivability is a well predictable label using metadata extracted from the account and its corresponding tweets. These features were able to achieve close (around 0.9% worse F_1) state of the art performance of [47]. Incorporating usernames with richer encoding is very likely to improve on these results.

Finally, it is worth noting, that dimensionality reduction has decreased prediction performance as expected due to loss of information, however gave significant speedups during training time while the reduction in performance was not significant.

Chapter 6

Conclusion

6.1 Summary

The thesis investigated the relationship between usernames and survivability. In particular, it examined whether the survival states of accounts can be predicted based on segmented usernames.

First, the thesis presented several segmentations methods in Section 4.1. In particular: N-grams, *random*, *informed*, Morfessor and a *combination* of informed and Morfessor. The highlighted methods are novel techniques, as far as the author of the thesis is aware.

This was followed by an analysis based on a three step procedure. First, the segmentation methods were used to split usernames to produce a basis for predicting survivability of accounts on Twitter. The second step involved reducing the dimensions of the generated segmentations with LSA and PCA due to severe sparsity. In the third step, the segmentation data was combined with several other account and tweet related features to establish a connection, using classification with SVMs and random forests, between usernames and the survivability of accounts.

As the results of Section 5.2 showed, there is enough information encoded in usernames to be able to more accurately classify user accounts to survivability states than the baseline. This provides evidence that analysis using usernames is a viable research path. A mix of account plus tweet metadata and username segmentations have proven successful in classifying accounts to survivability states. Of all segmentation models tested, the informed segmentation model achieved the highest scores for complete feature sets and the minimising combined Morfessor model for only segmentation based data. This shows that using a combination of a hand-crafted segmenter with a black box model creates the best division of a username in terms of the amount of extracted latent information.

However, the username segmentation representations are very unique and sparse, which means that using them as a part of a larger dataset based on dense information introduces significant variances, that actually hurts classification performance as shown by

the results of Section 5.4. This could be mitigated with the use of more data or a richer username representation as discussed in Section 5.7.

As a bag-of-words model of usernames is by design static, this system would be difficult to extend to a useful interactive tool, that could for example, provide an online assessment of the survivability of a user account given its username. Creating a denser, more descriptive and easily computable representation would therefore be necessary.

6.2 Future Work

As it stands, using a BoW model to represent usernames for classification is sufficient to establish a connection between accounts and survivability, but it is neither efficient nor well performing. Therefore, more sophisticated models of username representation is necessary. One prospect to solve these issues would be to create an embedding of usernames with some contextual information such as account descriptions or tweets to create a low dimensional vector-representation of them similar to e.g. Word2Vec [36] or using variational autoencoders. Alternatively one could consider developing a novel mathematical model to create a representation that takes into consideration the domain specific knowledge about usernames and uses sparsity related research. A new representation could be integrated into a tool to provide interactive user survivability checking.

Bibliography

[1] Norah Abokhodair, Daisy Yoo, and David W. McDonald. Dissecting a social botnet: Growth, content and influence in twitter. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '15, page 839–851, New York, NY, USA, 2015. Association for Computing Machinery.

[2] Hazim Almuhimedi, Shomir Wilson, Bin Liu, Norman Sadeh, and Alessandro Acquisti. Tweets are forever: A large-scale quantitative analysis of deleted tweets. *Proceedings of the ACM Conference on Computer Supported Cooperative Work, CSCW*, February 2013.

[3] Eiji Aramaki, Sachiko Maskawa, and Mizuki Morita. Twitter catches the flu: detecting influenza epidemics using Twitter. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1568–1576, Edinburgh, United Kingdom, July 2011. Association for Computational Linguistics.

[4] Fabrício Benevenuto, Gabriel Magno, Tiago Rodrigues, and Virgílio Almeida. Detecting spammers on twitter. In *In Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS*, 2010.

[5] Geert Booij. *The grammar of words: An introduction to linguistic morphology*. Oxford University Press, 2012.

[6] Leo Breiman, JH Friedman, RA Olshen, and CJ Stone. Classification and regression trees. wadsworth & brooks. *Cole Statistics/Probability Series*, 1984.

