rijksuniversiteit
groningen

faculteit der letteren

Detecting news event commentary on Twitter

A bimodular approach to real-time data classification

Tim Kreutz

Bachelor Thesis Information Science

University of Groningen

June 2015

Acknowledgements

There is something strange in combining the good with the bad. It was my intention, when starting the day in day out toil of this project, to lighten its load by carefully composing a music playlist. This way, whenever I started working I would hear the cheerful tunes of Paul Simon, the energizing grit of the Arctic Monkeys or the white noise that Bob Dylan has recently become, and I would work harder and longer and have more fun.

Soon, the bad overcame the good. Favorite songs turned into haunting nightmares and became symbol for ill-fitting vectors, NaN's and being unable to understand a single sentence after reading it continually for minutes on end. It took only a few weeks to ruin half my music library.

There are also the songs, however, that you do not grow tired of. I remember last year when I spent one week, seven hours per day reading feminist literature and annotating Reddit posts on pornography. I should not forget to explain I was writing my Sociology bachelors thesis at the time. For some reason the Foo Fighters caught the essence of these strange days and nothing could set my mood better each morning than 'Stranger Things Have Happened'.

This year it was the Beatles' 'Something' that did the trick. I do not know if it the steady rythm in its verses or the crying out in the bridge, but it made work easier.

Other than with songs, something even stranger happened. Throughout the proces of doing this research I do not think I was ever particularly happy about it. Looking back, however, it has been one of the most interesting and rewarding projects I ever undertook.

I would like to thank my parents for supporting me during my bachelor degree, the 'Alfa-informatica' department for consistently leaving their doors open and offering advice and the friends that I sought out to distract me for not getting annoyed with me.

Abstract

Access to vast amounts of untructured digital data can shed new lights on existing problems. Combining traditional news media with social media can lead to more rapid news updates. Looking at Twitter specifically, tweets about news events reflect the public opinion. Techniques are reviewed for detecting relevant commentary on Twitter for a given news article. To deal with the size of the data, a bimodular approach is proposed. The first module selects tweet candidates based on text similarity between a news article and the tweet. The second module applies a machine learning algorithm to the candidates to determine their relevancy. Four machine learning models are evaluated and the Random Forest model by far outperforms the others. Combining the techniques in both modules proves to be a powerful way to deal with real-time data classification. This approach is demonstrated on the NieuwsTwiets website which selects relevant tweets for news articles in real-time.

Contents

Chapter 1

Introduction

The internet is full of text documents. They take many forms, from encyclopedia articles to blogposts; from restaurant reviews to user manuals. Unlike in databases, where the format of data is uniform and the relationship between datafields is usually explicitly stated, the article formats are very diverse and it is unclear how the articles relate to each other. We refer to these documents as unstructured textual data.

There are a lot of uses for the vast amounts of unstructured textual data available to us today. Using the right data, in the right way, can provide insights into the way people think and behave. Browser histories and website statistics, for example, can be analyzed to provide insight into browsing behavior; Digital medical records can be used to analyze physicians performances.

The challenge in using unstructured data lays in organizing it in a way that makes it useful and usable. Only when funneled through a system that is successfully calibrated to user information needs, the inherent noise of unstructured data is unproblematic and the advantages of huge, constant information sources emerges.

One sector in which frequent information supplies are prudent is the media; specifically news media. Nowadays, news events are reported through an increasingly wide range of channels, including news websites and social media. This development only adds to the necessity for news updates to be frequent and timely. Users have come to expect to be informed constantly and within minutes of news development.

Besides the velocity of current media outlets, they have also become interactive. Popular news websites have added social features to their websites, allowing users to respond to the news articles. In most cases, it does not stop there. The news media are also engaged with users on social media websites. On Twitter, news is broadcast in a tweet consisting of 140 characters. Users can then favorite the tweet, share it or reply to it.

In a more general sense, Twitter shows comments, opinions and discussions on news topics.

It reflects the social impact of news events; making up a vast arena of discussion in which public opinion and values are constantly negotiated. Social interaction with news articles is one step, but combining traditional news media with Twitter in real-time is a far more interesting one.

The relevance of such a task lays in extending the traditional reporting of the news with real-time updates and subjective meta-information, not only from the (potentially) biased (Reurings, 2008) user-base of a single news website, but from the much larger audience present on Twitter.

This thesis takes articles from Dutch news websites as a starting point for detecting relevant tweets about the events and topics central to these articles in real-time. Using a large data source as Twitter for a real-time application can harm performance. It is also important to recognize this problem and to describe how to go about finding a solution. This thesis poses the following research questions:

- What makes up the relevance of a tweet for a news article?
- How do we cope with the size of Twitter as a dataset in real-time applications?
- Which classification model is best suited to judge tweet relevance?
- How can we implement our findings in a user-friendly format?

The main question this thesis aims to answer is: *Which techniques can be used to automatically detect real-time news event commentary on Twitter?* As part of the extension for the Honours College, it goes beyond describing these techniques, and applies them for the website Nieuwstwiets.nl.

Chapter 2

Related work

There is a lot of related work that uses Twitter as a main data source. These publications do not only show the usefulness of Twitter and which techniques are commonly applied, but can also serve as a caution to the problems we can encounter in using Twitter as a data source. There is also related work that has aimed to combine news articles with tweets. We will describe how this thesis relates to earlier works and how it expands or improves on what has already been done.

2.1 Twitter research

In computational linguistics, Twitter has been successfully tapped for sentiment analysis, topic- and event extraction. In *Extracting Events and Event Descriptions* (Popescu et al., 2011) encouraging progress is made in automatically detecting events on Twitter, identifying entities central to these events and extracting audience opinions. Popescu, Pennachiotti and Paranjpe define an event as an activity or action with a clear, finite duration in which an entity plays a key role (Popescu et al., 2011, p. 105). They use a list of known entities to detect aboutness in a set of tweets in a certain time period. The aboutness is a measure which indicates a likelihood of a certain entity to be the main entity in the set of tweets, and is provided by a system with a Machine Learning (ML) approach. The system considers information about the length, category and language of the set of tweets and the words used. Such a supervised ML approach makes consistent improvement over the base line system (a TF-IDF similarity rank). Besides the entities involved in events, it is also important to extract the actions that took place. For this, the authors implement an off-the-shelf Part of Speech (POS) tagger to analyze which part of the text denotes an action. Here, no advances ML approaches are used because the extractable information is scarce and impairing execution time is not worth it (Popescu et al., 2011).

Twitter as a corpus for research is regularly scrutinized. On the one hand, its size and its up-to-date nature mark great potential for analyzing current topics (Popescu et al., 2011) and everyday use of language. On the other hand, the limiting format of tweets, its relatively young user base and the presence of meaningless tweets from automated tweeting services leads to a stream of data that contains non-natural language patterns and is notoriously noisy.

Han and Baldwin (2011) specifically focus on string quality of short texts on Twitter and in SMS messages. They note that quality varies significantly and that, in the poor quality spectrum, strings are brimming with typos, abbreviations and emoticons. Such ungrammatical texts are problematic for text processing tools and could result in a blind spot for any system that textually analyzes Twitter. Such ungrammaticalities are more often found in personal updates than anything else (Han and Baldwin, 2011) thus less problematic for this thesis. Implementing a lexical normalization as suggested by Han and Baldwin is beyond the scope of this research. The main entities in both news articles and related tweets are usually specific names and are less likely to be misspelled or altered. There is however always a possibility for missing important tweets because of user errors.

