University of Groningen

Master's Thesis

A supervised approach to categorizing Dutch Twitter events

Rik I. K. van Noord

March 2016

Human-Machine Communication
Department of Artificial intelligence

First Supervisor and Reviewer:
Dr. J. K. Spenader
Department of Artificial Intelligence - University of Groningen
Second Supervisor and Reviewer:
Prof. Dr. A.P.J. van den Bosch
Language and Speech Technology - University of Nijmegen
Third Supervisor and Reviewer:
F. Kunneman
PhD Candidate Centre for Language Studies - University of Nijmegen

Abstract

A supervised approach to categorizing Dutch Twitter events

by Rik I. K. van Noord

In this thesis we applied a supervised machine learning approach to automatically categorize Dutch Twitter events. One of the ten categories used is the category *social action*, which aims to predict civil unrest. Reliably detecting such events might have great practical value, since we are then able to alert the authorities when a (possibly violent) social action will take place. We employ the existing event set of Kunneman and van den Bosch (2015), who used explicit future time expressions to identify events. We show that it is difficult to categorize all events automatically, since the classifications are biased towards the dominant category *public event*. However, our general categorization system offers comparable performance to the best known approach in the literature and is even suggested to outperform that approach when categorizing the full event set of 93,901 events. We find that our final categorization system is very precise in its predictions for non-dominant categories, but that it does not offer those predictions very often. We obtained a 80% precision for detecting *social action events*, but also a low estimated recall. Due to this low recall and since we were limited to Twitter data, our approach got outperformed by the best known approach of predicting civil unrest. However, a follow-up approach that utilizes the ranking of the Bayesian probabilities increased the recall of *social action events* by 232%, while decreasing the precision by only 14%.

Acknowledgements

I would like to thank my advisors Florian Kunneman and Antal van den Bosch for their valuable feedback throughout the entire project. Florian provided great and fast help, always available for a quick chat and even mailing me back on New Years Day. I would like to thank Jennifer Spenader for helping me to focus the writing on the main issues. Many thanks also go out to everyone that helped in the annotation process. And finally, special thanks go out to my girlfriend Anna de Koster, who assisted and supported me throughout the entire project.

Contents

Chapter 1

Introduction

A lot of the instabilities across the world stem from civil unrest, often involving crowd actions such as mass demonstrations and protests. These types of events are often very dangerous, sometimes even resulting in injuries or casualties. Due to the sudden rise of social media, a lot of the discussion about these events takes place online. This became especially clear after the Arab Spring (González-Bailón et al., 2011), but the term *Twitter revolution* was already being used for the mass protests in Moldova (Mungiu-Pippidi and Munteanu, 2009) and Iran (Morozov, 2009). Since people comment on their experiences in real time, Twitter users are often the first to write about significant events in the world. Therefore, many researchers started to wonder whether these events could have been anticipated or even predicted if there was a way to utilize this Twitter information (e.g. Leetaru, 2011).

The most shining example of a mass crowd action in the Netherlands is probably the Project X party in Haren, Groningen on September 21st, 2012. A public Facebook invitation to a birthday party of a 16-year old girl ultimately lead to thousands of people rioting.[1] The riots could only be stopped by severe police intervention, resulting in more than 30 people injured[2], up to 80 arrests[3] and an estimated 250,000 euros in damages for the municipality of Haren.[4] A special committee was assigned to identify what exactly went wrong on this night and which actions should be taken to prevent such events from happening in the future. The committee (among others) concluded that the police were insufficiently prepared and that they were not well enough informed about the developments on social media, recommending a nation-wide system that is able to analyse and detect these threats in advance.[5] If such a system was already in

[1] http://www.nu.nl/binnenland/2915769/facebook-feest-haren-ontaardt-in-chaos-en-rellen.html

[2] http://www.ad.nl/ad/nl/11474/Facebookrellen/article/detail/3320601/2012/09/22/30-gewonden-na-Project-X-Haren.dhtml

[3] http://www.ad.nl/ad/nl/11474/Facebookrellen/article/detail/3343518/2012/11/06/Aantal-aanhoudingen-Haren-gaat-richting-100.dhtml

[4] http://nos.nl/artikel/454235-schade-project-x-haren-kwart-miljoen.html

[5] http://nos.nl/artikel/482043-cohen-fouten-politie-burgemeester.html

place, this would have alerted the authorities in due time, likely even preventing this event from happening at all. We will refer to these mass crowd actions relating to civil unrest as *social action events* or *social actions*. In this thesis, we create a system that will automatically predict *social action events* by only using data available from Twitter.

To be able to do this, we first have to detect events in general. We use the following definition of an event, similar to the definition in McMinn et al. (2013):

- Something is an event if it happens at a specific place, at a specific time and if it is important for a large number of people.

This includes events such as soccer matches, elections and demonstrations, while simultaneously excluding the uninteresting mundane, personal-level events. Twitter events are detected based on the observation that a set of tweets often refers to a single event in the real world. The detection method is based on the frequent co-occurrence of certain words combined with mentions of a specific date. These words are the most important words of an event and are referred to as *keywords*. A Twitter event is defined as a set of tweets with corresponding keywords that is predicted to take place on a certain date. An example of a Twitter event that is categorized as a *social action event* is shown in Table 1.1.[6]

Table 1.1: An example of a Dutch Twitter event.

Date	21-09-2013
Keywords	#demonstratie, bezuinigingen
Tweet 1	21 september: nee tegen nieuwe bezuinigingen! #demonstratie
Tweet 2	komen jullie eigenlijk ook naar de #demonstratie #21september? #actie is nodig!
Tweet 3	kom 21 september naar den haag demonstreren tegen kabinet #demonstratie
Tweet 4	het mag weer morgen, geen excuus zijn om niet naar #denhaag #verzet
Tweet 5	nog 8 dagen #pvv #demonstratie tegen #afbraakkabinet op #koeplein den haag!

The detection of events in general was already performed by Kunneman and van den Bosch (2015). Only using a large set of Dutch tweets, collected between 2010 and 2015, they identified a set of 93,901 Twitter events (henceforth referred to as events). We will use this set of events to create a system that automatically groups events into ten different categories, *social action* being one of those categories. We identify these different categories since a system that detects *social actions* should not only be reliable in detecting events where large groups of people come together, but it should also exclude

[6]The English translation of this event is shown in Table A.1 in the Appendix.

events that have large groups of people coming together due to a clear and harmless reason (e.g. soccer matches, music performances) where authorities will be sufficiently prepared for. This means that the focus will be on creating a system that is able to detect all categories reliably, also being useful for a wide range of other applications instead of just for detecting *social action events*. This way, the detection of *social action events* can be seen as an advantage of the general system as opposed to a system which aims to identify only such events.

To train a system on predicting the category of new events, it has to have access to already categorized events. Therefore, we select two subsets of events that we manually labeled by category. The first event set contains the 600 events that had the best *eventscore*. The *eventscore* is an automatically calculated measure of how certain it is that the event in question is an actual event. We are interested in whether events with a high eventscore are easier to categorize in general. This event set is referred to as the **best event set**. The second event set contains 600 events that were semi-randomly selected from the total set of 93,901 events. This is the representation of all events and will be referred to as the **random event set**. We also construct a third event set, which is a combination of the previous two event sets. This event set is referred to as the **combined event set**. This event set measures the impact of using more training data. We set the following three research goals:

- Automatically and reliably categorizing **all** events
- Automatically and reliably categorizing **the best scoring** events
- Automatically and reliably detecting **social action events**.

One of the challenges in this field is identifying the tweets that belong to a specific event. We therefore first explain how we obtained such a set of tweets for each event. Then we describe which features we extracted and subsequently used in the classification process, with the focus on inserting real world knowledge in the system by using information from Wikipedia. These features are used to guide classification of the Support Vector Machine and Multinomial Naive Bayes algorithms. We then explain a number of methods to possibly increase the accuracy of these algorithms, ranging from down-sampling the dominant class to selecting subsets of the best features to use in classification. This is followed by a test of the system on the full set of 93,901 events and an evaluation of the results by using an annotated set of events that was never used in the training phase.

We found that it is easier to classify a small subset of events with a (very) high eventscore, but that there is no effect of eventscore in general. We then found that it is difficult to reliably predict **all** events, but that it is possible to construct a system that has a

high precision, although this means that the estimated recall is low. This is also the case for *social action events*, meaning that if the system predicted a *social action* it did so correctly. However, it also failed to recognize many *social action events*. This is not preferable, since we rather have a higher rate of false alarms if that means that the authorities are always warned when an actual *social action* is taking place. We therefore applied a follow-up method that exploits the actual probabilities of classification. This resulted in a 232% increase in recall of *social action events*, while only decreasing the precision by 14%.

Chapter 2

Background

In this section we first describe the state-of-the-art event detecting systems, which show that event detection is not a trivial task in general. This is followed by a description of studies that categorized Twitter data, often using unsupervised topic modeling. We then explain why we opted to use supervised learning by outlining the results from Ritter et al. (2012), who obtained very promising results by applying a supervised Maximum Entropy algorithm. Subsequently, we describe the most succesfull approaches in detecting *social action events*, especially the already deployed EMBERS system of Ramakrishnan et al. (2014). We conclude with a detailed explanation of the general event detection system of Kunneman and van den Bosch (2015), who's event set we will be using in our approach.

2.1 Event detection

A necessary prerequisite for event categorization is event detection in general, often linked to what is called first story detection. Early approaches used traditional news media documents (e.g. Yang et al., 1998; Allan et al., 2000), but after the sudden rise of social media the attention shifted to Twitter (Petrović et al., 2010;Weng and Lee, 2011). Event detection on Twitter is not a trivial task. For example, the entropy-based approach of Petrović et al. (2010) only obtained a precision of 0.34 for recognizing events. The method of Weng and Lee (2011) received a precision of 0.76, but they achieved this by only predicting a very small number of actual events (21 events over a one month period), meaning that their estimated recall is low. In comparison, Ritter et al. (2012) detected more than 1000 events over a one month period and obtained a precision of 0.52 on the 1000 events with the best eventscores. Using their method, Kunneman and van den Bosch (2015) automatically identified 93,901 events over a 6-year period. They obtained a precision of 0.63 on the 250 events with the highest eventscore. This suggests that the precision on all events will be lower, since the eventscore is a linked to larger, clearer and more popular events.

Since we will use the event set from Kunneman and van den Bosch (2015), this means that at least 37% of our events will actually be non-events. Those events are not useful for us, since we cannot assign an event-category to something that is not an event. A non-event is usually detected due to people that use the same keyword in a tweet, while not referring to the same event (or even to an event at all). Another reason is that two separate events sometimes mistakenly get combined in a single event, resulting in the fact that the full set of tweets does not refer to an individual event. An example of a non-event is shown in Table 2.1. Multiple Twitter users posted a tweet using the phrase *#23juni* to refer to different events. This resulted in an inconsistent set of tweets that is classified as a non-event.

TABLE 2.1: An example of a **non-event.**

Date	2014-06-23
Keywords	#23juni
Tweet 1	dan zie ik het maar als lekkere muziek voor op mijn verjaardag #23juni
Tweet 2	bijna jarig bitchez! :) #23juni
Tweet 3	de volgende wedstrijd wordt pittig.. #nedchi #oranje #23juni #wk2014
Tweet 4	straks kijken naar oranje #nedchi #23juni
Tweet 5	de eerste 2 uren van de werkdag zitten erop. #23juni #afstuderen #pabo4

Unfortunately, there is no way of automatically removing these non-events from the event set. This might hurt the categorization process, since we try to categorize events for which it is impossible to assign a category to. However, this is the data all event categorization systems will realistically be working with, since event detection is a difficult task in general. In our manually annotated event sets, the non-events are removed in the training phase, while these non-events are not excluded in the full event set of 93,701 events. Together, these classifications will give us a complete picture of the potential accuracy of event categorization systems.

There are a number of approaches that already tried to automatically categorize Twitter data. They are described in the next section.

2.2 Categorizing events

While event detection is a widely studied topic, categorizing these events received less attention over the years. There is, however, a large body of research on topic modeling. A topic is defined as a subject discussed in one or more documents. A topic model (first

described by Papadimitriou et al., 1998) is a statistical approach that tries to discover the different topics that are discussed in a large number of documents. This is usually an unsupervised approach which returns a set of best fitting words for each topic, which must then be manually labeled to obtain meaningful categories. Although the studies performing topic modeling seem similar to ours, the majority of these studies used structured sets of large documents instead of disorganized and scattered Twitter data (e.g. Wallach, 2006; AlSumait et al., 2008), which makes the task considerably easier.

However, there are a number of studies which did use Twitter data when performing topic modeling. Hong and Davison (2010) attempted to classify Twitter users and their corresponding messages into different categories. However, they used a predefined set of Twitter accounts that were already labeled by category instead of selecting tweets from the stream of incoming tweets. It stands to reason that using this clearly categorized set of tweets will considerably simplify the task at hand. Also, they did not identify specific events before performing the categorization.

Zhao et al. (2011) performed topic modeling on Twitter data, comparing the results to topic modeling on a traditional news medium. They also did not detect events first, but did show a number of interesting differences between Twitter and news outlets, the most important difference being that Twitter is a good source of entity-oriented topics that do not have much coverage in the traditional media (e.g. topics about lesser known celebrities). This means that these entity-oriented topics might be an interesting category for automatic detection.

Ramage et al. (2010) performed labeled topic modeling using the hashtags in tweets automatically as labels, resulting in a set of 200 different topic models. They found that these topics could be divided in five different categories: *substance, social, status, style* and *other*. It is interesting to note that the *substance* category was defined as tweets related to events and ideas. This means that they essentially did the opposite of what we try to do: instead of first detecting events and then categorizing them, they first categorized all tweets and subsequently identified tweets relating to events. The main difference is that they did not detect specific events, but only tweets that related to some unspecified event.

The main difference between our approach and the previously described approaches is that there was no event detection procedure before dividing the data into different categories in those studies. The described approaches either simply identified which topics are often talked about on Twitter or focused on the categorization of users instead of events. This means that our system of categorizing specific events cannot be compared

to them. To our knowledge, the only approach that actually attempted to categorize automatically detected events is by Ritter et al. (2012). They used unsupervised topic modeling on a set of 65 million events to identify 100 different categories. Manual annotation resulted in a set of 52 meaningful categories, which dropped to 37 after combining the ones that were very similar. 46.5% of the events ultimately belonged to one of these categories, while 53.5% of the events were placed in the *other* category.

To evaluate the performance, Ritter et al. (2012) selected the best 500 events with highest association for manual annotation. Due to this event set that is now labeled by category, they were also able to train a supervised Maximum Entropy algorithm on those 500 events. They did this to show the effect of using more training data, since the unsupervised approach uses all available events in the training phase, while the supervised approach has to rely on only the set of 500 annotated events. The unsupervised approach obtained an F1-score of 0.67, outperforming the supervised approach which obtained a final F1-score of 0.59. However, they do show that the F1-score of the supervised approach steadily increases when using more training instances. This suggests that this supervised approach will ultimately outperform the unsupervised topic modeling approach if the set of annotated training instances is large enough. This encouraged us to employ a supervised approach for categorizing events, which is explained in detail in the next section. We will compare our results to the results of Ritter et al. (2012) in the Discussion section.

2.2.1 Unsupervised or supervised learning?

There are two main options for categorizing the events: using a supervised or an unsupervised approach. Unsupervised approaches are often used in topic modeling, which consists of dividing the dataset in a predefined number of topics which must be manually labeled to construct meaningful categories. Each event is then classified as its best fitting category. The main advantage of this approach is that all events are taken into account when creating the different categories. The main disadvantage is that the number of final topics needs to be set in advance and that there is no way of knowing which categories are eventually produced.

Supervised approaches are the most common in machine learning tasks. They employ algorithms that learn from a small set of labeled instances, after which they use this learned information to classify the instances that were not present in the learning phase. The main advantages of this approach are that it is able to learn from correctly labeled data and that we are able to define the specific categories in advance. The main disadvantage for us is that only a small sample of the data can be included in the learning phase, since this type of data is not available in large quantities.

The similar approach of Ritter et al. (2012) used unsupervised learning, stating that supervised learning is problematic for a number of reasons. For one, they assert that the appropriate categories are unclear in advance. We believe that this is not an issue, mainly because we can benefit from a number of papers which describe what people talk about on Twitter (e.g. Zhao et al., 2011; Ramage et al., 2010). Also, a manual inspection of the events will reveal a great deal about what types of events are usually detected. The second argument of Ritter et al. (2012) is that manual annotation will require a large amount of effort. While this is true, we believe that a better final classification will justify these efforts. Thirdly, Ritter et al. (2012) state that they believe that the set of important categories and entities will shift over time. While this might be true for entities, it is very hard to imagine this happening for categories, especially when using a broad range of general categories. Finally, they express that many important categories are relatively infrequent, resulting in the fact that even a large dataset of annotated instances will only contain a few instances of those categories. This is indeed the case for their approach, since they initially divided the data into 100 different categories. However, our more general approach will only use a set of ten categories, making it a lot less likely that we will end up with categories that only have a few annotated examples. Also, since we determine the categories based on the actual data, it is unlikely that we will pick categories that occur very infrequently.

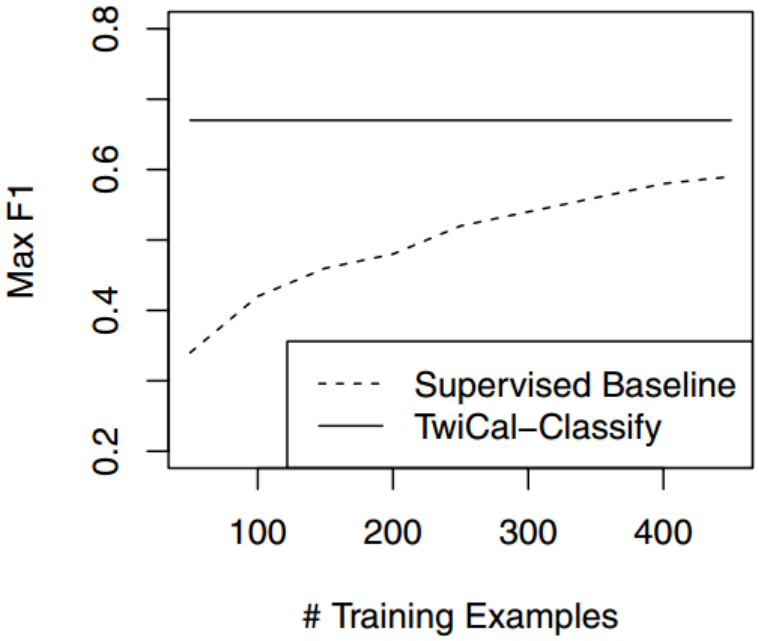

FIGURE 2.1: The comparison between the unsupervised TwiCal approach and the supervised Maximum Entropy approach by number of training examples taken into account, from Ritter et al. (2012).

Although they used unsupervised learning, Ritter et al. (2012) also presented a very compelling argument in favour of supervised learning, namely their actual results. To evaluate the results of the unsupervised approach, they compared the results to a supervised Maximum Entropy classifier that was only trained on a set of 500 annotated events. This as opposed to the unsupervised approach, which had access to the full set of 65 million events. Despite this enormous difference, the unsupervised approach outperformed the supervised approach by only 14%. Moreover, the results of the Maximum Entropy classifier revealed a trend that suggested that the supervised approach would ultimately outperform the unsupervised approach when the annotated set of examples is large enough. Their results are shown in Figure 2.1. For this reason and the arguments described in the previous paragraph, we ultimately opted to use a supervised learning approach to categorize the events. This means that we have to manually annotate a gold standard set of events. We did so by annotating 600 events with the highest eventscore as well as annotating 600 randomly selected events. This process is described in Section 3.3.

Ritter et al. (2012) and the other previously described categorization studies identify a large number of different categories, but none of the categories relate to *social actions*. However, there are a number of studies that tried to automatically detect these *social actions*, without aiming to categorize all different events that were detected. They will be described in the next section.