[7] N. Chavoshi, H. Hamooni, and A. Mueen. DeBot: Twitter Bot Detection via Warped Correlation. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 817–822, December 2016.

[8] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia. Detecting Automation of Twitter Accounts: Are You a Human, Bot, or Cyborg? *IEEE Transactions on Dependable and Secure Computing*, 9(6):811–824, November 2012.

[9] Karen M. Cornetto and Kristine L. Nowak. Utilizing Usernames for Sex Categorization in Computer-Mediated Communication: Examining Perceptions and Accuracy. *CyberPsychology & Behavior*, 9(4):377–387, August 2006.

[10] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995.

[11] S. Cresci, R. D. Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi. Social Fingerprinting: Detection of Spambot Groups Through DNA-Inspired Behavioral Modeling. *IEEE Transactions on Dependable and Secure Computing*, 15(4):561–576, July 2018.

[12] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. Fame for sale: efficient detection of fake Twitter followers. *Decision Support Systems*, 80:56–71, December 2015. arXiv: 1509.04098.

[13] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race. *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion*, pages 963–972, 2017. arXiv: 1701.03017.

[14] Stefano Cresci, Marinella Petrocchi, Angelo Spognardi, and Stefano Tognazzi. Better Safe Than Sorry: An Adversarial Approach to Improve Social Bot Detection. *arXiv:1904.05132 [cs]*, April 2019. arXiv: 1904.05132.

[15] Mathias Creutz and Krista Lagus. Unsupervised discovery of morphemes. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 21–30. Association for Computational Linguistics, July 2002.

[16] J. P. Dickerson, V. Kagan, and V. S. Subrahmanian. Using sentiment to detect bots on Twitter: Are humans more opinionated than bots? In *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014)*, pages 620–627, August 2014.

[17] S. T. Dumais, G. W. Furnas, T. K. Landauer, S. Deerwester, and R. Harshman. Using latent semantic analysis to improve access to textual information. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '88, page 281–285, New York, NY, USA, 1988. Association for Computing Machinery.

[18] Susan T Dumais. Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230, 2004.

[19] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. The Rise of Social Bots. *Commun. ACM*, 59(7):96–104, June 2016.

[20] Peter W. Foltz, Walter Kintsch, and Thomas K Landauer. The measurement of textual coherence with latent semantic analysis. *Discourse Processes*, 25(2-3):285–307, 1998.

[21] David Freedman and Persi Diaconis. On the histogram as a density estimator: L 2 theory. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 57(4):453–476, 1981.

[22] M. Ghiassi, J. Skinner, and D. Zimbra. Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network. *Expert Systems with Applications*, 40(16):6266–6282, November 2013.

[23] Yihong Gong and Xin Liu. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, page 19–25, New York, NY, USA, 2001. Association for Computing Machinery.

[24] Samira Hassa. Projecting, exposing, revealing self in the digital world: Usernames as a social practice in a moroccan chatroom. *Names*, 60:201–209, 12 2012.

[25] Philip N. Howard and Bence Kollanyi. Bots, #Strongerin, and #Brexit: Computational Propaganda During the UK-EU Referendum. SSRN Scholarly Paper ID 2798311, Social Science Research Network, Rochester, NY, June 2016.

[26] Rodrigo Augusto Igawa, Sylvio Barbon Jr, Kátia Cristina Silva Paulo, Guilherme Sakaji Kido, Rodrigo Capobianco Guido, Mario Lemes Proença Júnior, and Ivan Nunes da Silva. Account classification in online social networks with LBCA and wavelets. *Information Sciences*, 332:72–83, March 2016.

[27] Aaron Jaech and Mari Ostendorf. What Your Username Says About You. *arXiv:1507.02045 [cs]*, July 2015. arXiv: 1507.02045.

[28] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is Twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 591–600, Raleigh, North Carolina, USA, April 2010. Association for Computing Machinery.

[29] M. E. Larsen, T. W. Boonstra, P. J. Batterham, B. O'Dea, C. Paris, and H. Christensen. We feel: Mapping emotion on twitter. *IEEE Journal of Biomedical and Health Informatics*, 19(4):1246–1252, 2015.