2.2 Combining tweets and news articles

Similar to the goal of this thesis, Phelan et al. (2010) aim to combine data from news websites with the Twitter social stream. The content of a user's Twitter timeline and their favorited RSS channels are compared. Based on the co-occurence of terms in the tweets and articles in the RSS feed, users receive a list of recommended articles. The recommendation system is simple, only looking at text similarity between tweets and articles, yet effective: users have a 30 to 45 percent higher click-through rate when receiving article recommendation based on their Twitter timelines than when they are just presented with their favorite channels, suggesting that the recommendations are succesfully specifying user information needs.

There are also a lot of differences between the research at hand and the one by Phelan et al. (2010). In the latter, only the tweets that appear on specific users' timelines are used. The evaluation depends on user interaction, and does not unambiguously show the effectiveness of their approach.

In contrast, we aim to recommend tweets not based on user profiles, but based on news articles. Besides the recency of tweets, we therefore do not know which part of the Twitter data is relevant and should be analyzed.

This is very much like a commercial application introduced by Crowdynews (Crowdynews, 2015): the Article Engager. The Article Engager is a Twitter widget that automatically shows the relevant tweets to any given news article. Since it is a commercial product that is very client oriented, it also has filters for profanity and inappropriate content. For this section I reached out to Crowdynews, but received no reply as to the precise techniques that could inform my research. The website states, however, that *"the Twitter leveraging technology of Crowdynews is built on algorithms, based on computational linguistic and artificial intelligence"* (Crowdynews, 2015). Crowdynews is extremely effective at enriching news content and has applied the Article Engager to large newspaper websites like the Washington Times and the Chicage Tribune, but also non-English outlets like the Spanish Sport.es and the Turkish Fanatik.com.tr. They currently support ten languages.

The products that Crowdynews offers are powerful and widely applicable. It would be hard to compare the research results with theirs and even harder to improve them. There is value, however, in evaluating techniques that can be used to judge tweet relevancy. We do so openly, so that readers can extend on the findings of this research and improve its results.

2.3 Implications

Past research that uses Twitter as a main data source demonstrates its potential for improving systems and gaining valuable insights. It also serves to show the limitations of the social networking website. Basic techniques of text analysis are shown to be useful in rapid real-time applications (Phelan et al., 2010), whereas supervised ML approaches support better results (Popescu et al., 2011). Sometimes, specific parts of speech are relevant and should be extracted (Popescu et al., 2011). Data from Twitter can be gramatically incorrect which can prove to be a nuisance for text analysis, but ungrammaticalities mostly occur in personal updates (Han and Baldwin, 2011).

Another limitation which is inherent to using the Twitter corpus is its sheer size. In Phelan et al. (2010) the Twitter data that is used, is a miniscule portion of what is available. We have to look at making out own selection of the Twitter data so that a real-time application is possible. By openly sharing the methods and results, it should be easy to repeat this thesis research and extend or improve it.

Chapter 3

Methods

This section outlines the methods that were used for finding relevant tweets. It discusses the data that is used for the evaluation of our solution, as well as the way in which this data was processed. It further proposes a way to reduce the Twitter data so that it becomes more suitable for real-time detection of relevant tweets.

3.1 Data

We are working with two data sources. On the one hand we need news articles to analyze. A simple way to extract them was informed by Phelan et al. (2010) who use RSS feeds. On the other hand we need Twitter data to compare with the news articles.

3.1.1 News articles

The news articles are taken from RSS feeds of the three most visited news websites in the Netherlands: Nu.nl, NOS.nl and Telegraaf.nl (Vinex, 2013). At any time, this results in a list of 41 articles (20 from NOS.nl, 11 from Telegraaf.nl and 10 from Nu.nl). For every article, the following attributes are extracted and stored:

- a unique hash code
- a link to the full article
- the publication timestamp
- the full title
- the abstract

We do not use the full text from an article because the abstract is readily available in the RSS stream and contains the most important words and entities in the main article. Although untested, we argue that using the full article adds more context, and therefore noise, to the news article data. It should further be noted that NOS.nl includes a more extensive abstract in its RSS feed, whereas the abstracts from Nu.nl and Telegraaf.nl are shorter but do not vary much in length. Each article can be represented in format a:

$$a(hash) = (link, timestamp, title, abstract)$$

An example that we will keep using throughout this text is:

> a(ac4f7655fc3629cc2046) = (http://www.nu.nl/buitenland/4052890/oude-stad-palmyra-vrijwel-geheel-in-handen-van-is.html, Wed, 20 May 2015 23:53:20 +0200, 'Oude stad Palmyra vrijwel geheel in handen van IS', Terreurbeweging Islam(...))

3.1.2 Twitter

The Corpus Nederlandstalige Tweets is an extending data collection at the University of Groningen. The data is extracted from the ongoing Twitter data stream by a sophisticated language detection system which filters non-Dutch tweets and stores the remainder in a compressed JSON format every hour (Tjong Kim Sang and Bos, 2012). The corpus is available for staff and students at the Faculty of Arts. A wide range of attributes per tweet are stored, but we make a selection of relevant attributes quite similar to those for the news articles:

- a unique tweet ID
- the publication timestamp
- the user
- the tweet (text)

Each tweet can be represented as t:

$$t(id) = (timestamp, user, text)$$

For example:

> t(601114311315054592) = (Wed May 20 19:56:34 +0000 2015, CananYagmur, Arme Syriërs, hun land valt stad voor stad in de handen van IS. Verschrikkelijk. #Palmyra)

3.2 Proposed method for data reduction

As was pointed out in the related work section, it is important to find the best way to classify tweets as relevant or unrelevant. A supervised ML model can be very accurate at predicting the right label for a tweets, but these predictions cost a lot of processing power. It is very imaginable, therefore, that predicting a label for every tweet for every news article in this way, takes too much time. A real-time application depends on quick updates, so we need a way to reduce the data to more a manageable size whilst still benefiting from the accuracy offered by a supervised ML model.

We thus propose a module that precedes the classification task and selects candidates for the classification. Herein we are again informed by Phelan et al. (2010). Their research shows how simple text similarity analyses are sufficient to find articles based on tweets with a similar scope. This is not to say that only relevant tweets are found this way; but the news article and the tweets share similar words and can thus be implied to have some relation.

3.2.1 Module 1: Candidate selection

For this first module, textual data for both news articles (consisting of title and abstract) and tweets (consisting of tweet text) is used. The raw data is very noisy: the article abstracts containing HTML-tags, while the tweets contain links and hashtags. First, all textual data is stripped of HTML-tags and punctuation. The stripped text is then tokenized and assumes a bag-of-world format. We do not use lemmatization as this could compromise the weight of certain named entities.