2.3 Predicting social action events

The first approach to automatically detecting *social action events* was by Compton et al. (2013). They tried to predict civil unrest in South-America by simply selecting tweets that contained a specific civil unrest related keyword and a mention of a date. Also, the tweet must specifically mention one of the predefined *locations of interest*. Their system obtained a final precision of 0.55 on a set of 283 events. Their approach can be criticized for a number of reasons. For one, due to a predefined keyword list, the system is not able to detect *social action events* that use newly emerging keywords for a specific event. They have the same issue regarding location, since the detection system only accepts locations that are on a manually compiled list of locations of interest. This means that the system is unable to find *social action events* that take place on a novel location. Their system has thus no predictive ability, which is a big problem. If we want to prevent *social action events* from happening, we should forecast them in advance, not detect them after the event already took place. Also, they only include event dates which fit the specific date format of e.g. *20-12* for December 20th. There is no reasoning involving terms such as *tomorrow* or *next Wednesday*, which is a more common way to

refer to upcoming dates. Due to these reasons, it is very probable that the recall of their system will be very low, since a lot of *social actions* seem to go undetected.

Kallus (2014) predicted the specific occurrence, timeframe and location of large protests and demonstrations by collecting data from more than 300,000 websources, ranging over 7 different languages. He obtained a set of events that were already labeled as *protest* and implemented a threshold to differentiate between *significant protests* and *ordinary protests*. He then trained a classifier to automatically judge whether an upcoming protest will be significant or ordinary, reaching a balanced accuracy of 72.4%. This is different from our approach, since they were able to dismiss all non-protest events in advance. His approach might be very useful for troubled countries such as Egypt, but has little practical value for relatively calm countries such as The Netherlands. Our aim is to predict all *social action events*, not differentiate between significant and ordinary *social actions*.

The most notable approach on predicting *social action events* is probably the EMBERS system by Ramakrishnan et al. (2014). Their system tries to forecast civil unrest by using a number of open source data sources such as Facebook, Twitter, blogs, news media and even economic indicators. It is a deployed system that uses multiple models, which will issue a warning alert when it believes a *social action event* is going to take place. This is similar to our goal of automatically notifying the authorities when we believe a *social action event* is going to take place. Their system successfully predicted the June 2013 protests in Brazil and the February 2014 violent protests in Venezuela. Tested over a month, EMBERS obtained a precision and recall of respectively 0.69 and 0.82.

However, it must be noted that their approach is different to ours. For example, they were able to use information from Facebook events and news articles to determine exactly when **planned** events were going to take place. Also, they use information such as currency exchange rates and counts of requests to the TOR browser.[1] Since our goal is to predict *social actions* by only using Twitter data, we are not able to exploit these types of information. However, Muthiah et al. (2015) reported the results of EMBERS for just the planned forecast model. This included the results of the classifier when only taking Twitter information into account, which is very similar to our approach. They ultimately obtained a precision and recall of respectively 0.97 and 0.15 when only using Twitter data.

[1] An indication of the number of people who chose to hide their identity and location from the online community.

It must be noted that an actual *social action* predicting system in Dutch should not be limited to Twitter data. A deployed system should be able to exploit a number of other data sources as well, similar to the sources used in the EMBERS system. Our research should be viewed as a proof of concept, showing the impact that only using Twitter data already has on predicting *social action events*. However, it is interesting to see how the results could improve if we were to exploit additional data sources. Therefore, we will mainly compare our results on predicting *social actions* to the results of the EMBERS system in the Discussion section.

2.4 Approach of Kunneman and van den Bosch (2015)

To perform the automatic event categorization, we will use the extracted event-set from Kunneman and van den Bosch (2015), henceforth referred to as K & B. They automatically extracted events based on **Dutch tweets**, therefore mainly identifying **Dutch events**. This as oppposed to the existing approaches, that all used either English or Spanish tweets. We will focus on identifying Dutch events for two reasons. For one, we can be fairly certain that Dutch tweets will refer to events that take place in the Netherlands, which is a small geographical area compared to events in other languages. For example, tweets in English can range from the whole United States to Australia, while events based on Spanish tweets range over virtually all countries in South-America. This makes it more suitable for a general *social action* detecting system. Secondly, the sheer volume of tweets in Dutch will be lower than for English of Spanish tweets. For this reason, we are more likely to detect smaller, less popular events as well. Therefore, we are not only able to detect organized protests that take place in large city, but also small demonstrations in a local municipality. Local authorities are likely to be especially interested in the latter type of events, since they will not be as prevalent in mainstream media.

K & B based their general event detection approach on the method of Ritter et al. (2012), who used explicit future time expressions to identify events. In this section, we will explain their event extraction process and also concisely describe how they obtained the different attributes for each event (shown in Table 2.2).

K & B start the extraction process by scanning each incoming Dutch tweet for a future time expression. Only tweets that contain such an expression are retained. They are able to identify mentions of specific dates (e.g. *on October 28th*), but are also able to reason over phrase combinations such as *next Wednesday* or *in a week from now*. The remaining set of tweets is then scanned for meaningful words, henceforth called keywords. A phrase can only be a keyword if it has its own Wikipedia-page and is used as an anchor text (hyperlink) on other Wikipedia pages at least once. Aside from this

TABLE 2.2: The set of attributes for each event.

Attribute	**Explanation**
Date of event	Specific date when the event took place
Keywords	Set of most descriptive terms for the event
Keywordscores	Log-likelihood scores of the keywords
Eventscore	Greatest keywordscore of the event
Tweets	Set of tweets, including text, date and username

selection process, all words preceded by a hashtag are selected as well. This is done since hashtags are often employed as event markers and are also often made-up terms for a specific event that do not have their own Wikipedia page (e.g. *#huistekoop (#house-forsale), #deringmoetnaarfm (#theringmustgotofm), #vrouw2011 (#woman2011)*).

This results in a large set of dates with corresponding keywords, which is the base of the final event-set. However, such a date-keyword pair is not necessarily a significant event. The first step K & B take to select the best date-keyword pairs is to set a threshold of a minimum of 5 tweets in which the keyword must co-occur with the date. They then measure the goodness of fit between the date and the keyword by calculating the G_2 log likelihood ratio. This is a statistic that estimates whether the observations stem from a random sampling of a distribution, using the observed and expected frequencies. This means that keywords that occur evenly spread out through the event set will attain a low score for a certain date, while keywords that virtually only occur for a specific date will attain a high score for that date. K & B use this as a confidence measure that estimates whether the keyword-date pair actually referred to an event. They then prioritize pairs that are found in tweets by many different users, scaling the log likelihood score by the percentage of unique users that tweeted about the event. The resulting score is henceforth referred to as the keywordscore. Note that these scores are event-specific, meaning that keywordscores can differ for the same keyword when present in a different event.

This set of events or date-keyword pairs often contains duplicate events, since events are often described by multiple keywords (e.g. *#widm, wie is de mol (who is the mole), moltalk (moletalk)*). To solve this problem, K & B clustered the events on a certain date based on their *tf * idf* weighing[2], ultimately merging the events that exceeded their similarity threshold. A merge simply combined the keywords and the tweets in a new event, removing the doubles. Note that this can also result in a large number of date-keyword

[2]Explained in Section 3.4.

pairs being combined together, with 43 keywords as a maximum for a single event in the final event set. The greatest keywordscore is then used as the event score, reasoning that an event is as strong as its best predictor. In practice, this means that the large, popular and nation-wide events obtain high eventscores, while the smaller, local events receive low eventscores. For example, a soccer match of the Dutch national team will receive a high eventscore, while a local gymnastics event is likely to obtain a low eventscore.

They performed this procedure for all Dutch tweets in their database, all posted between 16-12-2010 and 16-02-2015, resulting in a set of 94,526 events. After some additional filtering and merging of double events, this resulted in a final event set of 93,901 events. This is the event set that we obtained and will be working with. The statistics that are described next are all derived from this event set and are not specifically mentioned in the K & B paper.

The events mainly take place in the aforementioned window of 16-12-2010 to 16-02-2015. However, 1868 of those events (1.99%) had an event date after 16-02-2015 and thus were predicted to take place in the future at the time of extracting. 69 of those events were predicted to take place after 2016, with the most distant event taking place on October 4th, 2038.[3]

The full event set contained 210,704 keywords, with an average of 2.24 keywords per event and a median of 2. Accounting for hashtags (i.e. *#workshop* and *workshop* are considered identical) provides us with 33,807 different types of keywords, with 17,212 (50.9 %) of those keywords being unique keywords that only occurred for a single event in the whole event set. These unique keywords are spread out over 15,871 events, meaning that 16.9% of all events contain at least one keyword that is exclusively used to describe that event. Both these percentages are rather high for a general approach, suggesting that K & B succeeded in extracting the meaningful terms for the different events.

Their final event set is used for our categorization system. In the Method section, we explain which event sets are annotated as a gold standard and how we use this gold standard to evaluate the performance of our general categorization system.

[3]A joke by a single user that counted down the days to World Animal Day in 2038, starting from 10000 days and doing this for a week, thus being picked up by the event detection system.

Chapter 3

Method

In this section, we first describe which subsets of events are selected for manual annotation. This is followed by an explanation of how we collected a set of extra tweets for each event. We then define which categories we will use for the classification task, after which we carefully describe the annotation procedure and the obtained results of that procedure. We then give an overview of all different features that are extracted and present the most informative features. Following that, we define the training and testing procedure when using those features in a classification task and outline how we plan to evaluate the performance of the classifiers. Finally, we describe three different methods that we will use to try and increase the final accuracy and F1-score.

3.1 Three event sets

The categorization process is started by creating three different event sets, selecting the events for each set from the full event set of 93,901 events from Kunneman and van den Bosch (2015). Our first event set contains the 600 events with the highest eventscore. We selected these events for two reasons. For one, we want to investigate whether it is easier to categorize the clearest and most popular events, since we might later use this information to discard unclear events in advance. Secondly, this enables us to have a detailed comparison to Ritter et al. (2012), since they evaluated their system by looking at the classification of the best 500 events. We refer to the set of events with the highest eventscores as the **best event set**.

Our second event set is the representation of the full event set, so we are able to evaluate how we perform on categorizing all events. Therefore, we semi-randomly select 600 events from the full event set. We create this set by randomly selecting an event from the ranked total event set for every interval of 155 events, excluding the best 600 events.[1] We refer to this event set as the **random event set**.

[1] We randomly select one event in the range between the 601st ranked event and the 756th ranked event, one between 756 and 911, one between 911 and 1,066, etc.

The third event set is selected to measure the effect of using more labeled training instances. Therefore, we simply combine the best and random event set and refer to that event set as the **combined event set**.

3.2 Collecting extra tweets

A problem with the set of 93,901 events from Kunneman and van den Bosch (2015) is that it only contains tweets that contain future time expressions. This means that, even though their system was able to reason over phrases such as *morgen (tomorrow)* and *over 2 dagen (in 2 days)*, it never identified tweets that did not use such a time expression. Also, this means that it never identified tweets that were posted **after** the event took place. These are often the tweets that express an opinion about the event instead of just mentioning when it will take place. This different type of tweets will make a valuable addition to the existing set if we are able to reliably detect them. Note that collecting tweets after the event took place is evidently not useful for **predicting** when future events will take place. However, since we are creating a training set for the classifier, we want to extract as much information about the events as possible. For this reason, we also search for relevant tweets that were posted after the event took place.

What we do have for each event, is a set of words that are the most indicative of the event (keywords) and a specific date when the event took place. It stands to reason that tweets that were posted in a small window around that date and contain one or more of the keywords are also related to the event. For each of the events in the best and random event sets, we collected a set of tweets that were posted seven days before or after the event date that contained at least one of the keywords. We did this by using TwiNL[2] (Sang and van den Bosch, 2013), a database of Dutch tweets. This database contains about 40% of all Dutch tweets in the period December 16th, 2010 to today, resulting in a set of approximately 3 billion tweets. We refer to the set of tweets obtained by Kunneman and van den Bosch (2015) as the **old tweets**, while we will refer to the newly identified tweets as **extra tweets**.

3.2.1 Filtering process

Initial runs of this process resulted in an enormous set of extra tweets, mainly due to the influence of some general keywords that were not indicative of a specific event at all. After careful inspection, we found out that especially names of Dutch cities resulted in a huge number of extra tweets, while barely being relevant to the event. We therefore

[2] www.twiqs.nl

decided to filter all tweets that only contained a city as one of the keywords. Tweets that did contain a city name, but another keyword as well, were kept in. Since the aim is to have a large variety of tweets and not just duplicates of tweets, retweets are filtered out as well. They do not contribute anything new to the information we already have about the event and only make it more difficult to evaluate whether the set of extra tweets relates to the event. Ultimately, we ended up with a set of extra tweets for each event. The specifics are shown in Table 3.1.

TABLE 3.1: Data about the number of events, old tweets and extra tweets.

	Best events	**Random events**
Number of events	585	586
Number of old tweets	8,560	6,288
Number of extra tweets after filtering	17,364,430	399,567
Number of extra tweets with at least 2 keywords	433,345	68,410

We see that the best 600 events returned significantly more tweets than the randomly selected events. This makes sense, since the high-ranked events are usually events that have a lot of social impact (e.g. Christmas, elections, important soccer matches) and will be tweeted about often. There are still more than 400,000 tweets left if the tweets need to contain at least two keywords, about six times as much as the random events. However, it needs to be noted that the division of those extra tweets is skewed. The best five events already account for 23% of the total extra tweets and the best 25 events account for 53% of the extra tweets. Also, the average number of tweets per event is 1,261, while the median number of tweets is only 302.

We now face the problem of having to select the best tweets in the sets of extra tweets and filtering out the tweets that are not so relevant after all. To do this, we created a ranking that orders the extra tweets based on their relevance by using four different features:

(1) Number of keywords present in tweet

(2) Accumulated keywordscores

(3) Absolute number of days to event

(4) Binary value whether the tweet was posted before or after the event

The first feature is evident: the more keywords a tweet contains, the more relevant the tweet generally is. The same reasoning applies to keyword scores, a higher score means that the keyword is a better indicator for specific events, with keywords that are cities generally scoring very low and keywords that only occur for a single event generally scoring very high. For the third feature, it stands to reason that a tweet that is posted very close to an event is more probable to be relevant than a tweet that is posted a couple of days away. The fourth feature is based on the fact that we already have a set of tweets that were posted before the event took place. All else being equal, we rather include a tweet that is posted after the event, to obtain a more representative sample of the event. There is no weighing of the features involved. The ranking is first performed by just taking feature 1 into account. Events with the same score are then ranked based on feature 2, and so on. This means that a tweet that contains three keywords will always be higher on the ranking than tweets that contain only two keywords, no matter what the other featurevalues are. This ranking is then used to display the best tweets in the annotation process, which we will describe in the next section.

3.3 Annotation of gold standard

To create a gold standard set of categorized events, we selected the best and random event set for manual annotation. First, we performed a preprocessing step by removing all events that turned out to be non-Dutch. The TwiNL system detects Dutch tweets automatically, but is not 100% precise. This means that on occasion, English, German or even Norwegian tweets slip through. These tweets then sometimes form an event in the event set and are subsequently removed by us. We removed 15 of the best events and 14 of the random events, resulting in sets of 585 and 586 events that remained for annotation.

Similar to Ritter et al. (2012), we identify three different steps in our annotation process. For each event, the annotators have to answer the following three questions:

- Is this an actual event according to the definition?
- Is the group of extra tweets relevant to the event?
- What is the category of this event?

These steps are described in detail below.

We have to keep in mind that the events themselves were also obtained automatically. This means that it is not necessarily the case that an event in the full event set is actually a clear event in the eyes of a human annotator. In our event set of 93,901 events, it

was estimated that approximately 37% of the identified events were actually non-events. Therefore, the first step in the annotation process is that annotators have to determine whether the presented event is actually an event. This is more difficult than it seems, since there is no clear consensus in the literature on the exact definition of *event*. Also, most studies avoid providing an exact definition (e.g. Ritter et al., 2012, Zhao et al., 2011). We will explain how we arrived at our definition of an event below.

Aggarwal and Subbian (2012) define a *news event* as something that happens at a specific time and place that is of interest to news media. The definition of McMinn et al. (2013) is very similar, although in their case events do not necessarily need to be of interest to news media, but they need to be *significant*. Weng and Lee (2011) essentially turned the definition around. Instead of finding events on Twitter based on a definition, they simply defined an event as a burst in the usage of a related group of terms on Twitter. The definition of Becker et al. (2011) is similar, since they define an event as a real-world occurrence with an associated group of tweets discussing the event. Since our goal is to categorize the events, we do not want to include all events simply based on a burst in usage of certain terms, but only want to retain clear events that have a potential category. This is why we opt for a sligtly more specific version of the definition of McMinn et al. (2013):

- Something is an event if it happens at a specific place, at a specific time and if it is important for a large number of people.

This definition still leaves some room for discussion. For example, the celebration of Christmas is a clear event, but not everybody celebrates it at the same place. However, people do celebrate it with a large number of people at a specific place (just not the same as everybody) and specific time. Therefore, we extended the definition of event and instructed the annotators to still judge these events as actual events. To determine whether it is an actual event, the annotators first examine the keywords and the date of the event. After that, a random selection of seven of the old tweets (tweets detected by their future time expressions) is presented to the annotator. There are usually more old tweets available, but since ultimately 1,200 events need to be annotated, we do not want annotators spending too much time reading about a single event. Based on this information, they have to determine whether it is an actual event or not.

The seven shown tweets are selected randomly from the set of tweets of each event, but we do exclude tweets that are too similar to tweets already in the set. This overlap between *tweet x* and *tweet y* is calculated by taking the number of unique words that are both in *tweet x* and *tweet y* divided by the maximum length of *tweet x and tweet y*.

This means that we use the following equation:

$$overlap = \frac{set(x)\ \&\ set(y)}{max\_len(x, y)}$$

This overlap is measured for every tweet that is shown up to that point. This means that the first randomly selected tweet will always be added, the second will be evaluated against the first, the third against the first two, etc. Tweets that exceed the empirically set overlap threshold of 0.8 are excluded.[3] This process is continued until we arrive at a diverse set of seven tweets. If we run out of tweets before we reach the total amount of seven tweets, we randomly select tweets from the excluded tweets set until we do reach seven tweets.

If the annotator annotates the event as not an actual event, they immediately move on to the next event for annotation. We are not interested in the category of something that is not an event. Otherwise, the annotation process for the event is continued. In the next step the annotators are presented with the seven best extra tweets according to the ranking described earlier. They have to judge whether these tweets are relevant for the event or not. This is done for the whole group of tweets, not for every individual tweet. This is because later on, we want to possibly include a large number of the extra tweets in the tweet set. If we annotate on the level of individual tweets we are not able to make more general judgements about the whole set.

The rules for this judgement are very strict. We have to keep in mind that we selected the best seven tweets of the set. This means that if one of those tweets already is not relevant for the event, we can conclude that it is very probable that the other (not shown) tweets are not relevant as well. Even if there is only one tweet that is not relevant in the group of seven, annotators have to judge the whole group as not relevant. However, sometimes annotators are simply unsure whether a tweet is relevant to the event or not. We allow one tweet to be doubtful, if the other six tweets are indeed relevant. If there is doubt about more than one tweet, the set of extra tweets should be classified as irrelevant. The full rule system is lined out in table A.3 in the Appendix.

For all events with relevant extra tweets, we add the seven annotated tweets to our dataset. However, recall that those tweets were ranked based on their initial relevance. If for example the 7th and 8th ranked tweet obtained the same score, it was randomly decided which tweet was included in the set of seven best tweets. Since we now learned

[3] A few examples of the overlap between tweets are shown in Table A.2 in the Appendix.

that at least one tweet with that score was relevant, it is reasonable to assume that all tweets with that same score are relevant as well. This means that all tweets that obtained at least the same score as the 7th ranked tweet are included in the final set of tweets for the event. The specifics are shown in Table 3.2. For both the best and random event set, the adding of the extra tweets resulted in a considerable increase in total tweets. This extra information will be valuable in the classification process.

TABLE 3.2: Number of extra tweets that are added in comparison with the number of old tweets.