[30] Kyumin Lee, James Caverlee, and Steve Webb. Uncovering social spammers: Social honeypots + machine learning. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, page 435–442, New York, NY, USA, 2010. Association for Computing Machinery.

[31] Kyumin Lee, Brian David Eoff, and James Caverlee. Seven Months with the Devils: A Long-Term Study of Content Polluters on Twitter. In *Fifth International AAAI Conference on Weblogs and Social Media*, July 2011.

[32] Clare Llewellyn, Laura Cram, Robin L. Hill, and Adrian Favero. For Whom the Bell Trolls: Shifting Troll Behaviour in the Twitter Brexit Debate. *JCMS: Journal of Common Market Studies*, 57(5):1148–1164, 2019.

[33] Edward Loper and Steven Bird. Nltk: the natural language toolkit. *arXiv preprint cs/0205028*, 2002.

[34] Juan Martinez-Romo and Lourdes Araujo. Detecting malicious tweets in trending topics using a statistical analysis of language. *Expert Systems with Applications*, 40(8):2992–3000, 2013.

[35] Michele Mazza, Stefano Cresci, Marco Avvenuti, Walter Quattrociocchi, and Maurizio Tesconi. RTbust: Exploiting Temporal Patterns for Botnet Detection on Twitter. In *Proceedings of the 10th ACM Conference on Web Science*, WebSci '19, pages 183–192, New York, NY, USA, 2019. ACM. event-place: Boston, Massachusetts, USA.

[36] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[37] N. Senthil Murugan and G. Usha Devi. Feature extraction using LR-PCA hybridization on twitter data and classification accuracy using machine learning algorithms. *Cluster Computing*, 22(6):13965–13974, November 2019.

[38] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[39] Sasa Petrovic, Miles Osborne, and Victor Lavrenko. I wish i didn't say that! analyzing and predicting deleted messages in twitter. *arXiv preprint arXiv:1305.3107*, 2013.

[40] Peter J Potash, Eric B Bell, and Joshua J Harrison. Using topic modeling and text embeddings to predict deleted tweets. Technical report, Pacific Northwest National Lab.(PNNL), Richland, WA (United States), 2016.

[41] J. Schnebly and S. Sengupta. Random forest twitter bot classifier. In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0506–0512, 2019.

[42] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. Fake News Detection on Social Media: A Data Mining Perspective. *SIGKDD Explorations*, 19:22–36, 2017.

[43] Stefan Stieglitz, Florian Brachten, Björn Ross, and Anna-Katharina Jung. Do Social Bots Dream of Electric Sheep? A Categorisation of Social Media Bot Accounts. *arXiv:1710.04044 [cs]*, October 2017. arXiv: 1710.04044.

[44] Herbert A. Sturges. The choice of a class interval. *Journal of the American Statistical Association*, 21(153):65–66, 1926.

[45] Kurt Thomas, Chris Grier, Vern Paxson, and Dawn Song. Suspended Accounts in Retrospect: An Analysis of Twitter Spam. *Proceedings of ACM SIGCOMM*, page 14, 2011.

[46] Sami Virpioja, Peter Smit, Stig-Arne Grönroos, and Mikko Kurimo. *Morfessor 2.0: Python Implementation and Extensions for Morfessor Baseline*. Aalto University, 2013. Accepted: 2013-12-12T10:00:59Z ISSN: 1799-490X (electronic).

[47] Svitlana Volkova and Eric Bell. Identifying Effective Signals to Predict Deleted and Suspended Accounts on Twitter Across Languages. In *Eleventh International AAAI Conference on Web and Social Media*, May 2017.

[48] Alex Hai Wang. Detecting spam bots in online social networking sites: A machine learning approach. In *Data and Applications Security and Privacy XXIV*, pages 335–342. Springer Berlin Heidelberg, 2010.

[49] Chao Yang, Robert Harkreader, and Guofei Gu. Empirical Evaluation and New Design for Fighting Evolving Twitter Spammers. *IEEE Transactions on Information Forensics and Security*, 8(8):1280–1293, August 2013.

[50] J. Zhang, Darrell Carpenter, and M. Ko. Online astroturfing: A theoretical perspective. volume 4, pages 2559–2565, 01 2013.