To compare them, both news articles and tweets have to be made comparable. They should no longer be represented by their words, but rather by vectors of word frequencies. For each term (t), its frequency in a given document (news article or tweet; tf) and the amount of tweets that contain the term (df) are determined. We get the inverse document frequency as follows (where N is the total number of tweets) (Manning et al., 2008):

$$idf_t = \log \frac{N}{df_t}$$

The tf-idf weight of the term for a given document (d) is calculated as follows:

$$tfidf_{t,d} = tf_{t,d} \times idf_t$$

Both the news articles and the tweets can be represented as a vector of tf-idf weights. We determine the similarity between them by comparing the vectors. Similar to an information retrieval model (Manning et al., 2008), the news article takes on the role of a query, and the tweets take on the role of possible results. For example:

$$query = (0, 0, ..., 0.4236, 0.2245, ..., 0)$$

$$doc = (0.2101, 0, ..., 0.3291, 0, ..., 0)$$

The cosine similarity between the vectors of the query and a given tweet is computed to normalize vector lengths.

$$sim(d_1, d_2) = \frac{\vec{V}(d_1) \times \vec{V}(d_2)}{\left|\vec{V}(d_1)\,||\,\vec{V}(d_2)\right|}$$

From the similarity measures, a ranked list of results for a given news article can be constructed. For the document that served as an earlier example:

1. t(601131596683415552): 'Oude stad Palmyra vrijwel geheel in handen van IS': Terreurbeweging Islamtische Staat (IS) heeft woensdag het... http://t.co/o55dDYjo7g

244. t(601117980475465729): Ben in de stad.

The candidate selection module uses the Natural Language Toolkit (NLTK) package for Python (NLTK, 2015). NLTK is a feature-rich platform for working with textual data and, although few of its capabilities are demonstrated in this first module, will be used throughout the rest of this analysis.

3.2.2 Module 2: Relevance classification

The second module is confronted with a classification problem. Each candidate selected by the first module can be labeled relevant or non-relevant. As mentioned before, a supervised ML approach can accurately estimate class boundaries and generalize weights or rules to classify new tweets.

To do so, features that can influence the relevancy need to be selected. Here, it is useful to point out how this is no straightforward binary classification. In sentiment analysis, for example, tweets can be 'positive' and 'negative'. That particular classification task will look at word with positive and negative polarity. For the task of classifying into relevant and non-relevant tweets however, the classification is always relative to the news article. As we will see, the six selected features too are never static, but always relative to the news article.

Further, the extracted information for the tweets, apart from the user, is content-related. We will also not be using the user information, but focus purely on content-based classification. We extracted a limited amount of information from the news articles and the tweets. From them we can come to the next six features that may indicate relevance:

Timestamp difference

This feature contains the difference in seconds between the publication date of the article and the publication date of the tweet.

Named entity overlap

There was no off-the-shelf Dutch POS-tagger available to use with the NLTK package. We therefore trained a tokenize and a chunker on the CoNLL2002 Dutch Corpus and use them to determine the tags (Tjong Kim Sang, 2002). Named entities in both news articles and tweets were extracted and compared. This feature is the amount of named entities shared by the news article (title and abstract) and the tweet.

Length difference

This feature measures the difference in length (in characters) between the article title and the tweet. Like Bigram similarity, a small difference in length could signal a repetition of an article in the data.

Title similarity

The cosine-similarity between the title of the news article and the text of the tweet. The title of a news article should contain the most important words, including entities and actions.

Abstract similarity

The cosine-similarity measure between the abstract of the news article and the text of the tweet. The abstract can supplement the title in naming more related terms and entities than the title does. A user can be commenting on a very specific part of an article, or talk about implications of the discussed event.

Bigram similarity

Beyond simple term overlap, bigram similarity adds a notion of word order. The cosine-similarity measure of bigrams could signal whether tweets are just a repetition of an article title, or a retweet of a news website. In both cases, there is a high overlap, but the content of the tweet is not relevant as it is just a repetition.

All values in the eventual feature vectors are reals. Before an actual analysis, we can already anticipate on potential values and their implications. As described in the features, relevance of a tweet has to do with its contribution to a news article. It has an overlap, in the central entities, but should also represent new information. Some of the features are thus not linearly connected to relevancy, and the supervised ML algorithm used will have to assess the boundaries between the classes.

As a result, a supervised discriminative model will be used for the relevance classification module. Evenso, this leaves a large selection of models to choose from. Since the choice for a model is very influential of the results, we will evaluate four common supervised discriminative models:

K-nearest neighbors

The first is the K-nearest neighbors algorithm. It determines the location of each tweet feature vector in the vector space and deduces the tweet label from its K-nearest neighbors (Manning et al., 2008). We use the 25 nearest neighbors for each tweet.

Naive Bayes

The Naive Bayes algorithm is usually more effective in text classification, whereas our vectors contain real numbers. Naive Bayes is a probabilistic learning model in which feature values receive a probability of occuring in a class. The eventual class label for a tweet is determined by combining the seperate probabilities.

Decision Tree Classification

Decision Tree Classification is a ML method that produces simple rules. These decision rules specify the choices to be made for each analyzed tweet depending on their feature values. It could, for example, infer a rule from the training data that every tweet that does not share a named entities with an article, is non-relevant. An advantage to the resulting decision tree is that it provides insight into the most important features.

Random Forests

Random forest is an application of decision trees in which randomization is introduced to create a large variety of trees using different features. The random trees make up a random forest, and the best fitting tree models are averaged to produce the eventual predictive model.

3.3 Evaluation

Both modules need to be evaluated. For the first module, we use four news articles from Wednesday the 20th of May 2015 and Thursday the 21st of May 2015 (Table 3.1):

Table 3.1: Selected news articles

No.	hash id	timestamp	title
1	ac4f7655fc3629cc2046	Thu, 02:53:20	'Oude stad Palmyra vrijwel geheel in handen van IS'
2	2e5a12fe8b781f776740	Wed, 13:06:51	'Gegarandeerd geen gedwongen ontslagen bij Bel(...)
3	a5490acd9dab165467e7	Wed, 20:42:41	Oranje met debutante
4	9c261db23a3ae5bff921	Wed, 22:51:27	Wethouder A'dam bezorgd na undercoveractie bi(...)

For each of the four news articles we select candidates from 24 hours of Twitter data from the Corpus Nederlandstalige Tweets. From noon on Wednesday the 20th of May 2015, until noon on Thursday the 21st 2015, this results in 829,708 tweets. We will select 250 candidates per news article. We can annotate which candidates are relevant and which are not. We can then show how the destribution of relevant tweets is amongst the ranks, and determine if the cut-off point at 250 tweets is the right choice. Once a cut-off point is determined, we use the annotated data as our gold standard to evaluate the second module.

For each of the supervised ML algorithms we do a 5-fold crossvalidation on the annotated data. We will look at their accuracy, precision, recall and F-score in comparison to the baseline model but evaluate them on accuracy rather than anything else: for our real-time application is is better to show only tweets messages, even if this means missing other relevant tweets.