	Best events	**Random events**	**Combined events**
Old tweets	8560	6288	14848
Added extra tweets	14259	19282	33541
Increase	167%	301%	226%

3.3.1 Selection of the categories

The most important and final step in the annotation process is the actual categorization of the events. First, we had to define the categories annotators are able to choose from. We do not want to end up with categories that are too general, but also certainly not with a set of too specific categories. We want the classifier to be able to reliably distinguish between the different categories, but the final category should still contain valuable information about the event. There are a few approaches that are similar to ours in that they tried to categorize information obtained from Twitter. They will be described below.

Ritter et al. (2012) applied an unsupervised learning approach to automatically divide their event set in 100 different categories. These categories were then manually inspected to identify which categories referred to coherent event types. They ultimately identified 37 different categories and manually annotated the topic of these categories. This included some specific categories, such as *VideoGameRelease*, but also general categories such as *Sports*. These types are clear, but they are not suitable for our purposes, since there is a lot of overlap between the categories. For example, there is a category for *sports*, but also for *wrestling* and *fitness*, and they also identified the similar categories *music*, *performance* and *concert*. We cannot expect that annotators are able to distinguish between those types and we definitely cannot expect that a classifier will be able to differentiate between those types either.

Another similar approach was used by Zhao et al. (2011), who applied unsupervised topic modelling to compare Twitter topics to topics in the New York Times newspaper. Due to this comparison, they just used the categories that were already defined for every article by the New York Times. Although this resulted in very clear categories with barely any overlap, they were not able to exploit any Twitter-specific characteristics in the creation of the categories. Since we do not make any comparison with news articles, we do want to exploit these properties. For example, Zhao et al. (2011) used *finance* as a category, but we know that regular people do not tweet often about finance-related events. Also, they did not have a category for musical performances, while we know that this is probably one of the most important topics on Twitter.

Ramage et al. (2010) also performed unsupervised topic modelling to divide their set of tweets into 200 different categories. They subsequently divided these categories into five general categories, *social*, *substance*, *style*, *status* and *other*. Those five categories are too general for our approach, because our aim is to categorize specific **events**, not individual tweets.

None of the previous studies provided us with a list of categories that we could duplicate for our purposes. However, they do provide us with a clear overview of Twitter topics. We manually inspected about 200 of our events and use this information to create a set of categories that we judge as the most suitable for our purposes. They are shown in Table 3.3.

TABLE 3.3: The different categories with examples.

Category	**Example events**
Sport	soccer match, local gymnastics event
Politics	(local) election, public debate
Broadcast	Television show, premiere of a movie
Public event	Performance of a band, festival
Software	Release of game, release of new iPhone
Special day	Mother's Day, Christmas
Social action	Strikes, demonstrations, flashmobs
Celebrity news	Wedding or divorce of a celebrity
Advertisement	Special offers, retweet and win actions
Other	Everything that does not fit in one of the other categories

Most categories are very straightforward, such as *sport* and *politics*. Events related to television and movies are combined in *broadcast*, while the events related to music, concerts and performances are combined in the general category *public event*. *Software* includes both releases of new smartphones as well as releases of games or operating systems. The category *special day* is different from what we observed in other papers. There are a couple of reasons why this is included as a separate category. For one, there are probably more special days than people realize. Aside from the obvious ones such as Mother's day or Christmas, there are even specials days for e.g. your pet or your secretary. Secondly, since these events have a very clear date and are often accompanied by specific and unique hashtags, they are easily picked up by the event detection system. Thirdly, the most popular special days such as Christmas and Queen's day often result in multiple events in the event set. For example, the number of events for which one of the keywords contains the word *kerst (Christmas)* is 1,581 in the whole event set. They do not necessarily all belong to the category *special day*, but it at least shows that it deserves its own category.

The category *social action* is necessary as well, since we are interested in automatically predicting social uproar. We include all events where people come together in large groups to achieve a (higher) goal. It is often a good indicator of a social event if this happens unexpectedly or on short term notice, but this is not a necessity. This way, we include very clear social actions such as demonstrations and protests, but also include harmless social events such as flashmobs. We do this since it might still be necessary to notify the authorities if a large number of people suddenly come together in a specific place, even though they initially do this with good intentions.

We first believed that *celebrity news* deserved its own category as well, since Zhao et al. (2011) found that Twitter users often tweet about celebrities and their lives. However, after the annotation process we observed that this category was only chosen a handful of times. We therefore simply removed this category and merged the instances that were classified as *celebrity news* with the category *other*. The final category is the category *advertisement*. There are often people promoting special offers on Twitter that start or end on a certain date, so they are picked up by the event detecting system. This does not really belong in one of the other categories, so we decided to assign it its own category. However, it is disputable that these events are actually events in the first place. There is usually no clear date when the event is taking place, but more of a window in which it is possible to participate in the offer. This makes it questionable whether it fits the definition of an event. We decided to still include the category so we could observe how the annotators judged those events.

We believe that the category *social action* is most similar to the categories *politics* and *public event*. Demonstrations and protests usually arise from civilians being unhappy about decisions of the government and are thus often politically loaded. This might make it difficult for the classifier to distinguish between them. On the other hand, *social actions* such as the Project X party are very similar to *public event*. This type of *social action events* take place at a public place, with most people positively looking forward to it. This is also the general theme in *public events*.

3.3.2 Annotation results

First, we annotated the set of the 585 best events. 195 of the events had a double annotation so we could measure the inter-annotator agreement. The other 390 events events were annotated once and are only usable if there are barely any discrepancies between annotators. Seven different annotators were involved in the process, who all at least annotated 40 events and at most 175 events. These are the conclusions about the 195 double annotated events:

- Annotators disagreed whether it is was an event for 14 of the 195 events (7%)
- Annotators agreed on a positive event for 138 of the 195 events (71%)
- Annotators agreed on a negative event for 43 of the 195 events (22%)

We are only interested in events that are annotated as actual events, so events that are negative or where the annotators disagreed are discarded. These are the conclusions on the 138 events that were annotated twice as an actual event:

- Annotators disagreed on the relevance of the extra tweets for 5 of the 138 events (3.6%)
- Annotators disagreed on the category for 12 of the 138 events (8.7%)

We are able to assess the inter-annotator reliability by calculating Krippendorff's alpha (Hayes and Krippendorff, 2007). This is a conservative measure that corrects for chance agreement among annotators and is considered to be an appropriate measure for tasks that involve more than two annotators. We obtained a Krippendorff's Alpha score of 0.812 for judging whether or not it was an actual event, a score of 0.92 for judging whether or not the extra tweets were relevant and a score of 0.895 for the agreement of categorizing the events. These scores are considered excellent and show that we can reliably view the events that were annotated once as if they were annotated correctly.

We are also able to calculate the mutual F1-score for each of the three steps. For the first two steps we only have a positive and a negative class. For the agreement on whether

or not it was an actual event, we obtained a mutual F1-score of **0.93**, while we obtained a mutual F1-score of **0.96** for the relevancy of extra tweets. However, for the categories we are able to calculate the mutual F1-score per class. This gives us a nice overview about which categories are the most difficult to assign. The results are shown in Table 3.4.

TABLE 3.4: The mutual F1-score for the double annotated events. The number of instances is the number of events where at least one of the annotators assigned the given category.

Category	**F1-score**	**Instances**
Sport	0.89	5
Politics	0.97	16
Broadcast	1.00	5
Public event	0.96	44
Software	0.94	19
Special day	0.85	31
Social action	0.94	9
Advertisement	0.73	7
Celebrity news	NA	0
Other	0.78	14

In general, we see very high agreement among the annotators for a 10-class problem. It seems that it is harder to identify *advertisements* and *special days* than it is to identify political and public events. However, it must be noted that most of the classes have very few instances, making it difficult to draw reliable conclusions. The F1-score for the *celebrity news* category is even ill-defined, since none of the annotators annotated an event as such.

Events that were classified as not an actual event by at least one of the annotators are excluded. This is the case for 148 out of the 585 best events (25.3%), leaving a set of 437 events. We also exclude events where annotators disagreed on the category. As shown before, this only happened 12 times, resulting in a final set of 425 of the best events. For the random event set, 224 out of 585 events (38.2%) were annotated as not an actual event. This is very similar to the estimated 37% of non-events that the full event set expected to contain. It is also a considerable increase over the best events, suggesting that the best scoring events are indeed more clear events in general. The randomly selected events only had a single annotator for each event, meaning that the remaining 362 events are all added to the random event set.

3.3.3 Category results

We are now able to show the results of the annotation process in terms of the categories. The distribution of those categories is shown in Figure 3.1 (red) for the best events and Figure 3.2 (blue) for the random events.

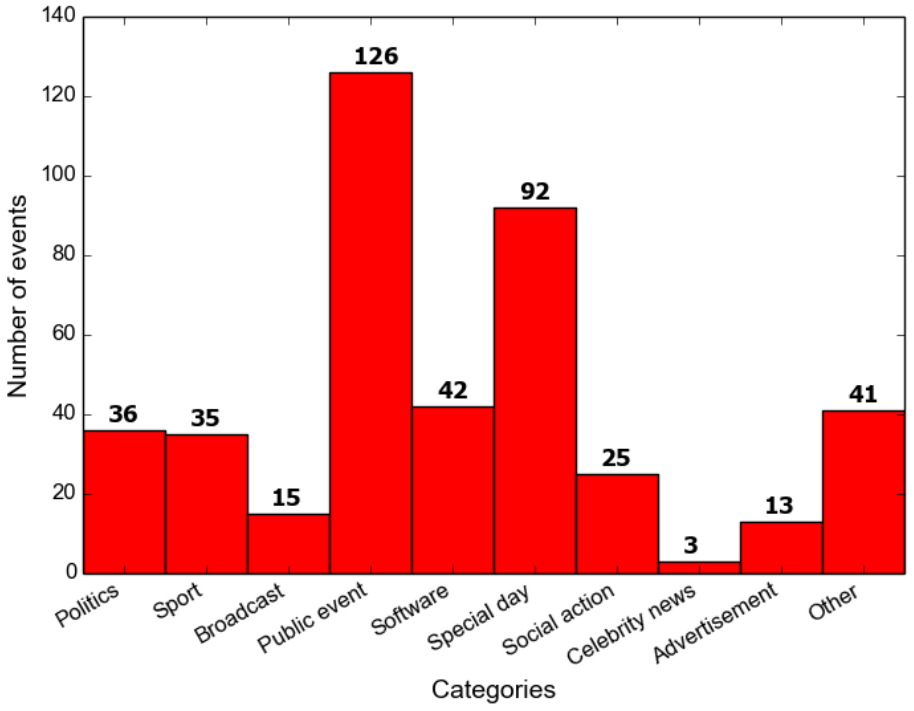

FIGURE 3.1: The distribution of categories for the 425 **best** events.

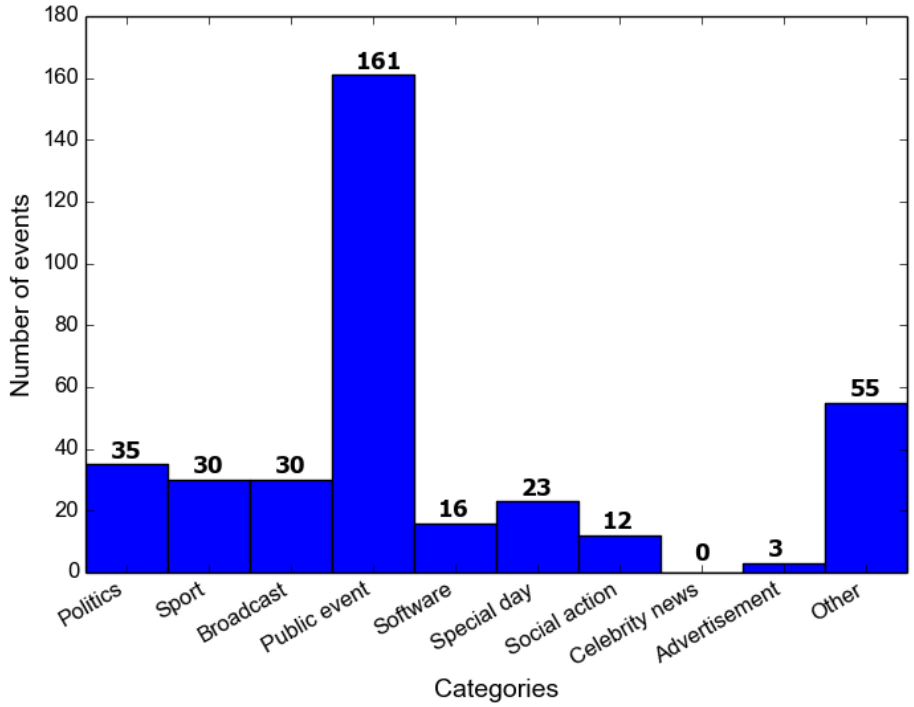

FIGURE 3.2: The distribution of categories for the 362 **random** events.

It immediately stands out that *public event* is the dominant category, with 29.6% of the best events and even 44.5% of the random events. Other events are less frequent, but still occur regularly, with the exception of *advertisement* and *celebrity news*. The latter category was so infrequent that the category was removed from both event sets. *Advertisement* was removed from the random event set, but was retained for the best and combined event set.

A major difference between the two sets of annotations is the frequency of the *special day* category. 21.6% of the best events were annotated as a *special day*, but this dropped to 6.4% for the random events. The explanation is evident: most of the special days are popular events with high eventscores (e.g. Christmas, Mother's day, Valentine's Day) and thus occur relatively often in the best event set. This is not the case in the random event set. When a special day event occurs here, it is usually a lesser known special day or a small sub-event of one of the bigger events (e.g. the celebration of Christmas in a specific school instead of the main Christmas event). For the *other* category it is the other way around, it is more frequent in the random set (15.2%) than in the best event set (9.6%). Although this difference is small, it might be due to the fact that the random events (with a lower eventscore) are simply less clear than the popular, high-scoring events in the best event set, with the result that the annotators had to choose *other* more often.

3.4 Feature extraction

The aim is that the classifier learns the specific properties of each category, so that it can use those in the classification process. Therefore, we have to extract a predefined number of features for each event. We are then able to reason over these features using different classification algorithms, resulting in an assigned category for each event.

Most of these features are very straightforward. We include the eventscore, the keywordscores, the date of the event and the number of tweets included for each event. Some features are more specific, such as the number of tweets before, after and during the event. Although these types of features might be valuable, they are probably too general to allow the classifier to easily distinguish between the different categories. The most evident way of storing information about the events it to use features for all possible words in the event set. This means that every word has its own feature, with its number of occurrences in the set of tweets as its featurevalue. This way, the algorithm is able to reason over the different words, e.g. events where the word *stemmen (vote)* occurs often are probably of political nature, while events that contain the word *winnen (win)* are probably related to sports. This process of incorporating all possible

words is called the **bag-of-words approach**. Note that we include the Twitter usernames in this process as well. It is potentially very valuable if we are able to identify Twitter accounts that for example only tweet about broadcasts or about political events.

For every category, we identified the most indicative words for that category based on their *tf *idf* score. This is a measure that reflects how important a word is for a certain document, or in our case, a certain category. The term *tf* stands for term frequency and is simply the frequency of a word for a certain category. On the other hand, *idf* is the inverse document frequency. This is calculated by taking the logarithm of the amount of events where the term appeared, divided by the total number of events. The full equation is thus as follows:

$$tf * idf_{word} = freq(word, cat_c) * \log_{10} \frac{events_{word}}{events_{total}}$$

The most indicative words per category are shown in Table 3.5. The results seem very intuitive. For *sport*, *politics*, *broadcast* and *software* the results are especially clear. They even describe specific sports teams, political parties and software corporations. The most indicative words for *social action* are very clear as well, with predictors such as *protest*, *demonstratie (demonstration)* and *manifestatie (manifestation)*. For this reason, we expect that these categories are more easily distinguishable from the other categories, resulting in a better general classification.

However, the categories *public event*, *special day* and *other* do not have distinct indicators. For *other* and *public event* this is as expected, since these are very broad categories. However, the results for *special day* are surprising. For example, there are no words relating to Christmas, Valentine's Day or Queens Day, while we know there were a lot of events regarding those special days. The only words that clearly relate to special days are *vieren* (*to celebrate*) and *dag* (*day*), while the other words are very general. The remaining category *advertisement* shows signs of over-training, since all the indicative words are only present in a few specific events. They do not seem as words that will generalize well to other *advertisement* events that are not related to one of those specific actions. The categories described in this paragraph seem to lack a general theme that is present across all their events. This means that these events are simply less distinguishable in general, resulting in the fact that they are the most difficult to categorize for the classifier.

TABLE 3.5: An ordered list of the 12 most indicative words per category according to their *tf * idf* score.

Category	**Most indicative words**
Sport	wedstrijd tegen voetbal wint rt ajax psv vanavond oranje speelt om league
Politics	stemmen verkiezingen stem cda vvd pvda d66 groenlinks stemt partij gr2014 sp
Broadcast	tv kijken vanavond aflevering zien tvtip serie seizoen nieuwe begint vanaf spannend
Public event	rt vanavond was vandaag wat wie veel zo morgen bij zijn er
Software	apple iphone microsoft uit gta wachten komt windows heeft gtav niet nieuwe
Special day	vandaag rt wat iedereen dag aan als vieren morgen dat mij toch die fijne
Social action	tegen iedereen protest respect ze demonstratie allen succes manifestatie vandaag staken haag
Advertisement	cadeau nvmopenhuis openhuizendag nvm cadeauapp openhuis huizen route 12day
Other	niet rt die geen wat wordt worden dus vandaag 1 dat heeft

This shows that the bag-of-words features provide the classifier with a lot of information. However, there is still no deeper reasoning involved concerning the words and events in question. That is why we extracted a number of features that are able to abstract over the different words and events, instead of simply having a superficial consideration of the words themselves. These features are described below.

3.4.1 Polarity and subjectivity measures

First, we performed sentiment analysis to extract the general polarity-value and subjectivity value for each event, using the Python-module Pattern (De Smedt and Daelemans, 2012b). The **polarity-value** aims to predict the polarity of a certain document, meaning that it tries to determine whether the text is positive or negative of nature. The **subjectivity-value** on the other hand tries to measure how subjective the author was

when writing the document. The measures are calculated based on an annotated lexicon of Dutch adjectives (De Smedt and Daelemans, 2012a), for which every adjective has a polarity value between -1 (negative) and 1 (positive) and a subjectivity-value between 0 (objective) and 1 (subjective). All values of the adjectives that occur in the document are then combined to obtain a general measure of polarity and subjectivity for the whole document. We simply use the combination of all tweets of the event as a single document, resulting in two general values that are added as features for each event.

This information might be valuable, since, for example, we might expect events that are annotated as *public events* (performances) to be very positive in general (people enjoy those events), while events such as *social actions* are usually negative of nature. Also, people describing sports events are expected to be very subjective, while events relating to software are expected to be more objective. This means that this information might be helpful for the classifier to distinguish between the different categories.

TABLE 3.6: The average polarity and subjectivity scores per category for both event sets.

	Best events		**Random events**	
Category	Polarity	Subjectivity	Polarity	Subjectivity
Sport	0.034	0.551	-0.059	0.552
Politics	-0.006	0.569	0.083	0.494
Broadcast	0.053	0.660	0.127	0.588
Public event	0.037	0.656	0.078	0.586
Software	0.133	0.617	0.135	0.612
Special day	**0.215**	0.606	**0.197**	0.627
Social action	0.034	0.619	0.046	0.563
Advertisement	**0.311**	0.651	**0.176**	0.633
Other	0.048	0.534	0.055	0.515

We are able to check these predictions by calculating the average polarity and subjectivity per category, which is shown in Table 3.6. We see that the average polarity is usually neutral, except for *special day* and *advertisement*. For *special day* this is evident, since special days are usually happy occasions (e.g. Mother's day, Valentine's day) and this will result in Twitter users that tweet positive messages for those events. For *advertisement* this is somewhat surprising, but it is already much clearer when we recall how polarity is calculated. This is done by only using the individual polarity-scores from the adjectives in the annotated corpus. There is no abstract analysis performed

on the actual meaning or emotions of the sentences. And one thing we know about advertisements is that they usually incorporate positive adjectives to recommend their products and actions, thus resulting in a high polarity-score. Since *advertisement* is a small category that does not seem as easily distinguishable as *sport* or *politics*, this might be a valuable feature for recognizing that category.

However, the subjectivity-score does not seem as a very informative feature. The average score is very neutral in general, meaning that no category is either very objective or very subjective. On top of that, the difference between the average subjectivity-scores is very small, with all scores ranging between 0.49 and 0.66. This means that we do not expect that this feature will be very beneficial for the classifier.

The two previously described features do not seem beneficial for predicting *social action events* specifically. The average polarity is very neutral (0.034 and 0.046) and does not stand out from the other categories. We expected *social action events* to be negatively loaded, but it turned out that they were even slightly positive in general. We described earlier that the categories *public event* and *politics* are the most similar to *social action* in general. Unfortunately, the polarity and subjectivity values of these two categories are very similar to *social action*, meaning that these values will probably not help the classifier to make this distinction.

3.4.2 Periodic events

Other more abstract features that are extracted are related to periodicity. It is often the case that events are of a periodic nature, with equal time intervals between the events, usually characterised by the same keywords. This is the case for very popular annual events such as Christmas and Valentine's Day, but also for smaller weekly events such as television programs. This might also provide the classifier with valuable clues about the category, since we expect that periodic events are for example often *special days* and are very unlikely to be *social actions*.

Luckily, Kunneman and van den Bosch (2015) already performed a periodicity analysis on our set of events, resulting in a set of 1500 events that were considered periodic with their corresponding periodicity-score. We are now able to check whether our events are periodical by matching them by their date and keywords to the obtained set of periodic events. Events are allowed to mismatch by at most three days and one keyword to still be classified as a periodic event. This is the case since events might take place on a slightly different date and still be the same event (e.g. annual events that always take place in the weekend) and also because a minor change in keywords does not often influence the event (e.g. *appelpop, festival, Tiel* is probably the same as *appelpop, festival, #zinin*).

For each event, a binary feature is added for whether the event is periodic or not. If the event is indeed periodic, the confidence score, support score and type of periodicity (e.g. annual, weekly, monthly) are also added. [4]

TABLE 3.7: The percentage of periodic events for each category.

Category	**Periodic**	**Instances**	**Percentage**
Sport	6	71	8.5%
Politics	7	65	10.8%
Broadcast	7	45	15.6%
Public event	44	287	15.3%
Software	7	58	12.1%
Special day	**53**	**115**	**46.1%**
Social action	4	37	10.8%
Advertisement	3	16	18.8%
Other	15	84	17.9%

The percentage of events that is periodic is shown for each category in Table 3.7. We see that there is not much difference between the categories, except for the category *special day*. As expected, *special days* are often periodic events, making the periodicity features very valuable for recognizing them. However, these features will probably not help in identifying *social action events*, since the percentage of periodic events for this category does not stand out from the other categories.

3.4.3 Wikipedia features

The previously described features are all potentially very valuable, but are in the end still features that abstract over the event as a whole, instead of over individual phrases. That type of information can be very valuable as well. Consider the two examples in Table 3.8.

For humans, these events are very clear. The first event is a *public event*, since people are tweeting about the singer Justin Bieber. The second event will be categorized as *sport*, since Feyenoord is one of the biggest soccer clubs in the Netherlands. For the classifier this is not as clear, since it does not possess this real world knowledge to guide its decision making. If *Justin Bieber* and *Feyenoord* did not occur in the training set, it has no other clues to base its decision on. We need a way to insert this real world knowledge in the system, to ensure that the system is able to reliably model human decision making.

[4] For a full explanation of these scores see Kunneman and van den Bosch (2015)

TABLE 3.8: Two hypothetical events.

	Dutch	**English**
Keywords	Justin Bieber, #zinin	Justin Bieber #excited
Tweets	Vanavond naar Justin Bieber #zinin	Going to Justin Bieber tonight #excited
	Eindelijk naar Justin Bieber vanavond #zinin	Finally going to Justin Bieber tonight #excited
	Vanavond met de trein naar Justin Bieber	Taking the train to Justin Bieber tonight
	Over een uur naar Justin Bieber #zinin	Going to Justin Bieber in an hour #excited
Keywords	Feyenoord, #zinin	Feyenoord, #excited
Tweets	Vanavond naar Feyenoord #zinin	Going to Feyenoord tonight #excited
	Eindelijk naar Feyenoord vanavond #zinin	Finally going to Feyenoord tonight #excited
	Vanavond met de trein naar Feyenoord	Taking the train to Feyenoord tonight
	Over een uur naar Feyenoord #zinin	Going to Feyenoord in an hour #excited

A named entity tagger might be helpful here, since this might allow us to differentiate between a person and an organization. However, this is often not specific enough, since it does not help in e.g. differentiating between political organizations and sport organizations or between singers and soccer players. It would be very beneficial for the classifier if we could somehow denote *Feyenoord* as *soccer club* and *Justin Bieber* as *singer*, because the classifier might then able to reason that *soccer clubs* often occur in events labeled as *sport*, while *singers* often occur in events labeled as *public event*.

To accomplish this, we used the database version of Wikipedia, DBpedia (Bizer et al., 2009). DBpedia is designed to structure the information available on Wikipedia, making it accessible on the World Wide Web. It is a large database that contains multiple entries for each Wikipedia-page, that can be queried to extract the specific information people are interested in. The information is structured in the form of triples based on the page, ontology-type and the specific value for the ontology-type. This database possesses an enormous amount of real world knowledge that can possibly be extracted.

Since we have to generalize over the different terms, we are especially interested in the `type` attribute of the ontology. This is often a general descriptor of the main type of the page and furthermore, essentially all DBpedia pages possess this attribute. A few examples are shown in Table 3.9. We see that Feyenoord, AFC Ajax and PSV all received the value *SoccerClub*. This means that the classifier is now able to generalize over these examples, reasoning that *SoccerClubs* often co-occur with events that are categorized as *sport*. Another advantage is that most pages have multiple values for the `type` attribute. This provides us with a broader range in comparing different terms,

Table 3.9: A few examples of the type values in DBpedia.

Page	**Type**
AFC_Ajax	SoccerClub
PSV	SoccerClub
Feyenoord	SoccerClub
Feyenoord	Organisation
Kanye_West	MusicalArtist
Justin_Bieber	MusicalArtist
Justin_Bieber	Artist
Justin_Bieber	NaturalPerson

e.g. Feyenoord is a *SoccerClub*, *SportsTeam* and *Organisation*, while Justin Bieber is an *Artist*, *MusicalArtist*, *MusicGroup* and even a *NaturalPerson*. This gives us a broader description of the terms than just a single value would have provided us with.

A problem that arises is locating the correct page for obtaining the type. It is often the case that the terms are ambiguous, resulting in a DBpedia page that serves as a disambiguation page, simply listing the possible pages for that term. For example, Ajax can refer to the soccer club Ajax, but also (among others) to the Ajax from Greek mythology, a village in Ontario, a cleaning product and even to Asynchronous JavaScript and XML (AJAX). This means that it is important to construct a system that is able to reliably choose the correct page given a list of possible pages.

This disambiguation process is type of named entity disambiguation and is usually called Wikification when using Wikipedia entities (Mihalcea and Csomai, 2007). There are a few approaches that applied sophisticated machine learning techniques to solve this problem (e.g. Bunescu and Pasca (2006); Zhang et al. (2010)), but doing this falls outside the scope of this project. We will perform the disambiguation by simply looking at what other pages refer to when using a certain term. Luckily, it is possible on Wikipedia to refer to a certain page by using the general term and redirecting the link to a specific page. For example, we are able to mention *Ajax* in the text and then link to the page *AFC_ AJAX*. This means that we are able to search Wikipedia for a specific term that it is also a reference (hyperlink) to another page. We simply rank the found pages for a phrase in order of occurrence and use the page that is referred to most often as the page we will extract the `type` attribute from. We reason that the page that is most often used as a reference is also the most probable to be the relevant page in the context. This process was performed by using the Dutch Wikipedia dump.[5]

[5]Available at https://dumps.wikimedia.org/

A manual check of the results showed that this will provide us with the correct page most of the time, for example people indeed refer to *AFC_ AJAX* most often when using the term *Ajax* anywhere on Wikipedia. Of course, this method is not infallible. Twitter users often have a different style of writing than the formal language in Wikipedia articles. For example, *AJAX* in all capitals most often refers to *Asynchronous_ JavaScript_ and_ XML* on Wikipedia, since it is just used as an abbreviation. However, on Twitter people often use words in all capitals to strengthen their message, resulting in the fact that they do actually refer to *AFC_ AJAX* when writing *AJAX*. All in all, mix-ups like these are rare, and the described system will usually return with the correct DBpedia page.

We automatically extract the DBpedia type for every keyword of the events. First, we obtain the right page by the procedure described above. We then specifically look for the `type` of that page and collect the results. We do this for all keywords and **not** for all individual words for a number of reasons:

- Keywords are the most important words of an event, so knowing their type will be the most valuable.
- We do not want to clutter the feature-set with loads of unimportant and uninformative features about general words of an event, possibly even overshadowing the types of the keywords.
- Sending DBpedia queries is a time consuming process; doing this for all words in the total event set of 93,901 events would be impossible.

Every DBpedia type that was present in the combined set of events has its own binary feature, which can either be positive (when present) or negative (not present). We are now able to determine which DBpedia types are the most informative by calculating their chi-squared score. We only show types that occur at least five times in their event set.[6] The results for the three event sets are shown in Table 3.10.

In general, these results seem intuitive. *ClubOrganization*, *SportsTeam* and *Soccerclub* are very indicative of the category *sport* for all three event sets. *TelevisionShow* is evidently a predictor of a *broadcast* event, while *PoliticalParty* is a predictor of a political event. We also observe a few indicators of *software* events in the best event set (*Company* and *Software*) that do not show up in the random event set. On the other hand, the random event set has five very clear indicators of a *public event* (rank 6 to 10) that

[6] This is explained in more detail in Section 3.3.4.

Table 3.10: The best DBpedia type featurevalues for the three event sets, ranked by their chi-squared value.

Rank	Best events	Random events	Combined events
1	TelevisionShow	ClubOrganization	SoccerClub
2	SoccerClub	SoccerClub	SportsTeam
3	SportsTeam	SportsTeam	ClubOrganization
4	ClubOrganization	DutchFootballClubs	DutchFootballClubs
5	PoliticalParty	Company	PoliticalParty
6	SportsLeague	Artist	Activity
7	Company	MusicalArtist	Sport
8	Software	Band	TelevisionShow
9	CreativeWork	Building	TVShow_IBT
10	InternationalOrganization	Festival	Broadcaster

are not present in the best event set. The combined event set has a few indicators that did not show up in either the random and best event set. This is due to the fact that they did not exceed the frequency threshold of five in their individual event set, but did exceed this for the combined event set. For example, *Broadcaster* and *TVShow_IBT* now occurred eight times and are an indicator of a *broadcast* event.

3.4.4 Evaluation of all features

Extracting all the features resulted in a different number of total features per event for each event set. This is the case since we added a feature for each word, user, keyword, DBpedia-type and date in the training set, which is evidently different for every set. The specifics are outlined in Table 3.11.

Table 3.11: The number of features for each group of features.

	Date event	Dates tweets	Users	Keywords	Words	Total
Best events	252	912	15,421	1,282	34,073	52,474
Random events	311	1,122	18,050	723	42,389	63,129
Combined events	494	1,348	32,506	1,832	66,938	103,652

It is remarkable that there are more unique dates for events in the random event set than in the best event set, especially when taking into account that the former set consists

of 362 events, while the latter set consists of 425 events. A probable reason for this is that the best events cluster around certain popular events (Christmas, New Year, Elections), while the random events indeed are a random selection of small events that do not cluster around specific dates. The same logic applies to the number of unique words, a larger variety in events will also lead to a larger variety of words that are used to describe the events. However, this was not the case for the keywords, since there were significantly fewer keywords in the random event set than in the best event set. This is due to the fact that an event in the best event set on average has more keywords (5.2) than an event in the random event set (2.4). Since the best events are so popular, they usually have more descriptors that can be considered as valid keywords. However, the ratio of unique keywords to total number of keywords is still higher in the random-event set (723/865 = 0.84) than in the best event set (1,282 / 2,207 = 0.58), so a larger variety in events will still result in a set of relatively more unique keywords. Note that in this case, the keywords are unique in the sets of best and random events, not necessarily in the full event set of 93,901 events.

Now that all features are extracted, we are able to calculate the most informative features for all three event sets. We do this by calculating the chi-squared value for each feature in relation to the category label. The chi-squared method provides us with a measure of likelihood that two features are independent. Very informative features will not be independent of the category label, since they occur specifically for certain categories, while uninformative features will be independent from the category label, since they occur evenly across all categories. The best 15 features of the best event set and the random event set are shown in Table 3.12. Since a chi-squared test is unreliable on a small sample and since we do not want to show features that obtained a high score due to a coincidence in the training set, we will only show features that occurred at least ten times in the given event set.

Again, these best features seem intuitive, especially for the set of best events. The features that are indicative of a political event are either related to the word *stemmen (to vote)* or are simply names of political parties in the Netherlands. We are even able to deduce that there was probably an election on March 19, 2014, and indeed there were *gemeenteraadsverkiezingen (municipal elections)* on that exact day. We see a few clear clear predictors of *sport* events, namely *wedstrijd (match), #ajax* and *#voetbal (soccer).* The only feature that is not an indicator of a sports or political events is the word *#apple*, which is evidently an indicator of a *software* event. There are four DBpedia features among the best features as well. This shows that these features indeed provided the classifier with valuable information about the category of the events.

TABLE 3.12: The 15 best features for the best and random event set, based on their chi-squared value. Non-word features are in italics. Features are only included if they occurred at least ten times in their event set.

Best events		**Random events**	
Feature	**Category**	**Feature**	**Category**
stemmen	Politics	*ClubOrganization*	Sport
stem	Politics	*SoccerClub*	Sport
19-03-2014	Politics	*SportTeam*	Sport
SoccerClub	Sport	wint	Sport
SportsTeam	Sport	wedstrijd	Sport
#vvd	Politics	2015	Politics
wedstrijd	Sport	seizoen	Broadcast
ClubOrganization	Sport	haag	Sport
#cda	Politics	tv	Broadcast
pvda	Politics	tegen	Sport
#ajax	Sport	kijken	Broadcast
PoliticalParty	Politics	vind	Broadcast
#apple	Software	vieren	Special day
#voetbal	Sport	*Company*	Software
#d66	Politics	elke	Broadcast

For the random events, the DBpedia features even make up the best three features of the whole feature set. The other features are not as clear as in the random event set, but still seem intuitive. We see some of the same features related to *sport*, but the features related to political events seem to have disappeared. The only remaining predictor of such events is the word *2015*, which is not something that would be expected in advance. The number of training instances labeled as *politics* is similar for the best and random event set. This suggests that the political events in the best event set have a clear general theme, while political events in the random event set lack this consistency. Another contrast with the best event set is that we also observe four predictors for *broadcast* in the random event set, namely the words *seizoen (season)*, *tv*, *vind (find)* and *elke (each)*. Finally, we see an intuitive indicator of a *special day*, namely the word *vieren (to celebrate)*.

There were no indicators of *social action* present among the best features for both event sets. The clear indicators of *social action* based on the *tf * idf* score of the words did not occur at least ten times in either event set. If we lower this threshold to five for the best

event set, then *demonstratie (demonstration)* is the 38th most informative feature, while *protest* is ranked 134th. Even with this lowered threshold, these two words do not show up for the random event set. However, we do observe two interesting predictors of *social action*, namely the word *minder (less)* (ranked 21st) and the username of an individual user (ranked 34th). *Minder* does not seem as a clear predictor of a *social action event*. However, manual inspection showed that there were a few events concerning demonstrations about *minder bezuinigingen (less cutbacks) or minder schooluren (fewer school hours)*, resulting in *minder* being an indicator of a *social action event*. The individual user tweeted about six different events, five of which were *social action events*, thus also being a predictor of *social actions*. This shows that sometimes unexpected features are able to provide the classifier with valuable information, although it is questionable whether a single user will generalize to the full event set.

There were also a number of features that did not turn out to be strong predictors of a category. For both event sets, the polarity-score and subjectivity-score were even one of the most uninformative features. Although we expected that especially the polarity-score might be valuable, those features did not bring much extra information to the table that proved to be useful for the classifier. The correlation was probably not strong enough to be useful in classification.

The scores of the periodicity-features are somewhat better, with the binary feature indicating whether an event is periodic scoring around the 250th (top 25%) place of features that had a non-zero value at least ten times. However, the specific scores and type of periodicity did not bear any extra information and scored in the bottom 10% of features.

3.5 Training and testing

Using the extracted feature-sets along with the annotated categories, we are now able to train and test on all three sets of events. The training and testing is performed by two different supervised machine learning algorithms, namely Multinomial Naive Bayes and Support Vector Machines. We will use the machine learning Python module scikit-learn to apply the algorithms (Pedregosa et al., 2011). Finally, we will test the algorithms using 5-fold cross validation. An explanation of the two algorithms will be given in the following section.

3.5.1 Algorithms

Multinomial Naive Bayes is a multi-label variant of the very popular probabilistic Naive Bayes classifier that is based on Bayes Theorem. For each instance, it assigns a probability to each of the possible categories, ultimately classifying the instance as the category with the highest probability. It is important to note that Naive Bayes assumes that all features are independent of each other, meaning that it assumes that every feature is generated independently from the other features. Since this is not the case in natural language and we use a large set of word-features, this assumption is violated. This might lead to a suboptimal calculation of the probabilities. Although there is a large body of research on addressing these resulting inaccuracies (e.g. Webb et al. (2005); Zaidi et al. (2013)), this falls outside the scope of this thesis. Also, as Rish (2001) pointed out, the fact that the resulting probabilities are inaccurate does not necessarily hurt classification accuracy. The classifier will make the correct prediction anyway as long as the correct classification is more probable than any other class. This way, the algorithm is often robust enough to overcome the inaccuracies in the underlying probability calculations.

The other algorithm that is employed is a **Support Vector Machine** (Vapnik, 2013). A Support Vector Machine (SVM) is a supervised learning algorithm that represents the features as points in space, subsequently constructing a hyperplane (or set of hyperplanes) that divides the separate categories in that space by as large a margin as possible. New instances are then classified by which side of the hyperplane they fall on. SVMs have a number of parameters than can be tuned in order to achieve optimal performance. They will be briefly explained below.

The first parameter that has to be set is the kernel, a similarity function that returns the similarity between two instances. We will include 3 types of kernels in the parameter search, the linear kernel, the Gaussian radial basis function kernel (RBF) and the polynomial kernel (based on polynomial regression). The latter kernel has the specific parameter *degree*, which specifies the degree of the polynomial function. The second parameter is the C parameter, which is a method of telling the classifier how general our final hyperplane should be. Very large values of C result in a separating hyperplane that has a very small margin, but does separate (almost) all positive training instances from the negatives instances in the training set. Very small values of C will result in a separating hyperplane with very large margins, resulting in a classifier that misclassifies some of the training instances, but might generalize better to unseen data. The *gamma* parameter is similar to the C parameter. It affects the radius of the area of influence of the different support vectors. When gamma is too large, the support vector will only

influence itself, which leads to no generalization. When gamma is too small, the model is too constrained and thus not able to capture the complexity of the data, resulting in a final classifier that is too general. The exact range of the parameters that are tested is shown in Table 3.13.

TABLE 3.13: The exact parameter ranges that are tested in the SVM parameter optimization phase.

Parameter	**Range**
Kernel	{linear, RBF, polynomial}
Degree	{0, 1, 2, 3, 4}
C	{0.001, 0.005, 0.01, 0.1, 0.5, 1, 2.5, 5, 10, 15, 25, 50, 75, 100, 500, 1000}
Gamma	{0.0005, 0.001, 0.002, 0.008, 0.016, 0.032, 0.064, 0.128, 0.256, 0.512, 1.024, 2.048}

It has to be noted that, although it is interesting to compare the different properties of each classifier, in the end it comes down to which algorithm performed best. This is not a thesis that is focused on comparing the different algorithms, the algorithms are merely a tool to achieve optimal performance. Therefore, aside from a brief comparison, we will only show the results of the best performing algorithm in the Results section. The optimal parameter values of the SVM classifier will be shown in Table A.4 in the Appendix section.