3.4 A bimodular approach to real-time data classification

We can visualize the proposed methods and its resulting system. The image is a herald of how a system would be applied in a real-time application (Figure 3.1)

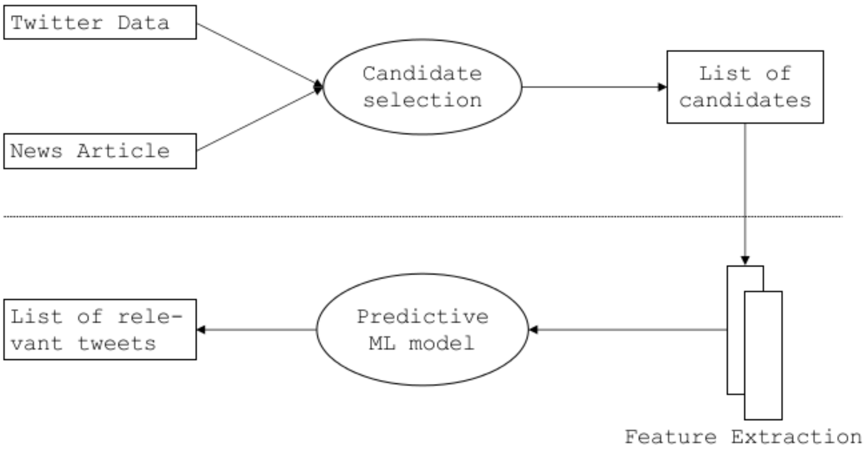


Figure 3.1: a bimodular approach to real-time data classification

Chapter 4

Results

This section contains the finding of the research. Still, some methodological considerations are discussed because part of results in evaluating the supervised machine learning models relies on the findings in the next section.

4.1 Cut off point

The ranked tweets were annotated by two annotators. Two important indications of non-relevant tweets are when a tweet only repeats information in the news article (1) and when a tweet is unrelated to the news article (3). A relevant tweet is related to the news article and adds information to the information already present in the news article (2):

1. t(601131596683415552): 'Oude stad Palmyra vrijwel geheel in handen van IS': Terreurbeweging Islamtische Staat (IS) heeft woensdag het... http://t.co/o55dDYjo7g (unrelevant; rank 1)
2. t(601114311315054592): Arme Syriërs, hun land valt stad voor stad in de handen van IS. Verschrikkelijk. #Palmyra (relevant: rank 184)
3. t(601117980475465729): Ben in de stad (unrelevant; rank 244)

We can plot the result of annotating 250 tweets per news article to cumulatively shows the amount of relevant tweets. The distribution of relevant tweets is important, because it allows us to see beyond which rank the gain of relevant tweets is declining. Choosing a cut-off point for the amounts of tweets used to make up the gold standard is still relatively arbitrary. The choice to use 250 tweets per article is partly informed by the distribution of relevant tweets among the candidates (Figure 4.1 shows this distribution per article, whilst Figure 4.2 shows the average), and partly informed by the fact the choosing 250 candidates per article makes up a gold standard of 1,000 tweets, which should prove sufficient to train and test the classifier. The first module has succesfully reduced the Twitter data for each article to 0.03 percent of its original size (from 829,708 tweets to 250 candidates).

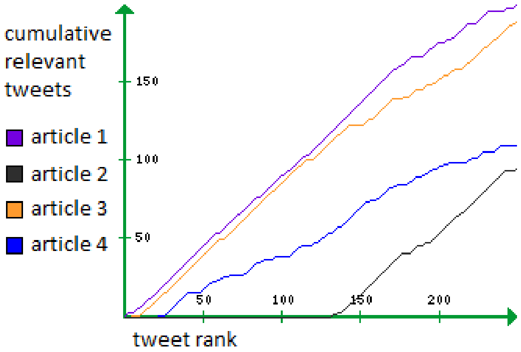


Figure 4.1: Cumulative relevant tweets per rank

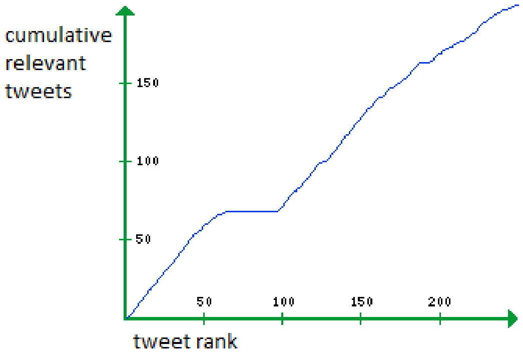


Figure 4.2: Average cumulative relevant tweets per rank

Our golden standard for the second module hence consists of 1,000 annotated tweets. The first annotator rated 593 relevant tweets, the second annotator rated 571 relevant tweets. The total agreements are shown in table 4.2. They results in a Kappa coefficient of roughly 0.934 (Manning et al., 2008).

Table 4.1: Annotation results

	Annotator 2		
Annotator 1	Relevant	Non-relevant	Total
Relevant	566	5	571
Non-relevant	27	402	429
Total	593	407	1000

$$\kappa = \frac{n_a - n_\varepsilon}{n - n_\varepsilon} = \frac{968 - 513.206}{1000 - 513.206} = 0,934267883$$

Viera and Garrett (2005) propose an interpretation for the Kappa coefficient in which a score between 0.81 and 0.99 is considered ”almost perfect agreement”. This shows that it is relatively easy for humans to judge tweet relevancy for news articles.

I decided to use the annotated data by Annotator 1 as our gold standard. For the next section it gives us the baseline performance for precision of 0.593 (the results of guessing that all tweets are relevant) and an upper bound of 0.934 (the accuracy that can be expected from humans).

4.2 Evaluation of classification algorithms

The results in table 4.3 show clear differences in algorithm performance after the 5-fold crossvalidation. As said, we will use accuracy as the leading factor in this evaluation.

Table 4.2: Evaluation metrics per supervised machine-learning model after 5-fold crossvalidation

Model	Accuracy	Precision	Recall	F-score
Baseline Model	0.5930	0.5930	1.0000	0.7445
K-nearest neighbors	0.6679	0.7322	0.7707	0.7321
Naive Bayes	0.7439	0.7489	0.8951	0.8092
Decision Tree	0.7871	0.8330	0.8060	0.8142
Random Forest	0.8671	0.8988	0.8735	0.8736

The random forest algorithm outperforms the other models significantly in all metrics but recall. Naive Bayes has a higher recall, but is likely to have labelled a large part of the data, because precision is low at 0.7489.

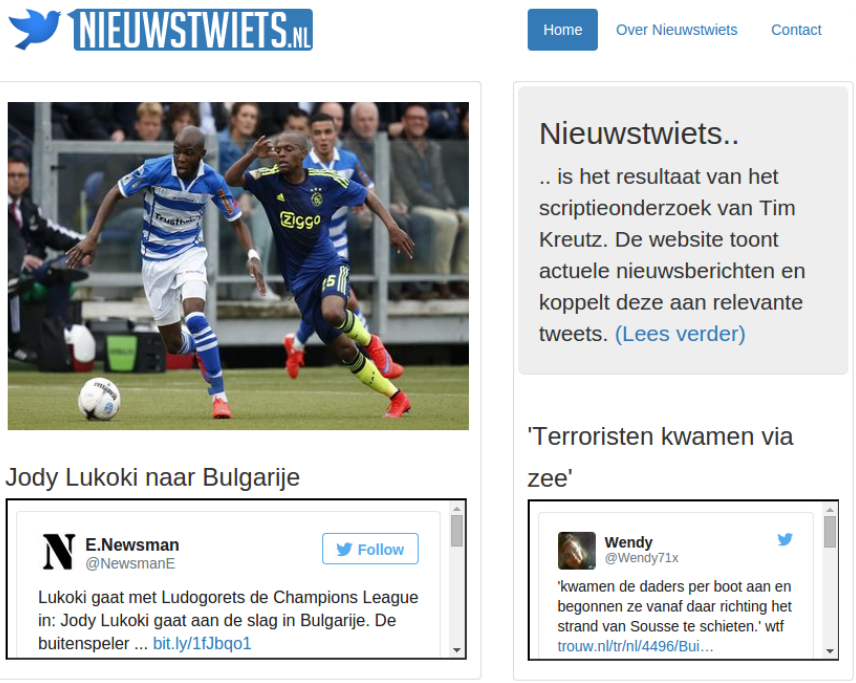
4.2.1 Most important features

Since the algorithm with the best performance, the random forest algorithm, is a decision tree algorithm, it can easily return the most important features. In order of importance they are: (1) timestamp difference, (2) title similarity, (3) length difference, (4) abstract similarity, (5) named entity overlap and (6) bigram similarity. Length difference and title similarity account for roughly 50 percent of the decisions in the tree. The other features vary from ten to thirteen percent. The predictive decision tree is included in Appendix A.

Chapter 5

Nieuwstwiets.nl

The merit of using a bimodulair approach to tweet relevancy classification is to boost performance for real-time classification. Nieuwstwiets.nl will serve as a case to demonstrate the performance of the proposed system, not in any statistics, but through a user-friendly frontend (Figure 5.1).



Nieuwstwiets..

.. is het resultaat van het scriptieonderzoek van Tim Kreutz. De website toont actuele nieuwsberichten en koppelt deze aan relevante tweets. (Lees verder)

Jody Lukoki naar Bulgarije

E.Newsman @NewsmanE

Lukoki gaat met Ludogorets de Champions League in: Jody Lukoki gaat aan de slag in Bulgarije. De buitenspeler ... bit.ly/1fJbqo1

'Terroristen kwamen via zee'

Wendy @Wendy71x

'kwamen de daders per boot aan en begonnen ze vanaf daar richting het strand van Sousse te schieten.' wtf trouw.nl/tr/nl/4496/Bui…

Figure 5.1: Layout of Nieuwstwiets

The Twitter data is not updated in real-time, but rather per hour. The website adheres to this guideline and updates itself every hour, at fifteen minutes past the full hour. It then retrieves the news articles from the RSS feeds and the last hour of Dutch tweets. Since an hour of tweets contains a lot less relevant tweets than a day of tweets, only 25 candidates are selected for each of the 41 news articles. We have exported the Random Forest model and will use it to predict the label of all the candidates. This results in a list of news articles and their relevant tweets. Some additional components were implemented to suit the user-friendly format.

Ranking news articles

On the website, there is not enough space to show all 41 news articles present in the RSS feed at the same time. Therefore, a news article ranking module is added to the proposed system from Chapter 3. The rankingmodule will simply base its ranking on the amount of tweets that are deemed relevant to a given article, and return only the top 5 articles. The 'headline'-article will in theory be the article that Twitter-users most frequently comment on.

News images

The website attempts to retrieve an image from the link that belongs to the headline article. This is often unsuccesful because the article does not have an image, or the article is from NOS.nl in which case the image is not directly referred to on the webpage itself.

Contact and more information

Users can navigate to my bachelor thesis on the website. I make my thesis openly available so that users can gain insight to the techniques used, and can expand or improve on what I have done. If the thesis itself is not clear enough, or difficulties arise, there is also the option to send me an email. I will reply as soon and in depth as I can.

The research as well as the websites are by no means perfect, and so I want to promote improvement. Some of my own critical ideas about this thesis are discussed in the next Chapter.

Chapter 6

Discussion

In this Chapter, some of the methods and results will be critically assessed and suggestions will be made for future research. The bachelor thesis for Information Science should not be too extensive. This applies both to the report itself and the depth of the research. As a result, some concessions were made throughout the research.

6.0.2 Evaluation of the first module

As was mentioned before, the choice for a cut-off point for the candidate selection was mostly arbitrary. It would be more thorough to evaluate the system for a variety of cut-off points: 250 candidates, 500 candidates, 750 candidates or even more. The cut-off point should also be influenced by which article is currently being analyzed, for some topics are more popular on Twitter than others. Future research could focus on optimizing the candidate selection module to control for news article variability. The same applies for the real-time application on Nieuwstwiets.nl

6.0.3 Selecting supervised ML models

There are a lot of other supervised ML models that could have been tested. There are other variants of Decision Trees that could have performed even better than the Random Forest model. Looking at individual features however, there is a lot of reason to believe that non-linear models perform the best.

For example: we can plot the two most important features to get an idea of the vector space (Figure 6.1). Although the plotted graph is only an abstraction, it shows us that non-relevant tweets are either very similar or very unsimilar to news articles. Further, tweets that are posted very soon or very late after a news a article are usually non-relevant.

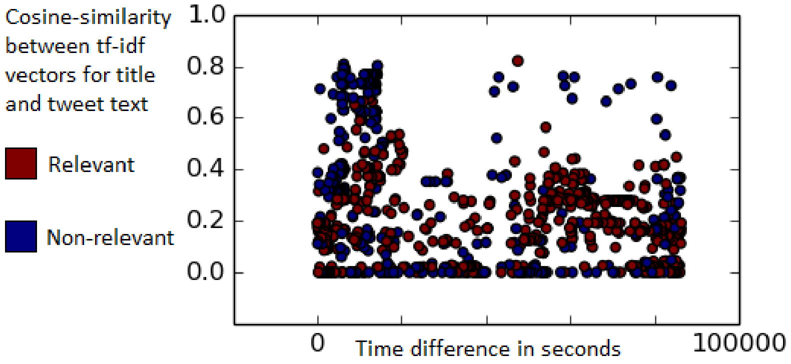


Figure 6.1: Scatterplot of relevant and non-relevant tweets

6.0.4 Results from Nieuwstwiets.nl

The results that are visible on Nieuwstwiets.nl are mostly interesting. It happens too often, however, that tweets beneath popular news articles are duplicates or retweets. There is not an additional filter for duplicates, which is an easy addition to be made, but for which I lacked the time.

Another problem is that some news articles do not list any relevant tweets below them. This is due to the news articles containing very little named entities or other rare words. The classification system is trained on four articles that all have a clear subject and for which it was relatively easy to find related tweets. During the selection of the data for the gold standard, this had not occurred to me. Future research can benefit from using a large variety of news articles in the training data.

6.0.5 Hybrid systems

What has been described as a bimodular approach, is a content-based classification system. This means that only the content of news articles and tweets, and very little meta-data, was considered for the analysis. It could prove fruitful to not only look at *what* people talk about on Twitter but also *who* was talking about it. It may turn out that particular user groups or specific users are more likely to discuss political news events, whereas other only comment on sport. Perhaps more than anything, a hybrid system in which both content and user-based information is analyzed, will have the best performance.

Chapter 7

Conclusion

Real-time detection of relevant tweets to news articles is not a straightforward task. We can conclude that when using real-time Twitter data, it is wise to make a pre-selection of the data. It can hugely reduce the problem of finding relevant tweets.