Both algorithms will be trained and test by applying 5-fold cross validation. This means that we will randomly divide the event set in five equal subsets and test on all five individual subsets after training the algorithm on the other four subsets. The results of each subset are then averaged out. For the SVM parameter search, we perform 5-fold cross validation for each combination of parameters, randomizing the folds for each of those combinations. For Naive Bayes, we perform the cross validation once, since it does not have a set of parameters that needs to be optimized.

3.6 Evaluation metrics

The performance of the classifiers is measured by accuracy, precision, recall and F1-score. Since we face a multi-class problem, we have to decide whether we use the micro average, macro average or weighted average of the precision, recall and F1-score. The **micro** average takes all individual instances into account without calculating class-averages first. For a multi-label problem, this means that we cannot differentiate between

precision and recall, since they are equal to the final accuracy. Another disadvantage of this measure is that the final scores are too heavily reliant on the most dominant class, since it does not take class averages in account. The **macro** average does take these individual averages into account. However, this is merely the average of all individual classes, thus overvaluing classes with only a few instances. For this reason, we opt for the weighted average. In that case, the precision, recall and F1-score are weighted by the number of true instances of the classes, calculated as follows:

$$average(S) = \frac{\sum_{lab \in L} T_{lab} * S_{lab}}{T_{total}}$$

This is the equation for the weighted average of the three scores S (precision, recall, F1-score), with T_{lab} as the number of true instances for label *lab*, S_{lab} as the score for that specific label and T_{total} as the total number of instances in the set. In essence, the score of all labels is weighted by its number of instances. For the weighted average, this means that it is also possible that the average F-score is not between the average precision and recall.

It is important to note that this way of calculating the average recall is the same as calculating the overall accuracy. The weighing of the percentage of instances that are correctly recalled by the total number of instances for that category is exactly the same procedure as dividing the number of correctly classified instances by the total number of instances. In the Results section we use both terms depending on which term is more appropriate, but one should keep in mind that those results are in fact identical.

Other evaluation metrics are accuracy-at and F1-score-at scores. These metrics are usually known as precision-at in binary classifications tasks (e.g. Kunneman and van den Bosch, 2015). Since the Naive Bayes classifier calculates the probability of each classification, we are able to rank the classified instances in order of these probabilities. This provides us with a list of instances that is ordered by the confidence of the classifier that the given prediction was correct. We are therefore able to calculate the accuracy and F1-score for ranked subsets of instances, measuring whether the classifier indeed predicted the instances with higher probabilities better than the instances with lower probabilities. This could for example tell us whether or not an event classified as *social action* is actually very likely to be a serious event, or that it just barely made the cut and is thus likely to be wrongly classified. The F1-score-at and accuracy-at graphs of the three event sets are shown in Figure A.2 and Figure A.3 in the Appendix.

3.6.1 Annotation of evaluation set

Another way of assessing the performance of the classifier is to perform a test on an annotated set of events that was never used in the training phase. This is considered to be the most reliable method to estimate the performance, since we are now sure that the results do not stem from an over-fitted, highly specific classifier that only performs well on a specific set of events. Since our goal was to automatically categorize all events, we will perform the classification procedure for the set of all events, with the previously annotated events used in the training phase excluded. This resulted in a set of 92,701 events. We will refer to this event set as the **full event set**.

Since we have three different training sets (best events, random events and the combined events) we are able to classify each event three times based on the different training sets. This means that we ultimately obtain three classifications per event. It is not necessarily the case that these classifications are equal. However, we do want to end up with one final category for each event. Therefore, we construct a minimal ranking of confidence of classification for these events. If an event obtained the same classification three times, we are fairly sure that that is the correct classification, while if an event obtained three different classifications, we barely have an indication to which category it belongs to. Events that obtained three different classifications are therefore discarded. The other events receive the category that occurred at least two out of three times as their final classification.

The most obvious next step would be to randomly select a subset of those events and annotate them to create an evaluation set. However, we know that *public event* is a dominant category in the training set, resulting in the fact that it will also be dominant in the full set of classified events. This means that if we simply randomly picked events for the evaluation set, we would end up with a set full of public events and only a few instances of other categories. Even though the classifier produced this majority of public events, it might still produce valuable results **if** the other (non public event) classifications are often predicted correctly. This way, we are still able to successfully predict events of certain categories, although we often overlook them as well.

This means that we are interested in the **precision per category** for all categories. The only way to calculate this precision per category is to select a sufficient sample of events of each category for annotation. Due to the dominance of the public events, we would have to annotate an enormous amount of events to obtain such samples if we would randomly select them. However, since we are mainly interested in precision, we simply select the events based on their classification. For all categories, we randomly

select 50 events that were classified as that category. However, we do add all 93 events classified as *social action*, since we are especially interested in that category. Also, the category *advertisement* only occurred 25 times in the full event set. This ultimately resulted in a total **evaluation set** of 468 events.

Note that the average precision, recall and F1-score will not be indicative of the full event set, based on this procedure. This is the case since we cherry-picked the events that already had the same classification two out of three times and also because, as described earlier, the resulting evaluation set is not a representative sample of the full event set. In a way, we sacrificed these overall scores to be able to determine a (more interesting) specific precision per class. The results of the described procedure are described in Section 4.5.

3.7 Accuracy increasing methods

There are a number of ways in which we can adjust the training and testing procedure to achieve optimal performance. We will perform three of these methods, which are explained in detail below.

3.7.1 Down-sampling the dominant class

A general problem in machine learning is that often one of the classes is represented by a large number of samples, hindering the performance of the minority classes. A possible solution is to randomly down-sample the biggest class, discarding instances of the dominant category until it is as large as the second most frequent category. This method is often performed for imbalanced datasets (e.g. Shriberg et al., 2000, Lewis and Catlett, 1994) and is described as one of the five solutions in the systematic study on imbalanced data sets of Japkowicz and Stephen (2002). Since we are especially interested in the performance of one of those classes (*social actions*), we apply this procedure on our data sets, even though in our case the imbalance problem is not as prevalent as in some other cases (e.g. in Lewis and Catlett (1994) only 0.2% of the instances are positive).

For each of our three event sets, we will randomly down-sample the dominant *public event* class until it contains as many examples as the second most dominant class. The specifics are shown in Table 3.14. It turns out that for the set of best events, only 27% of the total number of public events gets discarded. This probably means that down-sampling will only have a minor influence on this set. However, the other two sets discard about two thirds of their public events and close to 30% of their total number of events. This might actually make a big difference, since the clear dominant class has now disappeared.

TABLE 3.14: The specific amount of events that get discarded for each set when performing the down-sampling procedure.

Event set	**Total instances**	**Instances most frequent class**	**2nd most frequent class**	**Number of events discarded**
Best events	425	126	92	34 (27.0%)
Random events	362	161	55	106 (65.8%)
Combined events	787	287	92	195 (67.9%)

However, since we perform cross validation, we now face the problem of when to down-sample, especially when using the SVM classifier. It is possible to down-sample in advance and perform the parameter search with the exact same data set ór randomize the down-sampling for every set of parameters. We have to keep in mind that we are dealing with very tiny changes in parameter values in the parameter optimization phase, meaning that if we change the sampling of the training instances every cycle, we are no longer able to make a fair assessment of the influence of those slight changes. Also, a few test runs showed that there is little to no difference in performance when using a different set of instances for the *public event* category. This is why we chose to do the down-sampling in advance and perform the parameter optimization on the same set. Another option might have been to perform the parameter optimization phase for multiple randomly down-sampled sets of events and then to average over the results. However, since it is such a large search grid coupled with the fact that performing a classification task with a huge amount of features (>50.000) is a time consuming process, this turned out to be impossible to execute in a reasonable time frame.

For each of our three events set, we perform the down-sampling in advance. We perform an SVM grid search as described in Section 3.5 and ultimately test the Naive Bayes and SVM classifier using 5-fold cross validation on these down-sampled data sets. The results of the best scoring classifier and the effects of the down-sampling are shown in the Results section.

3.7.2 Feature pruning

The events in our three data sets are represented by a large number of features. On the one hand this is very useful, since it captures as much information about the events as possible. On the other hand it also comes with a lot of features that are probably not very helpful to the classifier. This might even hurt performance due to over-fitting. As discussed before, our data sets are especially prone to over-fitting due to the low number of instances compared to the number of features per instance. To combat this, we test

a range of different features that are selected based on their predictive score using a chi-squared test. We will evaluate the performance of the classifiers ranging from using just a small number of only the best features to virtually all features.

The features are ranked based on the same chi-squared procedure as was described in Section 3.4.2, only without the requirement of occurring at least ten times with a non-zero value. This ranking of features is then used to select the subsets of features based on how many features we want to keep in. For every subset of features, we train and test as described in Section 3.5, the only difference being that for the SVM classifier we immediately apply the parameters that were found in the previous grid search. The smallest number of features we use is 1,000 and we then increase this set with 2,000 features for each new measurement. This results in a range of `[1,000, 3,000, 5,000, 7,000,..,num_features]`. We calculate accuracy, precision, recall and F1-score for each of these subsets. For each event set, the results of the best classifier are shown in the Results section.

3.7.3 Performing bag-of-words classification in advance

Another option for reducing the feature space is performing the bag-of-words classification in advance. The resulting classification is then added as a feature to the remaining feature set without the word features. Bag-of-words classification is a classification that is often used in document classification tasks (e.g, McCallum et al., 1998). It creates a very simple representation of the documents by only taking the words themselves into account and ignoring other information such as grammar, word order or part-of-speech tags. The classification is performed by only using these straightforward features and is often used as a baseline to evaluate performance of more sophisticated feature sets.

The advantage of performing the bag-of-words classification in advance is that we are able to greatly reduce the feature space, while not disregarding the information of those features. The clear disadvantage is that it also often returns with the wrong classification, resulting in the addition of the wrong feature to the feature set. An additional benefit is that we are now able to compare the results of the classifier using all features, using just the bag-of-words features and using the classification added as feature. This will show the influence of our more abstract features, since the classifiers should perform better when having access to all features as opposed to having access to only the word features if the abstract features bear any additional information.

We will again perform the described grid search for the SVM classifier. The optimized parameters are then used for the final testing, together with the Naive Bayes classifier

that does not have such a parameter optimization phase. The bag-of-words classification is performed first with optimal parameter values and the resulting classification is added to the feature-set without the word-features. Note that in our case the user-features are treated as words as well, which further decreases the feature space. After performing the same parameter search for SVM, we are able to train and test both algorithms using the new feature-set. We will report the results of both classifiers in the Results section.

Chapter 4

Results

In this section, we first summarize the results of categorizing the selected event sets. We show that it is easier to classify the best events than the random events, but that we obtained the best F1-score for the set of combined events. The best results were obtained by doing the bag-of-words classification in advance and down-sampling the most dominant class in the training set. This is followed by the results of testing on the full event set after training on the three labeled event sets. The performance is evaluated by using a balanced evaluation set that was never used in the training phase. We show that the precision of the non-dominant categories is high, but that the recall of those categories is low. This includes the category *social action*, which obtained a high precision of **0.80**. However, the recall of *social action* is estimated to be low. We only predicted a *social action* for 0.2% of the events in the full event set, while this was expected to be 3.3%. However, by utilizing the obtained Bayesian probabilities, we were able to increase the recall by 232%, while only decreasing the precision by 14%.

4.1 Categorizing the selected event sets

Recall that we have three different event sets: the best events based on the eventscore, the randomly selected events and the combination of the previous two sets, the combined events. In Table 4.1 we present the most important results, namely the precision, recall (accuracy) and F1-score for each event set, also showing the specific scores for different feature-sets and for whether or not the data was down-sampled. In this table, we also compare the performances of the two different classifiers. In all other tables and figures, we will only show the scores of the best performing classifier.

It is evident from Table 4.1 that the Naive Bayes classifier greatly outperformed the SVM classifier.[1] For literally all test procedures, the Naive Bayes classifier produced a

[1] Optimal parameter values are shown in Table A.4 in the Appendix.

TABLE 4.1: Summary of the most important results when categorizing the three selected event sets.

			Down-sampled			**Not down-sampled**		
			Prec	Rec	F1	Prec	Rec	F1
Best events	All features	Bayes	0.65	0.60	0.56	0.68	0.61	0.57
		SVM	0.68	0.41	0.37	0.65	0.41	0.34
	Only bag-of-words	Bayes	0.67	0.59	0.55	0.65	0.58	0.54
		SVM	0.64	0.43	0.38	0.66	0.41	0.35
	Classification bag of words added as feature	Bayes	**0.67**	**0.67**	**0.65**	0.62	0.63	0.60
		SVM	0.66	0.49	0.48	0.67	0.54	0.51
Random events	All features	Bayes	0.62	0.58	0.55	0.62	0.60	0.56
		SVM	0.26	0.26	0.15	0.28	0.45	0.28
	Only bag-of-words	Bayes	0.61	0.60	0.57	0.62	0.59	0.56
		SVM	0.59	0.30	0.26	0.33	0.45	0.29
	Classification bag of words added as feature	Bayes	**0.64**	**0.60**	**0.58**	0.56	0.57	0.52
		SVM	0.30	0.31	0.22	0.46	0.46	0.30
Combined events	All features	Bayes	0.70	0.69	0.67	0.69	0.66	0.63
		SVM	0.67	0.41	0.39	0.63	0.45	0.36
	Only bag-of-words	Bayes	0.70	0.67	0.65	0.67	0.64	0.61
		SVM	0.65	0.49	0.43	0.62	0.43	0.32
	Classification bag of words added as feature	Bayes	**0.69**	**0.69**	**0.68**	0.68	0.55	0.65
		SVM	0.58	0.43	0.43	0.61	0.51	0.48

higher F1-score and accuracy. On the one hand this is a surprising result, since Naive Bayes is generally seen as a less sophisticated classifier. On the other hand, this is not so surprising if we consider the fact that we already noted that the SVM is prone to over-fitting due to its high ratio of features to instances. This result suggests that, although we performed an extensive parameter search, this did not stop the classifier from over-fitting and thus resulted in a poor performance. For this reason, we only discuss the results of the Naive Bayes classifier in this section. The main conclusions are listed below.

(1) It is easier to categorize the set of events with a high eventscore than it is to predict the random subset of events, but there is no significant effect of eventscore in general.

We see that it was considerably more difficult to predict the random events than it was to predict the best events, with a best F1-score of respectively 0.65 and 0.58 for the best and random event sets. We see this large difference if the bag-of-words classification is performed in advance. This suggests that the very large

and popular events are also more easily distinguished by category. Interestingly enough, the random events obtained a higher F1-score if we only take the bag-of-words features into account.

However, the best event set is only a small subset of the total set of 93,901 events. We are also interested whether there is an effect of eventscore in general. We investigate this in the representation of the full event set, namely the random event set. First, we check if we obtain a higher F1-score if we only retain events that exceed a certain eventscore-threshold. These results are shown in Figure 4.1.

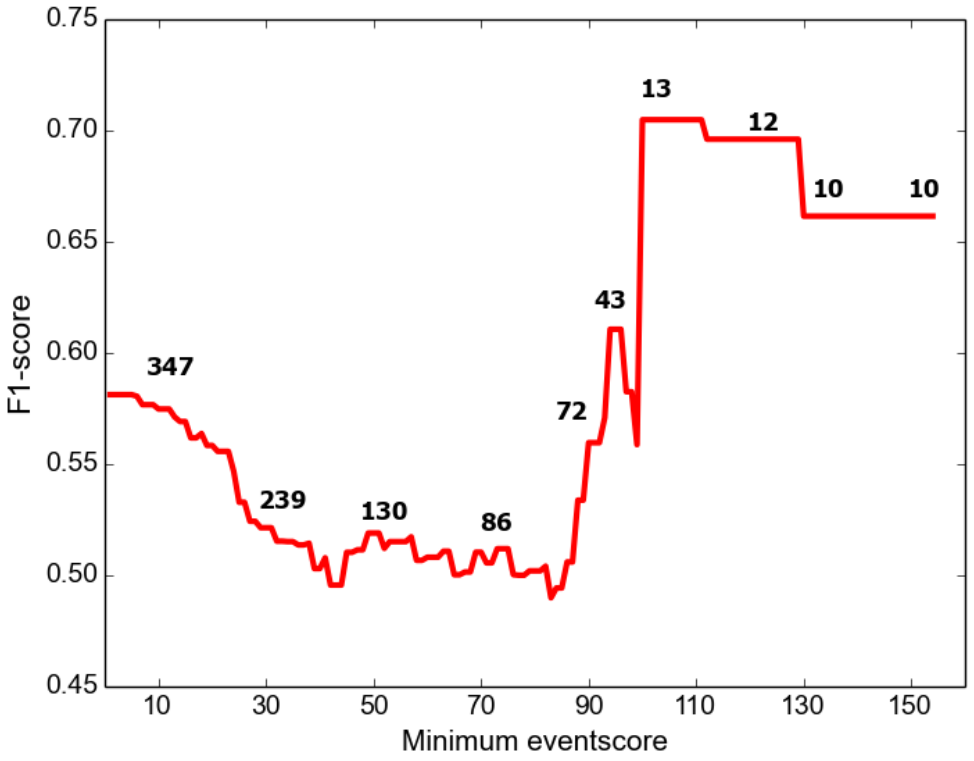

FIGURE 4.1: The F1-scores for subsets of random events that exceeded the eventscore threshold. The number of retained events is shown in bold.

We see no sign of a large, general eventscore effect. Excluding events based on their eventscore even hurts performance, until we reach an eventscore of approximately 100. However, this only concerns a small subset of events, since only 13 events were retained to achieve this F1-score (bolded numbers in Figure 4.1). We also performed a least-squares logistic regression test and found no significant[2] correlation between eventscore and whether or not the event was classified correctly, $r(360) = -0.05$, $p = 0.39$. We conclude that there is a small group of events with (very) high eventscores that is easier to classify, but that there is no significant effect of eventscore in general.

[2]The significance threshold is set at 0.05 in this thesis.

(2) Using more training data improved performance.

We obtained the best performance when combining the best and random events, with a best F1-score of 0.68. This is a 4.4% increase in F1-score over the best event set and even a 14.7% increase in F1-score over the random event set. The addition of more training instances thus resulted in better categorization.

(3) Precision of predicting social actions is high, but recall is low.

TABLE 4.2: The precision, recall and F1-score for the category **social action** for each of the event sets.

	Precision	**Recall**	**F1-score**
Best events	0.79	0.44	0.56
Random events	0.40	0.17	0.24
Combined events	0.74	0.38	0.50

Table 4.2 shows the precision, recall and F1-score for the category *social action* for each of the event sets, down-sampling the dominant class and performing bag-of-words in advance. We see that the precision is fairly high, but that the recall is lower. This means that the classifier will often miss a *social action event*, but if it predicts one, it will generally do so correctly. The random event set had only 12 instances of *social action* in the training set, which might explain the lower precision and recall. However, we also obtained a low estimated recall for *social action* in the evaluation set. Section 4.2.2 describes a method that largely increases the recall, while only slightly decreasing the precision.

(4a) Performing bag-of-words classification in advance resulted in better performance.

The best scores for each event set were all obtained doing the bag-of-words classification in advance and then adding that classification to the set of other features. This even resulted in an increase in F1-score of 16.1% for the best events. For the random and combined event set this effect was smaller, with an increase in F1-score of respectively 1.8% and 1.5%. It is interesting to note that for the random event set, only using the bag-of-words classification offered better results than using all features, with F1-scores of respectively 0.57 and 0.55. This means that for the random events, the addition of, for example, the DBpedia features did not result in better classification. This suggests that the most valuable information about the category of an event is already captured in the words themselves and that the addition of more abstract features only slightly increased performance.

(4b) The down-sampling of the most dominant class resulted in better performance.

For each event set, the down-sampling of the most dominant class resulted in an increase of the final F1-score. The random event set obtained the largest increase in F1-score of 11.5%, followed by the best and combined events with increases of respectively 8.3% and 4.6%. This means that down-sampling the most dominant class indeed increased the performance of the minority classes, resulting in a better overall F1-score.