A candidate selection module that looks at text similarity is effective in reducing the data and in selecting candidates for relevancy classification. The choice for ranking systems by text similarity is encouraged by the idea that similar texts discuss similar topics and events. An inspection of the trainingdata constructed in our first module shows how the relevant tweets are distributed amongst the similarity ranks. The frequency of relevant tweets becomes higher at first, but declines towards the lower ranks. We can imply that the data that is not used in further analysis, does not contain significant amounts of relevant tweets.

We tested four discriminative supervised ML methods on our feature vectors: K-nearest neighbors, Naive Bayes, Decision Trees and Random Forest. Random Forest performs the best by far, with an accuracy of 86.71 percent. It is very effective in randomly applying features and generating a multitude of trees to find the best fit.

We demonstrate the real-time application of the research by combining the candidate selection and the random forest predictive model on hourly Twitter data. Nieuwstwiets.nl shows the five news articles with the most relevant tweets in a user-friendly format. The results on the website are not always accurate and can be improved by filtering tweet duplicates and retweets.

The results can further benefit from adding user-based information to the analysis, which is something that I suggest for future research.

Bibliography

Crowdynews (2015). Crowdynews article engager. http://www.crowdynews.com/what-we-do/article-widget/. Accessed: 2015-06-20. 4, 5

Han, B. and Baldwin, T. (2011). Lexical normalisation of short text messages: Makn sens a twitter. *HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*. 4, 5

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA. 8, 11, 14

NLTK (2015). Nltk 3.0 documentation. http://www.nltk.org/. Accessed: 2015-04-10. 9

Phelan, O., McCarthy, K., and Smyth, B. (2010). Using twitter to recommend real-time topical news. *Proceedings of the third ACM conference on Recommender systems*. 4, 5, 6, 8

Popescu, A., Pennachiotti, M., and Paranjpe, D. (2011). Extracting events and event descriptions from twitter. *WWW '11 Proceedings of the 20th international conference companion on World wide web*. 3, 4, 5

Reurings, B. (2008). Publieke opinies in online discussies. Master's thesis, Utrecht University. 2

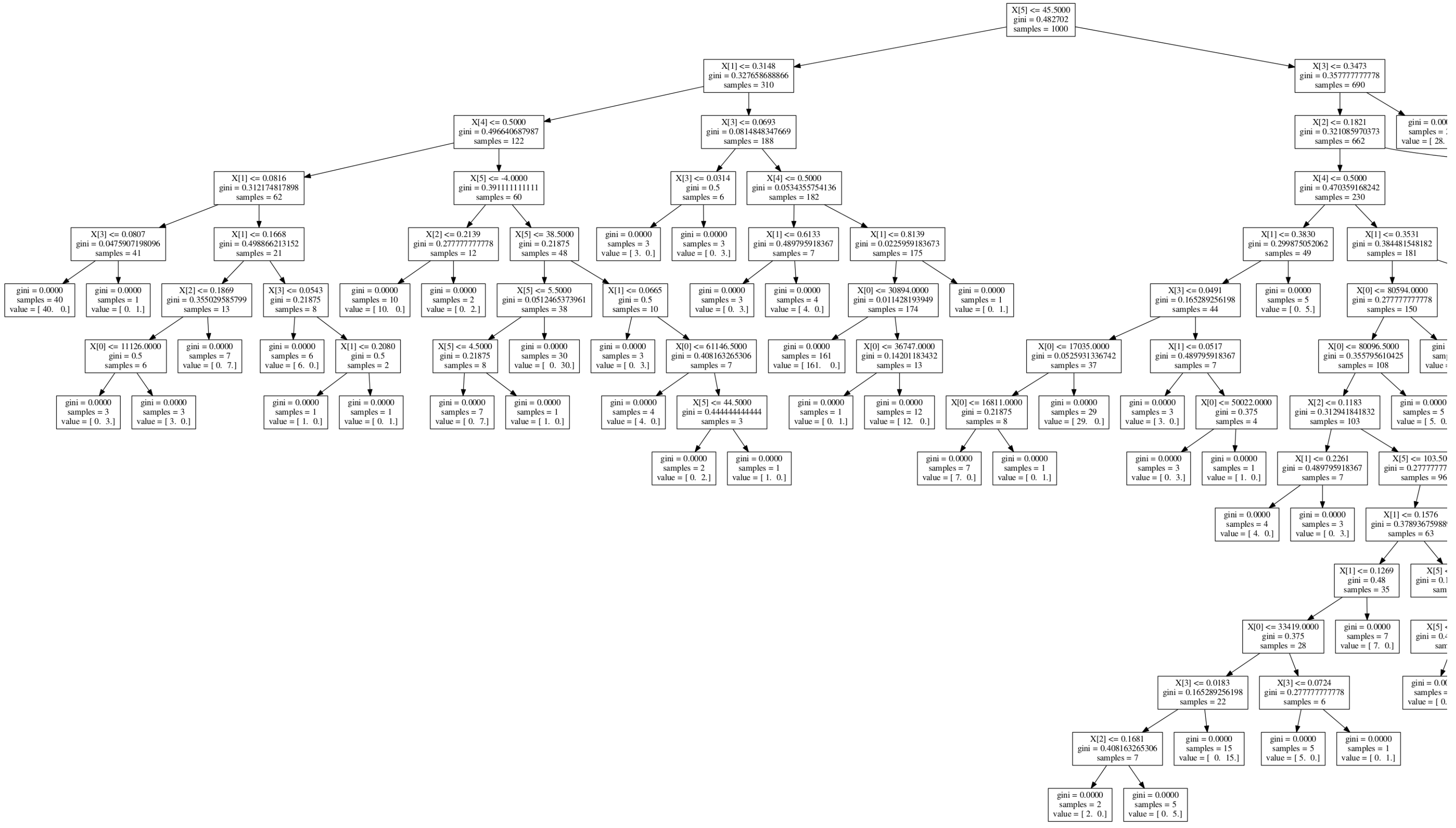
Tjong Kim Sang, E. (2002). Introduction to the conll-2002 shared task: Language-independent named entity recognition. *Proceedings of CoNLL-2002*. 10