(4c) Feature pruning did not result in better performance of the classifier.

The results of the feature pruning method are shown in Figure 4.2. They show the accuracy and F1-score for the three types of event sets when only using subsets of the most informative features, selecting these features by performing chi-squared feature selection. We see that in general, using fewer features would not be beneficial for the classifier, since the peak values of accuracy and F1-score barely exceed the final scores when using all features. For the random events, we observe a strange peak around 30,000 features that quickly disappears again. Since there is not even a slight trace of such a peak in the set of combined events (which contains all the random events), this is probably a random occurrence, meaning that it does not point to a subset of very informative features that somehow received mediocre scores in the chi-squared selection. However, for the combined events we see an early peak in F1-score that is comparable to the final F1-score at around 12,000 features. This is interesting, since it means that the additional 90,000 features did not carry additional information for the classifier.

The combination of feature pruning and down-sampling did not necessarily result in improved performance.[3] Although these methods might independently offer increased performance, the combination of these methods might actually hurt performance. This shows that future event categorization systems should carefully select these performance increasing methods, instead of automatically assuming that they will increase the overall performance.

[3]The results are shown in Figure A.1 in the Appendix.

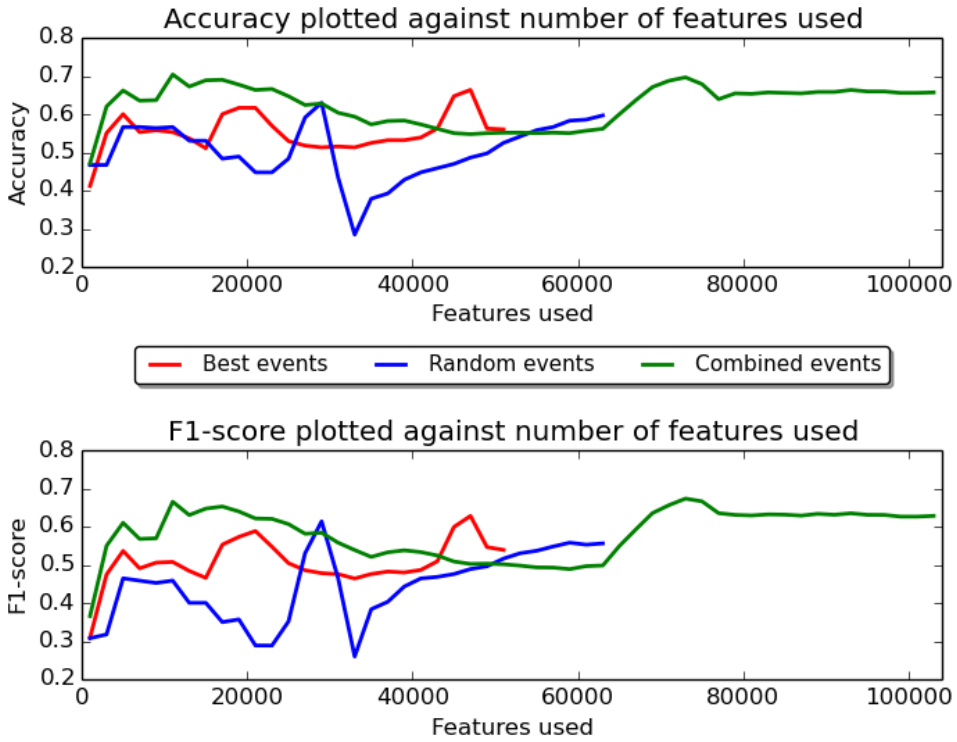

FIGURE 4.2: The accuracy and F1-scores when using chi-squared feature selection.

4.2 Categorizing the full event set

As described in Section 3.6.1, we tested the classifier on the full event set of 92,701 events, while training on each of the three event sets (best, random and combined). This means that we receive three different classifications for each event in the full event set. Table 4.3 shows the agreement of the classifiers. It turns out that all three classifiers had a different classification for only 308 out of 92,701 events. This means that the majority of the classifiers agreed on the classification for 99.7% of the events in the full event set. The 308 unclear events were discarded from the event set and did not receive a final classification.

All three classifiers agreed on the classification for 77,192 out of the 92,701 events (83.3%), which seems very high at first sight. However, only 2,918 events obtained the same classification three times if we filter out events that were classified three times as a *public event*. But, if we then remove events that obtained *public event* as a classification at least once, we see that the classifiers agreed on classification on 2,918 out of 3,368 events (86.6%), which is again very high. Note that this agreement is only an indication that the classification is correct, not that the these events are definitely classified correctly. It also only occurred 18 times out of 92,701 that the classifier produced three different classifications where none of these classifications was *public event*.

We only retained the events that obtained at least 2 out of 3 equal classifications. The

TABLE 4.3: Comparison on how often the 3 classifiers agreed on the classification. Note that 1 out of 3 times the same annotation effectively means 3 different annotations.

	Same annotation		
	3/3	**2/3**	**1/3**
All events	77,192	13,898	308
Filtered 3 times public event	2,918	13,898	308
3 times not a public event	2,918	432	18

TABLE 4.4: The number of final classifications (events) for each category.

Category	**Events**
Sport	2,771
Politics	2,170
Broadcast	206
Public event	81,538
Software	1,630
Special day	1,722
Social action	**93**
Advertisement	25
Other	1,535

division of final classifications is shown in Table 4.4.[4] We see that *public event* is by far the most frequent category. However, for most other categories there is still a fair number of classifications left. Unfortunately, we only identified 93 *social actions*, while there were also only 25 *advertisements* detected.

As described before, we created the evaluation set by selecting 50 classified instances of each category, with the exception that we did include all 93 instances of a *social action* and could only use 25 instances of *advertisement*. This resulted in an evaluation set of 468 events. These events were then annotated by the authors, so we could calculate the precision per class. 395 out of these 468 events were annotated as an actual event (84%), which is considerably higher than the 61.8% of events that were an actual event in the random event set. This suggests that if the classifier does not categorize an event as a *public event*, it is more likely to be an actual event. The non-events were discarded from the evaluation set. Using the remaining events, we calculated the precision per category. These scores are shown in Table 4.5.

[4]The division of categories per class for each event set is shown in Table A.5 in the Appendix.

TABLE 4.5: Precision, recall and F1-score for every category in the **evaluation set**.

Category	**Precision**	**Pred**	**True**
Sport	1.00	41	46
Politics	0.86	44	47
Broadcast	1.00	47	52
Public event	0.57	36	34
Software	0.96	46	48
Special day	0.78	36	39
Social action	**0.80**	**85**	**70**
Advertisement	0.51	20	17
Other	0.70	39	42

In general, these scores seem very high for a 9-class classification task. The precision per class is even 1.00 for *sport* and *politics*, meaning that if the classifier predicted those categories, it did so perfectly. Every category obtained a precision of at least 0.70, except for *public event* and *advertisement*. For *public event*, this is what we expected. This category was even more dominant in the classification on the full event set than it was in the annotated gold standard. It is then likely that a lot of these classified events actually belonged to a different category. This is exactly what we see in the results. The classifier was only 57% of the time correct when it classified an event as a *public event*, meaning that it wrongly classified an event as *public event* 43% of the time.

We are able to compare these results to the precision per class of the **random event set**. This is our representation of all events, so we expect the results to be similar. The precision scores are less reliable, since they were obtained by testing fewer instances per category. However, we do obtain a recall score per category for this set, while this is not the case for the evaluation set. The results are shown in Table 4.6.

We see that the results are similar: high precision for *sport*, *politics*, *broadcast* and *software*, while obtaining a lower precision for *special day* and *other*.[5] For the high-precision categories, the recall is a lot lower. *Sport* and *public event* are the only two categories that obtain both a high recall and a high precision. Even though the results are similar, there is not really a large bias towards *public event* for the random event set. Therefore, we suspect that the recall of non-dominant categories is even lower in our

[5]We are not able to compare the category *advertisement*, since it was removed from the random event set due to its infrequency.

TABLE 4.6: The precision, recall, F1-score, number of true instances and number of predicted instances per class for the **random events** with down-sampled data and performing bag-of-words in advance.

Category	**Precision**	**Recall**	**F1-score**	**True**	**Pred**
Sport	0.83	0.69	0.75	35	29
Politics	0.83	0.17	0.28	30	6
Broadcast	0.85	0.37	0.51	30	13
Public event	0.67	0.81	0.74	161	195
Software	1.00	0.44	0.61	16	7
Special day	0.50	0.13	0.21	23	6
Social action	0.40	0.17	0.24	12	5
Other	0.37	0.67	0.47	55	101

evaluation set. In the next section, we estimate the recall by comparing the expected and actual frequencies of classification.

4.2.1 Estimating the recall

Since we focused on precision per class in the evaluation set, we are not able to reliably calculate the recall of the individual categories in the evaluation set. We cannot solely rely on the recall of the categories in the random event set, since the large bias towards *public event* is not present there. However, we are able to **estimate** the recall of the evaluation set by comparing the expected frequency of classification to the actual classifications.

The expected frequencies are obtained by the frequencies of categories in the random event set, since this is the representation of the full event set. For example, an event was annotated as *politics* for 8.3% of events in the random event set. This means that we expect that 8.3% of events in the full event set are actually political events. However, we only classified 2.3% of the events in the full event set as *politics*. This means that we already failed to recall a lot of *politics events*.

However, we should also incorporate the fact that approximately 37% of the events in the full event set are actually a non-event. This makes it an unfair comparison, since the non-events were removed from the random event set. It is very likely that these non-events will be classified as the majority class *public event*, since they did not belong to a clear other category. For this reason, we calculate the actual frequencies of classification based on the event set that is 37% smaller than the full event set of 92,701 events, as if

the non-events were removed in calculating the frequencies. This leaves a set of 58,402 events. As shown in Table 4.4, we classified 2,170 events as a political event. The actual frequency of *politics* is thus $2,170/58,402 = 3.7\%$. This is still not the full picture, since these predictions were only estimated to be correct 86% of the time (as observed in Table 4.5). This means that of the 3.7% of recalled political events, only 3.2% is estimated to be actual political events. Therefore, our estimated recall for *politics* is $3.2/8.3 = 0.33$. We applied this method of estimating the recall on each category. The results are shown in Table 4.7.

TABLE 4.7: Estimating the recall by using the expected and actual percentage of classifications.

Category	**Annotated in random events**	**Predicted in full event set**	**Estimated recall**
Sports	9.7%	4.7%	0.49
Politics	8.3%	3.2%	0.33
Broadcast	8.3%	0.4%	0.04
Public event	44.5%	80.9%	1.00
Software	4.4%	2.8%	0.61
Special day	6.4%	2.9%	0.36
Social action	3.3%	0.2%	0.04
Advertisement	0.8%	0.04%	0.02
Other	15.2%	2.6%	0.12

We see that the recall of the non-dominant categories differs greatly. For *sports*, *politics*, *software* and *special day* the recall is between 0.33 and 0.61, meaning that we recalled a decent amount of events that belonged to one of those categories. However, for *broadcast*, *social action*, *advertisement* and *other*, the estimated recall is smaller than 0.12, meaning that we expect to fail to recall at least 88% of the events that belong to those categories. Especially for *social actions* this is disappointing, since a system that fails to detect 96% of *social action events* has little practical value. For this reason, we applied a method to increase the recall of *social actions*. This method is described in the next section.

4.2.2 Increasing recall of social actions

The main problem with our *social action event* detecting system is that it has a low recall. A method for increasing the recall is the investigation of the different probabilities that the Naive Bayes classifier returned. Events where *social action* obtained the second highest probability are now completely ignored, since the event got classified as a

different category. However, we could simply classify all events where *social action* had the **second highest** probability as a *social action* as well. We will refer to these events as **secondary social action events**. These events might not be as likely to be a *social action*, but this will greatly improve the recall of such events. However, it might hurt the precision in an equal manner.

We collected all events for which at least two out of three classifiers returned *social action* as the second most likely category. This resulted in a set of 226 events, which we will treat as if they were classified as *social action*. We annotated these events following the same procedure as described in Section 3.3. This way, we are able to calculate the precision of the classifier on this set of *secondary social actions* and measure how it will influence our total precision for identifying *social action events*. The results of the annotation are shown in Table 4.8. It turned out that 26 of the 226 events were annotated as a non-event. Therefore, these events are excluded from the event set.

TABLE 4.8: The results of the annotation of the *secondary social action* events.

Category	**Annotations**	**Category**	**Annotations**
Sports	5	Special day	2
Politics	4	Social action	130
Broadcast	3	Advertisement	0
Public event	44	Other	12
Software	0	Non-event	26

We see that 130 of the 200 events were indeed annotated as a *social action*, resulting in a precision of **0.65**. This is still high, especially when we take into account that all these obtained events only returned *social action* as the second most probable category. Adding these events to the *social action* events in the evaluation set results in a drop of total precision from 0.80 to 0.69. This is very minor, considering that the total recall more than tripled. This is the case since we first recognized 56 *social action events* that were actually an event.[6] This procedure added a total of 130 *social action* events, thus increasing the recall by **232%**, while the precision only dropped by **14%**. This seems to be a great method of increasing the recall, while only slightly hurting the precision.

[6]We know this specifically since we annotated each of the 93 *social actions events* obtained from the classifications of the full event set.

Chapter 5

Discussion

In this section, we first discuss the results of the general categorization system. We discuss our main findings and extensively compare our results to the results of Ritter et al. (2012). We offer comparable performance on categorizing the events with the best eventscore and suggest that we outperformed them on categorizing the full event set. This is followed by a comparison of predicting *social actions* to the known approaches that tried to forecast *social action events*. We offer comparable performance to the Twitter-only approach by Muthiah et al. (2015), but got outperformed by the deployed EMBERS system that uses a large variety of data sources (Ramakrishnan et al., 2014), mainly because of our low recall. We then discuss the results of a method that largely increases the recall of *social action events*, while only slightly hurting precision. We conclude with proposing a few directions for future research, including the transformation to a binary problem and the use of unsupervised learning as a feature for the supervised approach.

5.1 Discussion of the general categorization system

In this section we discuss our main findings of the general categorization system. They are listed below.

5.1.1 Effect of the eventscore

We found that it is easier to classify events with a high eventscore, since the best events outperformed the random events by 12.1%. This suggests that it is easier to categorize an event if it is larger, clearer and more popular in general. This might be simply an effect of the amount of data that was available for the classifier. The best event set contained 425 events, which is 17.4% more than the 362 events in the random event set. However, combining the random and best events only resulted in a 4.4% increase in F1-score for only using the best events. This suggests that the large, popular events

are indeed easier to categorize.

Even though the best event set obtained a higher F1-score, we found no significant effect of eventscore in general for the random event set. This means that a small sample of very high-scoring events is easier to categorize, but that this effect does not occur when classifying all events. We also found that excluding events with a low event-score even decreased performance. This means that we are not able to use eventscore to exclude the unclear and smaller events from categorization in advance, since we found no evidence that these events are more difficult to categorize.

In general, an F1-score of 0.58 (obtained by the random events) for a 9-class classification problem is still high. This shows us that automatically categorizing Twitter events is not an impossible task that can only be achieved by selecting events that have a high eventscore. The results are especially promising when we consider that they considerably improved when using more training data. This is described in the next section.

5.1.2 More training data improved classification

We found that using more training data resulted in better performance of the classifier, since we obtained the best F1-score for the set of combined events. This is as expected, since this set had twice as much training data as the other two event sets. This result is very promising when considering future applications. It shows that there is not necessarily a boundary on classification accuracy due to the nature of the events, since a larger training set resulted in better performance. This is often the case in machine learning tasks, but in this special case it could have been possible that the results were limited due to the events simply not being distinguishable enough, even for human annotators. We observed in the annotation process that annotators agreed on the category of an event 91.3% of the time. This means that our highest F1-score of 0.68 is not yet close to ceiling performance. These results suggest that we are still able to close this gap by obtaining more training data and that we do not have to solely rely on the extraction of more sophisticated features.

5.1.3 Minor contribution non bag-of-words features

Performing the bag-of-words classification in advance improved the final F1-score for all three event sets. This suggests that reducing the feature space helps preventing the classifier from over-fitting on the training data. However, it is surprising that only using the bag-of-words features only slightly decreased the final F1-score. This means that the addition of all other features only had a minor influence on the performance. This includes features that were deemed valuable in advance, such as the eventscore, the

polarity value and the DBpedia features. Especially for the DBpedia features this is surprising, since a few of those features were among the best features according to their chi-squared value. This suggests that the information captured by the DBpedia features is often already captured by the individual words.

For example, let's say we come across the phrase *PvdA* in a to-be-classified event. DBpedia will generalize this phrase to *PoliticalParty*, which is a great indicator of a political event. However, it is highly likely that the word *PvdA* itself already is an indicator of a political event. This means that the classifier does not benefit from knowing that *PvdA* is a *PoliticalParty* if there are enough instances of *PvdA* in the training set. The results suggest that this is often the case for the DBpedia features, resulting in only a slight improvement over the bag-of-words classification. For this reason, we believe that the addition of more sophisticated features might only increase performance if they describe a concept that is not easily captured by the words themselves.

5.1.4 The effect of down-sampling and feature pruning

Feature pruning did not result in a large improvement of performance. However, for the combined events, the results reached the best performance when only using a subset of the 12,000 best features. This means that the remaining 90,000 features were not beneficial for the classifier. An advantage of using fewer features is that the classifier will be less likely to over-fit on the training set. For this reason, we prefer to use a smaller number of features if the final performance is equal. For a general categorization system, this means that we should carefully select the features to include, instead of trying to capture as much information as possible.

We obtained the best results for each event set when we down-sampled the dominant category and performed the bag-of-words classification in advance, with an F1-score of respectively **0.65**, **0.58** and **0.68** for the best, random and combined event set. This decreased the dominance of the dominant *public event* category and improved the performance of the minority categories. For a general categorization system, this means that it might be beneficial to train on equal samples of all categories, instead of training on a set of training instances that reflects the actual distribution of categories across all events.

Performing bag-of-words in advance combined with the down-sampling of the most dominant class resulted in the highest F1-score for all three event sets. Therefore, we use these results in comparison with other general categorization systems as well as in comparison with other system that specifically tried to predict *social action events*.

5.1.5 Categorizing all events is difficult

The previously discussed results were all obtained by training and testing on selected subsets of events that were annotated by category. We were also interested in categorizing the full set of 92,701 events. It turned out that this resulted in a large bias towards the category *public event*, categorizing 88% of all events as such, while this was expected to be only 44.5%. We were able to estimate the recall for the non-dominant categories by comparing the actual and expected frequencies. It turned out the estimated recall of these categories is low. For all categories except *public event*, the estimated recall is lower than the precision. The estimated recall of *broadcast*, *social action*, *advertisement* and *other* is even lower than 0.12. However, the recall of *social action* is increased by 232% by applying a method that utilizes the actual Bayesian probabilities, which will be discussed in Section 5.4.

However, it is important to note that this does not paint the full picture. Since there is no reliable method of automatically discarding non-events, the full event set contained a lot of instances that were not actually an event. Remember that only 62% of the 586 randomly selected events were actually annotated as an event. This means that 38% of the events in the full event set are estimated to not be an actual event. In the training phase, these non-events were excluded, which means that it is very likely that the classifier will simply assign these non-events in the full event set to the most frequent category. This suggests that a large part of the bias to *public event* is due to the occurrence of non-events in the full event set. This means that if there is a reliable way to automatically exclude non-events, the results of the general categorizing would considerably improve.

In future research, a solution for this problem might be to simply annotate these events non-events as a separate category, instead of excluding them in the annotation and training phase. This way, we train the classifier to detect non-events and are able to remove these instances from the final results. This might decrease the bias towards *public events*. However, it probably also results in a number of actual events that get incorrectly classified as a non-event, which means we lose these actual events in the process.

5.1.6 High precision of non-dominant categories

Due to the large bias towards *public event*, we focused on the precision per class of the non-dominant categories in our evaluation set. The results of this set showed that the precision per category is generally high. This means that we constructed a system that is precise in predicting the correct category (*public events* excluded), but also often misses the correct category due to the dominance of such *public events*.