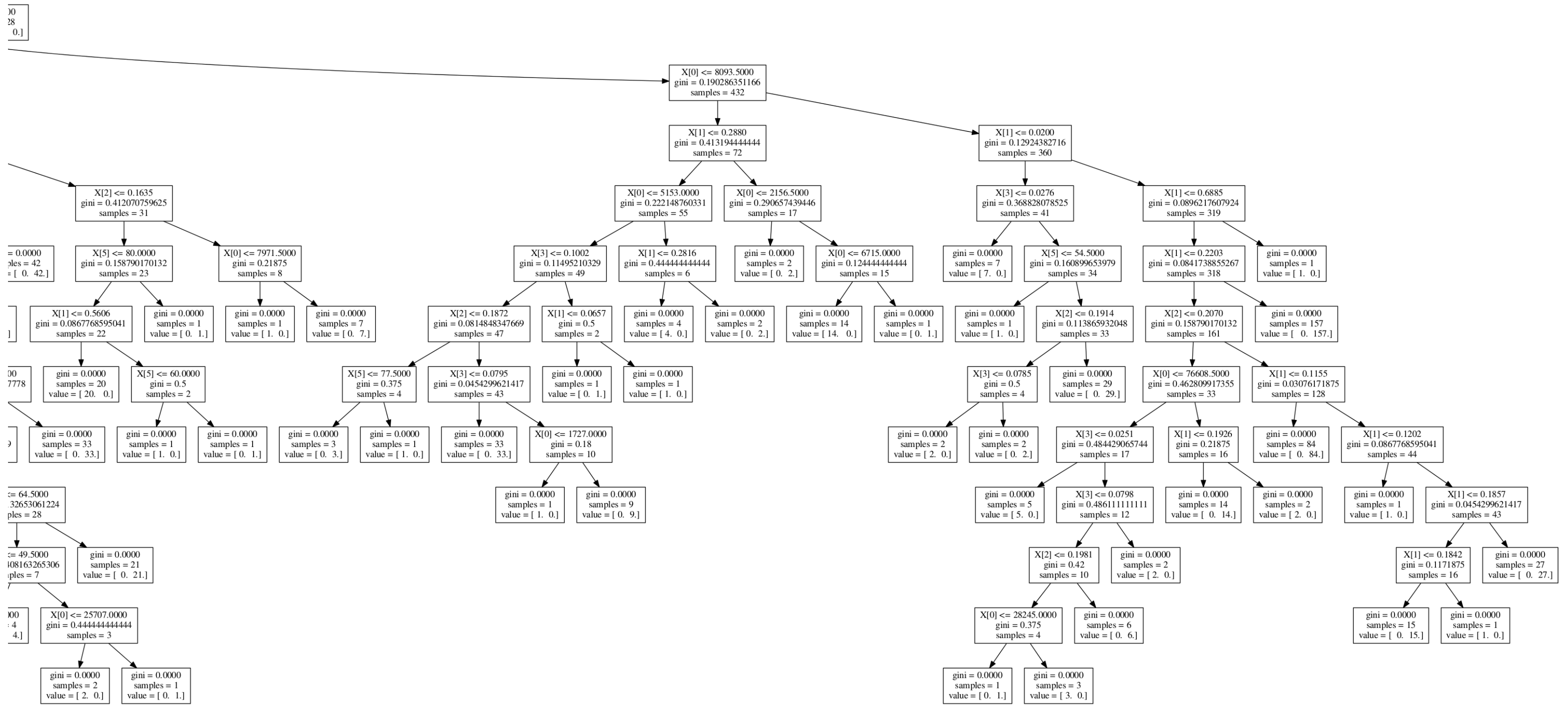
Tjong Kim Sang, E. and Bos, J. (2012). Predicting the 2011 dutch senate election results with twitter. *Proceedings of SASN 2012, the EACL 2012 Workshop on Semantic Analysis in Social Networks*. 7

Viera, A. and Garrett, J. M. (2005). Understanding interobserver agreement: the kappa statistic. *Family Medicine*. 14

Vinex (2013). Vinex resultaten 2013. http://www.vinex.nl/resultaten/archief/2013/. Accessed: 2015-05-09. 6

Appendix A: Predictive Model


X[5] <= 45.5000 gini = 0.482702 samples = 1000
X[1] <= 0.3148 gini = 0.327658688866 samples = 310
X[3] <= 0.3473 gini = 0.357777777778 samples = 690
X[4] <= 0.5000 gini = 0.496640687987 samples = 122
X[3] <= 0.0693 gini = 0.0814848347669 samples = 188
X[2] <= 0.1821 gini = 0.321085970373 samples = 662
gini = 0.00 samples = value = [28.
X[1] <= 0.0816 gini = 0.312174817898 samples = 62
X[5] <= -4.0000 gini = 0.391111111111 samples = 60
X[3] <= 0.0314 gini = 0.5 samples = 6
X[4] <= 0.5000 gini = 0.0534355754136 samples = 182
X[4] <= 0.5000 gini = 0.470359168242 samples = 230
X[3] <= 0.0807 gini = 0.0475907198096 samples = 41
X[1] <= 0.1668 gini = 0.498866213152 samples = 21
X[2] <= 0.2139 gini = 0.277777777778 samples = 12
X[5] <= 38.5000 gini = 0.21875 samples = 48
gini = 0.0000 samples = 3 value = [3. 0.]
gini = 0.0000 samples = 3 value = [0. 3.]
X[1] <= 0.6133 gini = 0.489795918367 samples = 7
X[1] <= 0.8139 gini = 0.0225959183673 samples = 175
X[1] <= 0.3830 gini = 0.299875052062 samples = 49
X[1] <= 0.3531 gini = 0.384481548182 samples = 181
gini = 0.0000 samples = 40 value = [40. 0.]
gini = 0.0000 samples = 1 value = [0. 1.]
X[2] <= 0.1869 gini = 0.355029585799 samples = 13
X[3] <= 0.0543 gini = 0.21875 samples = 8
gini = 0.0000 samples = 10 value = [10. 0.]
gini = 0.0000 samples = 2 value = [0. 2.]
X[5] <= 5.5000 gini = 0.0512465373961 samples = 38
X[1] <= 0.0665 gini = 0.5 samples = 10
gini = 0.0000 samples = 3 value = [0. 3.]
gini = 0.0000 samples = 4 value = [4. 0.]
X[0] <= 30894.0000 gini = 0.011428193949 samples = 174
gini = 0.0000 samples = 1 value = [0. 1.]
X[3] <= 0.0491 gini = 0.165289256198 samples = 44
gini = 0.0000 samples = 5 value = [0. 5.]
X[0] <= 80594.0000 gini = 0.277777777778 samples = 150
X[0] <= 11126.0000 gini = 0.5 samples = 6
gini = 0.0000 samples = 7 value = [0. 7.]
gini = 0.0000 samples = 6 value = [6. 0.]
X[1] <= 0.2080 gini = 0.5 samples = 2
X[5] <= 4.5000 gini = 0.21875 samples = 8
gini = 0.0000 samples = 30 value = [0. 30.]
gini = 0.0000 samples = 3 value = [0. 3.]
X[0] <= 61146.5000 gini = 0.408163265306 samples = 7
gini = 0.0000 samples = 161 value = [161. 0.]
X[0] <= 36747.0000 gini = 0.14201183432 samples = 13
X[0] <= 17035.0000 gini = 0.0525931336742 samples = 37
X[1] <= 0.0517 gini = 0.489795918367 samples = 7
X[0] <= 80096.5000 gini = 0.355795610425 samples = 108
gini samp value =
gini = 0.0000 samples = 3 value = [0. 3.]
gini = 0.0000 samples = 3 value = [3. 0.]
gini = 0.0000 samples = 1 value = [1. 0.]
gini = 0.0000 samples = 1 value = [0. 1.]
gini = 0.0000 samples = 7 value = [0. 7.]
gini = 0.0000 samples = 1 value = [1. 0.]
gini = 0.0000 samples = 4 value = [4. 0.]
X[5] <= 44.5000 gini = 0.444444444444 samples = 3
gini = 0.0000 samples = 1 value = [0. 1.]
gini = 0.0000 samples = 12 value = [12. 0.]
X[0] <= 16811.0000 gini = 0.21875 samples = 8
gini = 0.0000 samples = 29 value = [29. 0.]
gini = 0.0000 samples = 3 value = [3. 0.]
X[0] <= 50022.0000 gini = 0.375 samples = 4
X[2] <= 0.1183 gini = 0.312941841832 samples = 103
gini = 0.0000 samples = 5 value = [5. 0.
gini = 0.0000 samples = 2 value = [0. 2.]
gini = 0.0000 samples = 1 value = [1. 0.]
gini = 0.0000 samples = 7 value = [7. 0.]
gini = 0.0000 samples = 1 value = [0. 1.]
gini = 0.0000 samples = 3 value = [0. 3.]
gini = 0.0000 samples = 1 value = [1. 0.]
X[1] <= 0.2261 gini = 0.489795918367 samples = 7
X[5] <= 103.50 gini = 0.27777777 samples = 96
gini = 0.0000 samples = 4 value = [4. 0.]
gini = 0.0000 samples = 3 value = [0. 3.]
X[1] <= 0.1576 gini = 0.37893675988 samples = 63
X[1] <= 0.1269 gini = 0.48 samples = 35
X[5] < gini = 0.1 sam
X[0] <= 33419.0000 gini = 0.375 samples = 28
gini = 0.0000 samples = 7 value = [7. 0.]
X[5] < gini = 0.4 sam
X[3] <= 0.0183 gini = 0.165289256198 samples = 22
X[3] <= 0.0724 gini = 0.277777777778 samples = 6
gini = 0.00 samples = value = [0.
X[2] <= 0.1681 gini = 0.408163265306 samples = 7
gini = 0.0000 samples = 15 value = [0. 15.]
gini = 0.0000 samples = 5 value = [5. 0.]
gini = 0.0000 samples = 1 value = [0. 1.]
gini = 0.0000 samples = 2 value = [2. 0.]
gini = 0.0000 samples = 5 value = [0. 5.]


00
28
0.]
X[0] <= 8093.5000
gini = 0.190286351166
samples = 432
X[1] <= 0.2880
gini = 0.413194444444
samples = 72
X[1] <= 0.0200
gini = 0.12924382716
samples = 360
X[2] <= 0.1635
gini = 0.412070759625
samples = 31
X[0] <= 5153.0000
gini = 0.222148760331
samples = 55
X[0] <= 2156.5000
gini = 0.290657439446
samples = 17
X[3] <= 0.0276
gini = 0.368828078525
samples = 41
X[1] <= 0.6885
gini = 0.0896217607924
samples = 319
= 0.0000
ples = 42
= [0. 42.]
X[5] <= 80.0000
gini = 0.158790170132
samples = 23
X[0] <= 7971.5000
gini = 0.21875
samples = 8
X[3] <= 0.1002
gini = 0.11495210329
samples = 49
X[1] <= 0.2816
gini = 0.444444444444
samples = 6
gini = 0.0000
samples = 2
value = [0. 2.]
X[0] <= 6715.0000
gini = 0.124444444444
samples = 15
gini = 0.0000
samples = 7
value = [7. 0.]
X[5] <= 54.5000
gini = 0.160899653979
samples = 34
X[1] <= 0.2203
gini = 0.0841738855267
samples = 318
gini = 0.0000
samples = 1
value = [1. 0.]
X[1] <= 0.5606
gini = 0.0867768595041
samples = 22
gini = 0.0000
samples = 1
value = [0. 1.]
gini = 0.0000
samples = 1
value = [1. 0.]
gini = 0.0000
samples = 7
value = [0. 7.]
X[2] <= 0.1872
gini = 0.0814848347669
samples = 47
X[1] <= 0.0657
gini = 0.5
samples = 2
gini = 0.0000
samples = 4
value = [4. 0.]
gini = 0.0000
samples = 2
value = [0. 2.]
gini = 0.0000
samples = 14
value = [14. 0.]
gini = 0.0000
samples = 1
value = [0. 1.]
gini = 0.0000
samples = 1
value = [1. 0.]
X[2] <= 0.1914
gini = 0.113865932048
samples = 33
X[2] <= 0.2070
gini = 0.158790170132
samples = 161
gini = 0.0000
samples = 157
value = [0. 157.]
00
7778
gini = 0.0000
samples = 20
value = [20. 0.]
X[5] <= 60.0000
gini = 0.5
samples = 2
X[5] <= 77.5000
gini = 0.375
samples = 4
X[3] <= 0.0795
gini = 0.0454299621417
samples = 43
gini = 0.0000
samples = 1
value = [0. 1.]
gini = 0.0000
samples = 1
value = [1. 0.]
X[3] <= 0.0785
gini = 0.5
samples = 4
gini = 0.0000
samples = 29
value = [0. 29.]
X[0] <= 76608.5000
gini = 0.462809917355
samples = 33
X[1] <= 0.1155
gini = 0.03076171875
samples = 128
9
gini = 0.0000
samples = 33
value = [0. 33.]
gini = 0.0000
samples = 1
value = [1. 0.]
gini = 0.0000
samples = 1
value = [0. 1.]
gini = 0.0000
samples = 3
value = [0. 3.]
gini = 0.0000
samples = 1
value = [1. 0.]
gini = 0.0000
samples = 33
value = [0. 33.]
X[0] <= 1727.0000
gini = 0.18
samples = 10
gini = 0.0000
samples = 2
value = [2. 0.]
gini = 0.0000
samples = 2
value = [0. 2.]
X[3] <= 0.0251
gini = 0.484429065744
samples = 17
X[1] <= 0.1926
gini = 0.21875
samples = 16
gini = 0.0000
samples = 84
value = [0. 84.]
X[1] <= 0.1202
gini = 0.0867768595041
samples = 44
<= 64.5000
.32653061224
ples = 28
gini = 0.0000
samples = 1
value = [1. 0.]
gini = 0.0000
samples = 9
value = [0. 9.]
gini = 0.0000
samples = 5
value = [5. 0.]
X[3] <= 0.0798
gini = 0.486111111111
samples = 12
gini = 0.0000
samples = 14
value = [0. 14.]
gini = 0.0000
samples = 2
value = [2. 0.]
gini = 0.0000
samples = 1
value = [1. 0.]
X[1] <= 0.1857
gini = 0.0454299621417
samples = 43
<= 49.5000
408163265306
ples = 7
7
gini = 0.0000
samples = 21
value = [0. 21.]
X[2] <= 0.1981
gini = 0.42
samples = 10
gini = 0.0000
samples = 2
value = [2. 0.]
X[1] <= 0.1842
gini = 0.1171875
samples = 16
gini = 0.0000
samples = 27
value = [0. 27.]
000
4
4.]
X[0] <= 25707.0000
gini = 0.444444444444
samples = 3
X[0] <= 28245.0000
gini = 0.375
samples = 4
gini = 0.0000
samples = 6
value = [0. 6.]
gini = 0.0000
samples = 15
value = [0. 15.]
gini = 0.0000
samples = 1
value = [1. 0.]
gini = 0.0000
samples = 2
value = [2. 0.]
gini = 0.0000
samples = 1
value = [0. 1.]
gini = 0.0000
samples = 1
value = [0. 1.]
gini = 0.0000
samples = 3
value = [3. 0.]