We were able to compare these precision-scores to the precision per category for the random event set, which was the representation of the full event set. The results were indeed similar. The precision per class for *sport, politics, broadcast* and *software* were all high, while *other* received a lower precision score in comparison. This suggests that the latter categories are less distinguishable in general. For *other* we expected a low F1-score, since this is a set of events without a clear general theme. The low precision of *advertisement* in the evaluation set might be (partially) attributed to sparse data. It might be the case that there were simply not enough training instances for the classifier to learn from. We described in the Method section that the most indicative words of *advertisements* already showed signs of over-fitting on the training data. It is likely that the classifier learned the properties of specific *advertisement* events, instead of identifying a general theme that is present across all *advertisement* events, thus resulting in low precision. Unfortunately, we were not able to calculate the precision of *advertisement* in the random event set, since the we removed that category due to its low number of labeled instances.

For *special day* and *social action* the precision is different. We obtained a high precision for these categories in the evaluation set, while we obtained a lower precision for the random event set. The difference between the evaluation set and the random event set is that the set of best events was also used in the training phase of the evaluation set. This suggests that the addition of the best event set especially increased performance for *special day* and *social action*. This is probably simply an effect of using more training data. Both *social action* (25 versus 12) and *special day* (92 versus 23) were categories for which the best event set contained considerably more instances than the random event set. Therefore, we believe that the high precision scores for these categories are reliable, even though they were considerably lower in the random event set. A more detailed analysis on *social actions* is described in Section 5.2.

It has to be noted that it is sometimes difficult to differentiate between two categories, even for human annotators. For example, some events about Valentine's Day mainly consist of tweets that promote their special Valentine's Day actions. One could reason that the category should therefore be *advertisement*. However, it is not unreasonable to judge the main theme of the event and annotate it as *special day*.

Whether this high precision and low recall is good news for practical applications in general depends on the nature of such applications. Say we want to attend the next soccer match in our neighbourhood and ask the event detection system to provide us with a list of possible events. We would still be able to attend a game if the classifier provides us with nine irrelevant events, but also with an event that is indeed a nearby

soccer match. However, if the classifier does not output anything at all, we would never be able to attend the match. For these types of applications, we might prefer a high recall to a high precision.

However, there are other applications where a high precision is preferable. For example, we might want to construct a list of all sports events taking place all over the country. The only way that this list will be useful, is if the listed events are actually sports events that will actually take place. If such a list is full of irrelevant events, people will never use such an application to check whether they might have missed an interesting sport event taking place. In general, we prefer a high precision for applications that list an overview of events taking place, while we prefer a high recall for applications that aim to predict specific events.

5.2 Comparison with Ritter et al. (2012)

The study by Ritter et al. (2012) is the only one that produced an extensive evaluation of categorized **events**. Other studies that identify events are not interested in categorizing them (e.g. Petrović et al., 2010), while studies that categorize Twitter data usually do not extract events first (e.g. Zhao et al., 2011). This is why we will only compare our results on the general classification task to Ritter et al. (2012).

Ritter et al. (2012) evaluated the performance of their event categorization system on a set of 500 events with the highest association. This is very similar to our set of 425 events that obtained the highest eventscore. They ultimately obtained a precision, recall and F1-score of respectively 0.85, 0.55 and 0.67. Our system offered a comparable performance: a precision, recall and F1-score of 0.67, 0.67 and 0.65. Our supervised approach even outperformed their baseline supervised approach (which obtained a final F1-score of 0.59) by a 9% margin. However, Ritter et al. (2012) do not specify whether they use macro, micro or weighted precision, recall and F1-score. This is crucial information, since it might result in a considerable difference in our case, shown in Table 5.1. The micro average is similar, but if we would have used the macro average, our F1-score would drop by 10.3%. Ritter et al. (2012) also do not provide any information about the division of categories in the evaluation set of 500 instances.[1]

The division of categories and what type of F1-score they used are two very important factors in evaluating their performance. For example, it could very well be the case that one category is very dissimilar to the other categories in their event set. This would then result in the fact that a large sample of the 500 events with the highest association

[1] In personal communication, we asked Alan Ritter about this division. Unfortunately, he was unable to recover the document with the specific division of categories.

TABLE 5.1: The micro, macro and weighted precision, recall and F1-score for the down-sampled combined event set.

	Precision	**Recall**	**F1-score**
Micro	0.69	0.69	0.69
Macro	0.65	0.59	0.61
Weighted	0.69	0.69	0.68

actually belongs to a single category. This would result in a high F1-score when using the micro metric, even when classifying all events as the dominant category. However, the results would be a lot worse for the macro metric. If there is one very dominant category, that means it is likely that a lot of the other categories would receive an F1-score of 0.0. Since every category is equally important for the macro metric, these scores of 0.0 will greatly hurt the average F1-score.

On the other hand, if the 37-classes are fairly equally distributed in the set of 500 events, this means that the F1-score of 0.67 obtained by Ritter et al. (2012) is extraordinarily high. There are two reasons why it is unlikely that this is the case. For one, the division of categories is not equally distributed in their full event set (e.g. *sport* is 18 times as frequent as *medical*), which means that this it is very probable this is also the case in their set of best events. Secondly, it is improbable that a supervised classifier obtains an F1-score of 0.59 when it needs to distinguish between 37 different and sometimes very similar classes, while only having 500 instances to learn from. This raises the suspicion that there was one (or were a few) categories that were dominant in their evaluation set, which resulted in a high F1-score when using the micro metric. Note that this does not necessarily mean that their task was considerably easier than ours, since we only used nine different categories, one of which was dominant as well.

Unfortunately, Ritter et al. (2012) do not evaluate the performance of categorizing their full event set. It is questionable whether the results of their best event set will extend to their full event set. For one, we showed that classifying the best scoring events is easier than classifying a random subset of events. Secondly, differentiating between 37 categories is difficult enough for a classifier, but especially when we recall that they used very similar categories (e.g. *performance* and *concert*; *TV* and *film*). We described earlier that it is unlikely that their set of 500 best events had an equal distribution of categories. Thirdly, reliably classifying the full set of events was very improbable for them to begin with, since their unsupervised classifier only assigned a specific category to 46.5% of all events. The large number of categories combined with the fact that more than half of the events are classified as *other* leads us to believe that our supervised

approach outperformed their unsupervised approach on categorizing all events.

It is thus difficult to evaluate whether Ritter et al. (2012) outperformed us on either categorizing the best events or categorizing the full event set. All in all, we believe that we offer comparable performance on categorizing the best events, while we outperform their categorization system when categorizing all events.

5.3 Predicting social action events

An analysis of the categorization of *social action events* revealed that our system is very precise in predicting these events, but that the recall of these events is low. *Social action* obtained a precision of 0.74 on the combined event set, but only received a recall of 0.38. This means that 62% of *social action events* still go undetected. Testing on the evaluation set revealed even worse performance for *social actions*. The precision is slightly higher (0.80), but the recall is estimated to be much lower. Recall that we expected 3.3% of events to be a *social action*, while only classifying 0.2% of the events as such. We will compare the performance of our approach against the existing approaches that tried to predict *social action events* in the next paragraphs.

The first approach by Compton et al. (2013) was limited, only identifying events that contained predefined keywords and locations. Even with such limited range, they only obtained a precision of 0.55. Due to the mentioned limitations, their system is never able to detect events with newly emerging keywords or events that take place at novel locations. For one, this means that the recall will be low. Moreover, this means that they did not create a general system that is able to **forecast** *social actions*. It is only able to predict *social actions* in hindsight, after we already know which specific keywords and locations are indicators of civil unrest.

The most notable approach in this research field is probably the EMBERS system by Ramakrishnan et al. (2014). This is a deployed system that tries to forecast *social actions*, even issuing a warning alert when it believes a *social action event* is going to take place. They evaluated their system over a period of a month, resulting in a precision and recall of respectively 0.69 and 0.82. This means that we obtained a higher precision, but that their recall is higher than our estimated recall.

It is interesting to note how they received this recall score. They obtained a gold standard set of *social action events* by an independent organisation that had human analysts survey newspapers and other media for mentions of civil unrest. This means that they have a reliable way of calculating recall **in the real world**. On the other hand, our

approach is only able to recall events that were present in the set of Kunneman and van den Bosch (2015). We have no way of knowing whether they detected all *social action events* that actually happened in the real world. This means that we should consider the possibility that we might still miss *social action events* that were never detected as events in the first place, even further lowering our estimated recall. This as opposed to Ramakrishnan et al. (2014), who recalled 82% of *social action events* that actually happened in the real world. They ultimately obtained an F1-score of 0.75[2], which is considerably higher than our F1-score of 0.50 for *social actions* on the combined event set.

However, this is probably an unfair comparison. Ramakrishnan et al. (2014) use a set of multiple models that are all based on a rich set of features, including sources such as news media, Facebook and even currency exchange rates, while we only use information from Twitter. Especially the first two sources ensure that they will be accurate in predicting *planned* events, since those events will be talked about often in the media. Luckily, Muthiah et al. (2015) reported the results of EMBERS when just taking Twitter data into account. They report a high precision of 0.97, but only a recall of 0.15. This is similar to our approach, which also obtained a high precision (0.80) and a low recall. This means that our approach has a lot to gain from other data sources, but that it did satisfactory use the information available.

5.4 Increasing the recall for social action events

We showed that we are able to greatly improve the recall of *social action events* by looking at events that received *social action* as the second most probable classification. We were able to increase the recall by 232%, while only decreasing precision by 14%. However, a possible problem with the annotation of these events is that the annotator might have been biased. We knew in advance that these events were probable to be *social actions*, instead of a random sample of events. This is why we calculated the most indicative words for the set of events, based on their *tf * idf* value. We do this for the set of 226 *secondary social action* events as well as for the subset of those events that actually received *social action* as an annotation. The most indicative words are shown in Table 5.2.

For both sets of events, we clearly see a large number of words that refer to *social actions*. Especially *staking (strike)*, *protest (protest)*, *demonstratie (demonstration)* and *manifestatie (manifestation)* are very clear indicators of *social actions*. It is also interesting to see a number of specific entities that are often related to *social actions*, such as *Malieveld*, *GVB*, *HTM* and *Akvakabo*. There are virtually no words there that are

[2]This is not actually reported in their paper, but deduced from their precision and recall scores.

TABLE 5.2: The most indicative words for **classified** and **annotated** secondary social actions.

Most indicative words of classified secondary social action events
haag staking den demonstratie fnv tegen actie ad6
malieveld alpe protest manifestatie juni htm staken
denhaag amsterdam gvb ov abvakabo bezuinigingen
dhuez staakt trams rijden bussen geen demonstreren
Most indicative words of annotated secondary social action events
staking haag demonstratie tegen den protest fnv
manifestatie malieveld actie staken gvb htm ov
bezuinigingen staakt amsterdam denhaag trams
bussen ret metros abvakabo demonstreren rijden

not easily linked to *social actions* in general. We therefore conclude that the annotators were unlikely to be biased in the annotation process.

The most indicative words of the *secondary social action events* in general are very similar to the most indicative words of the annotated *social action events* in the combined event set, shown in Section 3.4. For this reason, we believe that the low recall of *social action* might simply be an effect of sparse data. *Social action* initially seemed similar to *sports, politics* and *software* in the sense that there was a clear theme present across all their events. However, those categories all obtained a decent estimated recall between 0.33 and 0.61. The difference here is that these categories had 58, 65 and 71 training instances, while *social action* only had 37. The high precision of *secondary social actions* suggests that the connections between the indicators of *social action* (e.g. protest, demonstration, strike) were simply not strong enough yet to overturn the classification of the dominant *public event* category. For this reason, we believe that using more training instances might greatly improve the classification of *social actions*.

Using the ranking of the Bayesian probabilities thus helped to increase the recall. However, we did not use the actual probabilities to influence the classification process, but only the ranking of these probabilities. An interesting direction for future research therefore is to use the obtained Bayesian probabilities in a more sophisticated manner. For example, it is possible to employ a certain probability threshold for *social action* and simply classify events that exceed this threshold as *social action*, regardless of the probability of other categories.

We investigated whether there is a correlation between the probability of *social action* for each event and whether the event was ultimately annotated as a *social action*. We determined this for the annotated set of *secondary social action events*. First, we use the probability as a threshold. We only maintain the events that exceed a certain probability threshold and calculate the precision for that subset of events. This means that for a threshold of 0.01, all events are included, while there are no events left for a threshold of 0.40. To be able to draw general conclusions, we only test on thresholds that contain five or more events. The results are shown in Figure 5.1. We see a slight correlation, but it is not clear whether this correlation is significant. To test this, we performed a least-squares logistic regression test. We found a significant correlation between the probability and whether or not it was a *social action event*, $r(198) = 0.20$, $p = 0.005$. This means that *secondary social action* events are indeed more likely to be an actual *social action event* if they have a higher probability.

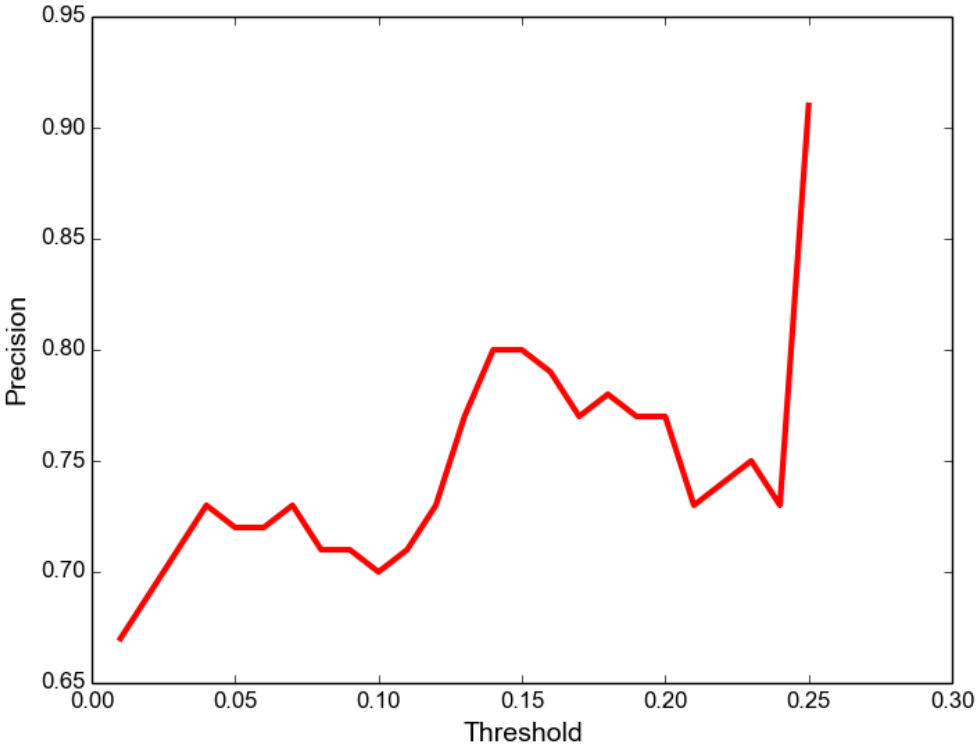

FIGURE 5.1: Precision for *social action events* per probability threshold.

The actual implementation of such a method requires an extensive search for the best threshold setting, since a lower threshold will result in higher recall and lower precision, while a higher threshold will have the opposite effect. The main advantage of this approach is that we are thus able to specify a specific precision-recall trade-off. The main disadvantage of this approach is the fact that the resulting probabilities of the Bayes classifier are not always accurate, since we violated the independence of features assumption.

A similar option is to investigate the **difference in probability** between the probability of *social action* and the probability of the most probable category of the event. It stands to reason that a small absolute difference in probability between those two categories will result in an event that is more probable to be a *social action*, while a larger difference suggests that an event is in fact not a *social action*. Again, we performed a least-squares logistic regression test and found a significant correlation between difference in probability and whether or not the event was annotated as a *social action*, $r(198) = -0.16$, $p = 0.022$.

Another option is to train a binary classifier specifically for this problem. It only has to distinguish between *social actions* and *non-social actions* for the set of *secondary social action* events. The different probabilities can now be used as features, since we know they have a significant correlation with whether or not an event is a *social action*. This might greatly increase the reduced precision, while still resulting in a high recall. A number of other interesting directions for future research regarding the prediction of *social actions* are outlined in the next section.

5.4.1 Future research for detecting social actions

A possible application of detecting *social events* is to notify the authorities when an (unexpected) *social action* will take place. Since there is a lot at stake here, we rather show a large list of events that might be a *social action event* that actually does contain all such events, than a system that often ignores large *social action events*. Since we are not talking about thousands of events daily, an analyst can manually annotate the set of possible *social action events* and only inform the authorities when a serious demonstration or protest will take place. False alarms are then easily ignored, but overlooking an actual *social action event* might have serious consequences. We thus prefer a high recall to a high precision, meaning that our approach is not very suitable. In this section we propose a number of ways to adjust the system to increase the performance of predicting *social action events*.

If our main goal is to increase the performance on predicting *social action events*, the best direction for future research if to exploit more data sources. Especially news media articles and specific Facebook events are valuable sources of information that are not utilized now. This is the main reason why we were outperformed by the EMBERS system of Ramakrishnan et al. (2014). However, this will only work if these data sources are already utilized in the general event detection system. The aim is to detect **all** *social action events*, not only the events that received a lot of attention on Twitter.

A different direction for future research is to transform the whole problem into a binary problem. An event is either a *social action* or not a *social action*. This method is applied by all other approaches that try to detect *social action events*. This way, it only has to be distinguishable from the average of all categories, instead of distinguishable from all categories independently. We might then even be able to differentiate between possibly violent *social actions* and harmless *social actions*. Such an approach will most likely result in a reduced precision, but also in a higher recall, which is preferable when detecting *social actions*. The main disadvantage of this approach is that it will annihilate all other practical applications of the event categorization system that have no interest in detecting *social actions*. However, we can simply implement this as a specialized extension of the main categorization system, ensuring that the categorization of all events will still take place.

Finally, it is interesting to apply unsupervised learning on the full set of events. The main disadvantage of supervised learning is that it is dependent on a small sample of annotated instances in the learning phase. Unsupervised learning does not have this disadvantage, since it is able to use all available data in the learning phase. This way, we will be able to learn from (the average of) all *social action events* in the event set, instead of just the few events that were presented in the set we annotated. We are then able to transform this information to different features in the supervised approach, which might greatly improve classification. We are then able to perform supervised learning while still using insights obtained from unsupervised learning, hopefully resulting in the best of both worlds.

Chapter 6

Conclusion

In this thesis, we automatically categorized Dutch Twitter events. We performed a supervised learning approach on three sets of events and evaluated the performance of the classifier by testing on a set of events that was never used in the training phase. We found that:

(1) It is easier to classify events with a high eventscore, but there is no effect of eventscore in general. We obtained a 10.8% increase in F1-score for the set of best events when compared to a set of randomly selected events. Our results on the best event set are comparable with the results of Ritter et al. (2012). However, it is difficult to compare the results, since the division of categories is unclear in their set of 500 best events. We found no significant effect of eventscore in general. This means that approaches that use the eventscore to exclude the unclear events in advance are not likely to considerably improve their F1-score.

(2) More training data resulted in better performance of the classifier, while sophisticated features improved performance only slightly.

The best performance was obtained when combining the best and random event set, resulting in a 14.7% increase in F1-score in comparison with the random events and a 4.4% increase in comparison with the best events. This means that using more annotated data would probably lead to an improvement in future work. The addition of sophisticated (DBpedia, polarity) features barely improved classification. We suspect that this is the case since most information is already captured by the words themselves. For this reason, we believe that the addition of more sophisticated features might only increase performance if they describe a concept that is not easily captured by the words themselves. In general, this suggests that the biggest improvements will be obtained by increasing the number of training instances and not by the addition of more sophisticated features.

(3) We are not able to reliably predict the full set of events due to the low recall of non-dominant categories.

The events in the full event set obtained *public event* as a classification 82-93% of the time, even though *public event* was predicted to only make up 44% of all events, resulting in an estimated 38% of the events that are classified incorrectly. This comes at the expense of the recall of the other categories, resulting in the fact that their recall will be low, or at least lower than is indicated in the results of the combined event set. However, it must be noted that the full event set contained approximately 37% non-events. These events have no category and should therefore not be viewed as incorrectly categorized. It is likely that the dominance of *public event* is simply due to these unclear non-events being categorized as the dominant category.

Although they do not evaluate it directly, we believe that the approach of Ritter et al. (2012) also did not perform well on their full event set. For one, they dismiss 53.5% of the events in advance that do not have a clear category. Secondly, they use a large number of categories, combined with the fact that these categories are often similar. This will make classification difficult and also makes it probable that their F1-score on the set of the 500 best events can not be extended to their full event set.

(4) The precision of the non-dominant categories is high.

When the classifier did predict a different category than *public event*, it generally did so correctly. The balanced evaluation set that was never used in the training phase revealed that the precision of the non-dominant categories was high, except for the categories *advertisement* and *other*, which was expected. This means that we are still able to make reliable predictions about the category of an event if we simply ignore events that are classified as *public events*. However, due to the low recall, this means that we often overlook important events as well.

(5) If we predict a *social action*, we generally do so correctly. However, we often overlook instances of *social actions* as well.

Our previously described high precision and low recall for the non-dominant categories is also the case for *social actions*. Evaluation on 93 events that were classified as *social action* revealed a precision of 0.80, while the recall was estimated to be 0.04. A method that utilized the Bayesian probabilities by selecting *secondary social actions* was able to increase this recall by 232%, while only decreasing precision by 14%. This method might also increase performance on other minority categories and is therefore an interesting direction for future research.

The precision of predicting *social action events* is promising , since we are confident that a *social action* event is indeed taking place if we predict one. However, the big downside is evident: we will also often miss events of this type. The most notable approach in this field by Ramakrishnan et al. (2014) did not have this problem, it obtained a reliable recall of 0.82, although it received a slightly lower precision than our approach. It used a rich set of features, ranging from (social) media sites to economic indicators. It turned out that our approach mainly got outperformed due to limiting our data sources to Twitter, since a comparison with the Twitter-only approach in Muthiah et al. (2015) revealed similar results.

Our low recall might simply be a result of sparse data. There were only 37 annotated instances of a *social action event*, which makes generalization difficult for the classifier. However, we believe that this category is not a special category that simply cannot be distinguished from other categories. *Social action* has a number of clear identifiers (e.g. protest, demonstration, strike) that are present in virtually all *social action events*. This is similar to the categories that did obtain a decent recall (e.g. *sports, politics*), the only difference being that those categories were more frequent in the training set. Therefore, we believe that using more training instances will especially improve results of the minority categories, including *social action*.

All in all, we constructed a system that is precise in its predictions, but does not offer those predictions often. This means that it will only be applicable to tasks that require extreme precision in the output and are not so interested in detecting all possible events. We believe that for most tasks it is preferred to only show the events we are sure about, instead of a list of events that contains more actual events, but also a lot of incorrectly categorized events. For *social actions*, this might be the other way around unfortunately. False alarms are easily ignored, while overlooking a large demonstration or protest might have serious consequences. In general, we believe that categorization systems might benefit most from increasing the number of training instances, while adding sophisticated features might not considerably increase performance. The automatic filtering of non-events might also increase the performance, since it is difficult to assign a category to something that is not an actual event. For detecting *social actions* specifically, we believe that the most promising direction of future research is exploiting data sources other than Twitter. These approaches should focus on increasing the recall of *social action events*, since we showed that only using Twitter data already resulted in a high precision.

Appendix A

Additional Figures and Tables

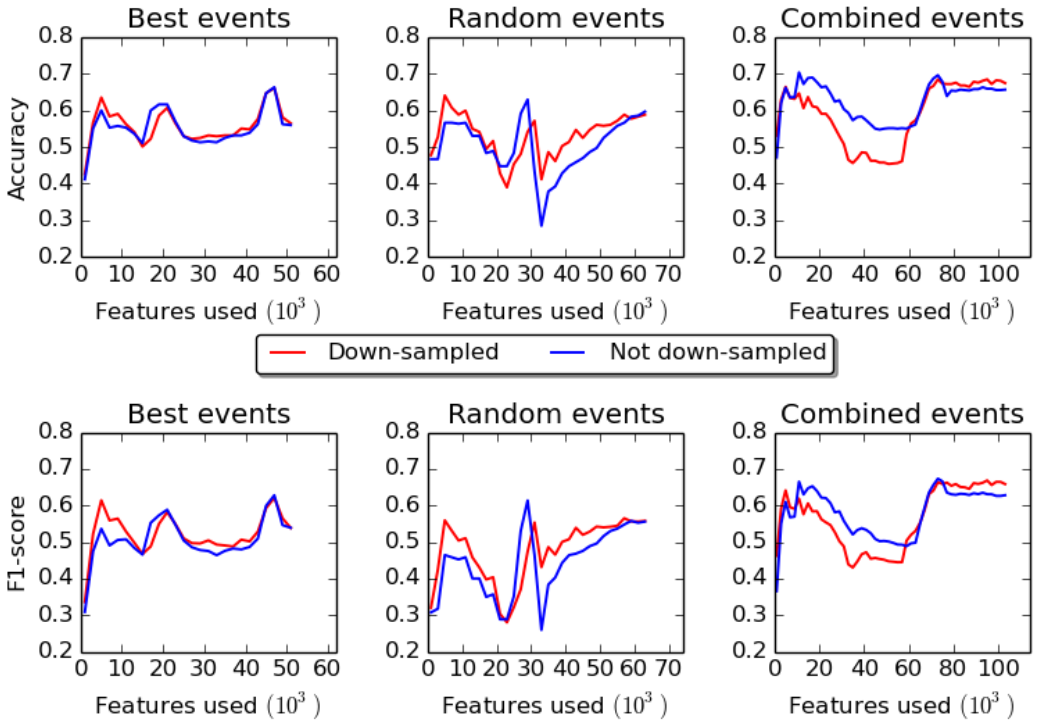

FIGURE A.1: The effect of down-sampling when using chi-squared feature-selection.

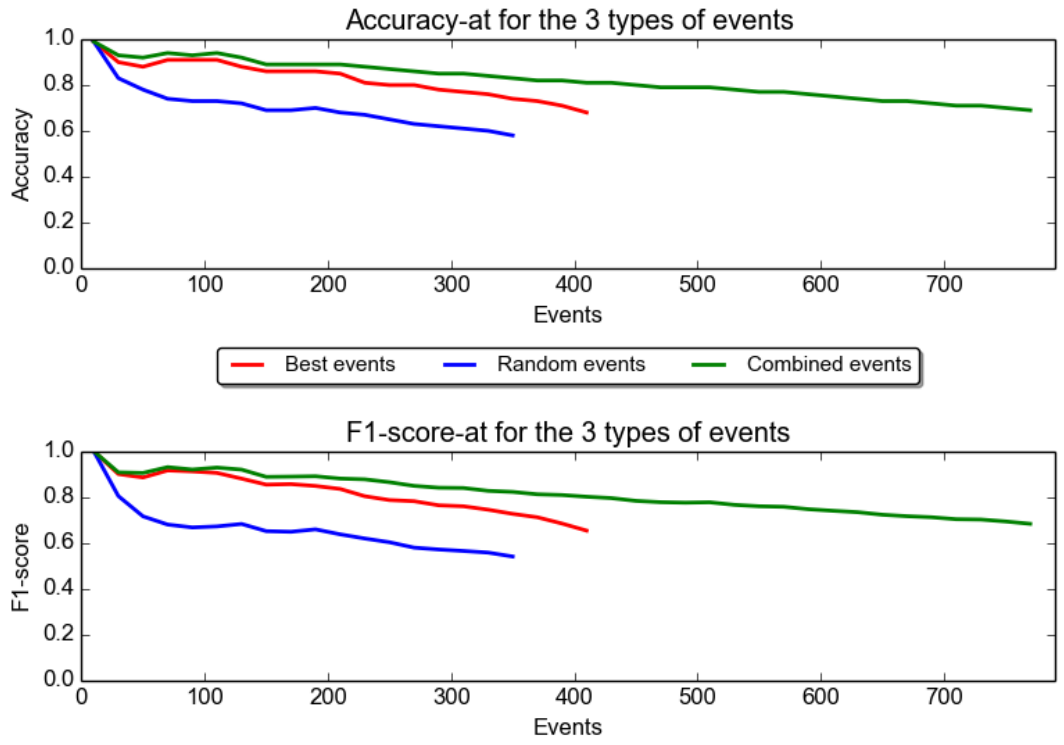

FIGURE A.2: Comparison of the accuracy-at and F1-score-at scores for the different types of data when testing on subsets of events with the highest classification probability.

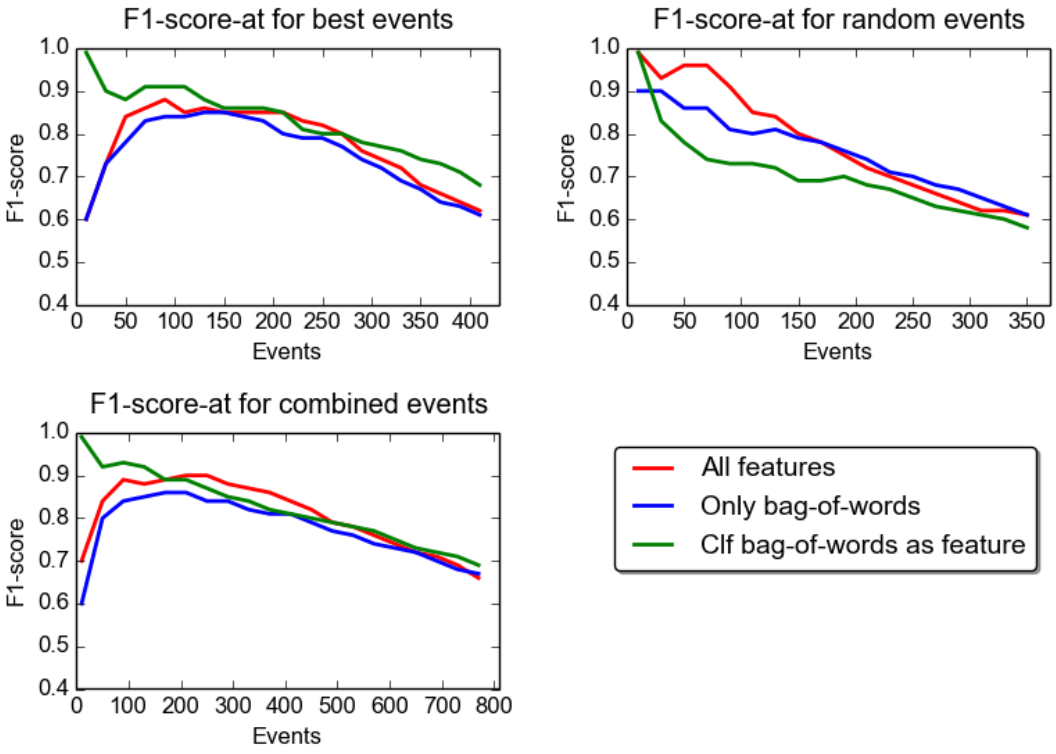

FIGURE A.3: Comparison of the different F1-score-at scores for the 3 different types of feature-sets when testing on subsets of events with the highest classification probability.

TABLE A.1: The English translation of the example of a Twitter event.

Date	2014-06-23
Keywords	#23june
Tweet 1	then I just view it as some nice music for on my birthday #23june
Tweet 2	almost my birthday bitchez :) #23june
Tweet 3	the next match is going to be difficult.. #nedchi #orange #23june #wc2014
Tweet 4	watching orange soon #nedchi #23june
Tweet 5	the first two hours of my workday have passed #23june #graduating #pabo4

TABLE A.2: A few examples of calculating overlap between tweets.

Tweet X	
Vanavond naar Ronald Goedemondt, wordt leuk!	
Tweet Y	**Overlap**
Naar Ronald Goedemondt vanavond, wordt leuk!	1.00
Ronald Goedemondt wordt vast heel leuk vanavond	0.71
Eindelijk naar nieuwe show van Goedemondt	0.33
Zin in de nieuwe show van Ronald Goedemondt vanavond	0.33
Vanavond wordt leuk #Goedemondt #zinin	0.50
Benieuwd naar Goedemondt vanavond #tamboerijn #gdansk	0.50

TABLE A.3: The full rule-system for determining whether the extra tweets are relevant.

Observation	**Relevant set?**
All 7 tweets are relevant	Yes
Doubt about 1 tweet, other tweets relevant	Yes
Doubt about two or more tweets	No
1 tweet is irrelevant, other tweets relevant	No
More than 1 tweet is irrelevant	No

TABLE A.4: The optimal SVM settings for each event set and type of down-sampling.

		Down sampled	**Kernel**	**Degree**	**C**	**Gamma**
Best events	All features	Yes	linear	NA	50	0.016
		No	linear	NA	50	0.128
	Only bag-of-words	Yes	poly	1	50	0.032
		No	poly	1	2.5	1.024
	Classification bag of words added as feature	Yes	poly	1	25	0.064
		No	poly	1	10	1.024
Random events	All features	Yes	poly	1	75	0.001
		No	linear	NA	0.5	0.256
	Only bag-of-words	Yes	linear	NA	5	0.128
		No	linear	NA	1000	0.128
	Classification bag of words added as feature	Yes	linear	NA	5	0.128
		No	linear	NA	1000	0.128
Combined events	All features	Yes	linear	NA	15	0.256
		No	linear	NA	50	0.032
	Only bag-of-words	Yes	linear	NA	15	0.128
		No	linear	NA	25	0.256
	Classification bag of words added as feature	Yes	linear	NA	50	0.064
		No	linear	NA	5	0.128

TABLE A.5: The classifications of the 92,701 events when trained on the different event sets.

	Algorithm trained on		
Classification	**Best events**	**Random events**	**Combined events**
Sport	3,257	1,632	2,909
Politics	2,439	1,156	2,424
Broadcast	151	235	228
Public event	76,778	86,176	80,290
Software	2,654	646	1,706
Special day	4,290	61	1,972
Social action	177	28	105
Advertisement	91	NA	27
Other	1,561	1,464	1,739

Bibliography

Aggarwal, C. C. and Subbian, K. (2012). Event detection in social streams. In *SDM*, volume 12, pages 624–635. SIAM.

Allan, J., Lavrenko, V., Malin, D., and Swan, R. (2000). Detections, bounds, and timelines: Umass and tdt-3. In *Proceedings of topic detection and tracking workshop*, pages 167–174.

AlSumait, L., Barbará, D., and Domeniconi, C. (2008). On-line lda: Adaptive topic models for mining text streams with applications to topic detection and tracking. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 3–12. IEEE.

Becker, H., Naaman, M., and Gravano, L. (2011). Beyond trending topics: Real-world event identification on twitter.

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., and Hellmann, S. (2009). Dbpedia-a crystallization point for the web of data. *Web Semantics: science, services and agents on the world wide web*, 7(3):154–165.

Bunescu, R. C. and Pasca, M. (2006). Using encyclopedic knowledge for named entity disambiguation. In *EACL*, volume 6, pages 9–16.

Compton, R., Lee, C.-K., Lu, T.-C., de Silva, L., and Macy, M. (2013). Detecting future social unrest in unprocessed twitter data:"emerging phenomena and big data". In *Intelligence and Security Informatics (ISI), 2013 IEEE International Conference On*, pages 56–60. IEEE.

De Smedt, T. and Daelemans, W. (2012a). " vreselijk mooi!"(terribly beautiful): A subjectivity lexicon for dutch adjectives. In *LREC*, pages 3568–3572.

De Smedt, T. and Daelemans, W. (2012b). Pattern for python. *The Journal of Machine Learning Research*, 13(1):2063–2067.

González-Bailón, S., Borge-Holthoefer, J., Rivero, A., and Moreno, Y. (2011). The dynamics of protest recruitment through an online network. *Scientific reports*, 1.

Hayes, A. F. and Krippendorff, K. (2007). Answering the call for a standard reliability measure for coding data. *Communication methods and measures*, 1(1):77–89.

Hong, L. and Davison, B. D. (2010). Empirical study of topic modeling in twitter. In *Proceedings of the first workshop on social media analytics*, pages 80–88. ACM.

Japkowicz, N. and Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449.

Kallus, N. (2014). Predicting crowd behavior with big public data. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, pages 625–630. International World Wide Web Conferences Steering Committee.

Kunneman, F. and van den Bosch, A. (2015). Automatically identifying periodic social events from twitter. *RECENT ADVANCES IN*, page 320.

Leetaru, K. (2011). Culturomics 2.0: Forecasting large-scale human behavior using global news media tone in time and space. *First Monday*, 16(9).

Lewis, D. D. and Catlett, J. (1994). Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the eleventh international conference on machine learning*, pages 148–156.

McCallum, A., Nigam, K., et al. (1998). A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer.

McMinn, A. J., Moshfeghi, Y., and Jose, J. M. (2013). Building a large-scale corpus for evaluating event detection on twitter. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 409–418. ACM.

Mihalcea, R. and Csomai, A. (2007). Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242. ACM.

Morozov, E. (2009). Iran: Downside to the" twitter revolution". *Dissent*, 56(4):10–14.

Mungiu-Pippidi, A. and Munteanu, I. (2009). Moldova's" twitter revolution". *Journal of Democracy*, 20(3):136–142.

Muthiah, S., Huang, B., Arredondo, J., Mares, D., Getoor, L., Katz, G., and Ramakrishnan, N. (2015). Planned protest modeling in news and social media. In *AAAI*, pages 3920–3927.

Papadimitriou, C. H., Tamaki, H., Raghavan, P., and Vempala, S. (1998). Latent semantic indexing: A probabilistic analysis. In *Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 159–168. ACM.

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

Petrović, S., Osborne, M., and Lavrenko, V. (2010). Streaming first story detection with application to twitter. In *Human Language Technologies: The 2010 Annual*

Conference of the North American Chapter of the Association for Computational Linguistics, pages 181–189. Association for Computational Linguistics.

Ramage, D., Dumais, S. T., and Liebling, D. J. (2010). Characterizing microblogs with topic models. *ICWSM*, 10:1–1.

Ramakrishnan, N., Butler, P., Muthiah, S., Self, N., Khandpur, R., Saraf, P., Wang, W., Cadena, J., Vullikanti, A., Korkmaz, G., et al. (2014). 'beating the news' with embers: Forecasting civil unrest using open source indicators. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1799–1808. ACM.

Rish, I. (2001). An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46. IBM New York.

Ritter, A., Etzioni, O., Clark, S., et al. (2012). Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1104–1112. ACM.

Sang, E. T. K. and van den Bosch, A. (2013). Dealing with big data: The case of twitter. *Computational Linguistics in the Netherlands Journal*, 3(121-134):2013.

Shriberg, E., Stolcke, A., Hakkani-Tür, D., and Tür, G. (2000). Prosody-based automatic segmentation of speech into sentences and topics. *Speech communication*, 32(1):127–154.

Vapnik, V. (2013). *The nature of statistical learning theory*. Springer Science & Business Media.

Wallach, H. M. (2006). Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd international conference on Machine learning*, pages 977–984. ACM.

Webb, G. I., Boughton, J. R., and Wang, Z. (2005). Not so naive bayes: aggregating one-dependence estimators. *Machine learning*, 58(1):5–24.

Weng, J. and Lee, B.-S. (2011). Event detection in twitter. *ICWSM*, 11:401–408.

Yang, Y., Pierce, T., and Carbonell, J. (1998). A study of retrospective and on-line event detection. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 28–36. ACM.

Zaidi, N. A., Cerquides, J., Carman, M. J., and Webb, G. I. (2013). Alleviating naive bayes attribute independence assumption by attribute weighting. *The Journal of Machine Learning Research*, 14(1):1947–1988.

Zhang, W., Su, J., Tan, C. L., and Wang, W. T. (2010). Entity linking leveraging: automatically generated annotation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1290–1298. Association for Computational Linguistics.

Zhao, W. X., Jiang, J., Weng, J., He, J., Lim, E.-P., Yan, H., and Li, X. (2011). Comparing twitter and traditional media using topic models. In *Advances in Information Retrieval*, pages 338–349. Springer.