



university of
 groningen

faculty of arts

PREDICTING SOMEONES POLITICAL PREFERENCE
WITH TWITTER
CLASSIFICATION BASED ON WORD USE, FRIENDS AND
HASHTAGS

Tomer Gabay

Bachelor Thesis
Information Science
Tomer Gabay
s2726769
August 28, 2025

ABSTRACT

A lot of research has been done in the field of Twitter classification and politics. However, classification by only looking to similarities between Twitter users' word use, accounts that they follow (*friends*) and hashtags, has not been done before. Up to 200 tweets and 200 friends of 1000 users for each of four different parties in the Dutch General Elections of 2017 (CDA, D66, GroenLinks, PVV) have been used in combination with Machine Learning algorithms to create a classification system. These 4000 users have been labeled in a *distant supervision* manner, with an accuracy of 90%. Five different features were tested: unigrams, bigrams, trigrams, friends and hashtags. The Tweets were split into three data sets: only personal tweets, only retweets, and all tweets. By comparing Naive Bayes and Support Vector Machine (SVM) classification algorithms for each of the five features on the three different data sets, accuracy scores between 61% and 90% were achieved, with the baseline set on 25%. The highest score of 90% was achieved by running SVM on unigrams on the retweets data set. This research provides evidence that analyzing similarities in data of Twitter users is useful to classify them to political parties.

CONTENTS

Abstract	i
Preface	iii
1 INTRODUCTION AND BACKGROUND	1
2 DATA AND MATERIAL	3
2.1 Collection	3
2.2 Annotation	4
2.3 Processing	5
3 METHOD	7
4 RESULTS AND DISCUSSION	9
5 CONCLUSION	10
Appendices	12
A DATA AND MATERIAL	13
B METHOD	16
C RESULTS	17

PREFACE

After only two and a half years since starting my Information Science bachelor, I was already starting with the final procedure: writing a thesis. I heard from friends and former students that it was worst part of a life as a student. Well, after having completed my thesis, I have no idea what they were talking about. I actually think that writing my thesis was not the worst, but the most fun part of my entire bachelor study. I was finally able to use all the skills I had acquired in other courses for a research in my own interests, which combines three fields I love: politics, programming and data analysis. I have enjoyed the process of making this thesis and I am happy with the final product. Of course, all of this wasn't possible without the aid of the M. Nissim, dr. A. Toral Ruiz and all other teachers of the University of Groningen. However, I must do a special thanks to dr. L. Bosveld-de Smet for helping me during my entire study, being not only my thesis supervisor, but also my study advisor and a very capable teacher.

1

INTRODUCTION AND BACKGROUND

It is the 15th of march, 2017. The Dutch adult population can vote for candidates and their responding parties for the House of Representatives. After the British population voted for the so called *bexit*, and the Americans for Donald Trump as their new president, Europe is nervously watching the Dutch election unfold. Is populism going to win again?

The political tensions in the world have increased since the global economic crisis of 2008 and the rise of radical Islamic terror during the Arab Spring. Right-wing populist parties like Front National of Marie le Pen in France and the Party for Freedom (PVV) of Geert Wilders in the Netherlands are increasing in popularity.

People all over the world express their opinion on the changing political landscape on social media. One of the most popular social media platforms is the micro blogging platform Twitter, where people can posts *tweets*, containing up to 140 characters on their timeline. If you are *following* a Twitter user, the tweets of that user will appear on your timeline. The Twitter users you follow, are called your *friends*¹.

Many researches have been done related to politics and the use of Twitter data to analyze and/or predict political opinions of users and the outcome of elections. Makazhanov et al. (2014) developed a system that can predict elections with high accuracy based on the alignment of a user's language model with the language model of a party's candidates. Sang and Bos (2012) created a program which used sentiment analysis on tweets with a reference to a political party to predict the Dutch Senate Election of 2011. Burnap et al. (2016) did a study somewhat like Sang and Bos, but for the English General Elections of 2015. Tumasjan et al. (2010) used LIWC's text analyzing system to predict the federal elections of Germany in 2009. Of course, Twitter combined with politics is not just about predicting elections. Golbeck and Hansen (2014) for example, built a system which analyzed the political preference of an organization's followers and used degrees of liberalism and conservatism.

What I noticed while reading these articles was that almost all researches were based around sentiment analysis of tweets that contained words referring to a party. I wondered if it was that necessary to use sentiment analysis with tweets about parties to know which party a user supports. Would finding similarities between users voting for a party be sufficient for building a classifying system which can predict the political preference of a user? For example, I expect voters of the right-wing populist PVV party to tweet more about immigration and Islam than voters of any other party, while the supporters of GroenLinks would tweet more about climate change. Friends can be a useful feature as I assume that almost every voter for party x follows the official Twitter party account of party x and the corresponding party leader, and thus that those accounts are one of the most informative friends a user has. I also expect voters for the same political party to often use the same hashtags, because most people do not make up hashtags themselves, but use already existing hashtags, and thus probably of the people they follow.

Being able to classify Twitter users to a political party is interesting for multiple reasons. For instance, it makes it possible for political parties to see who will vote for them, and who not. Being able to identify the people who are not intending to vote for you can lead to more accurate promoting campaigns, especially if you know for which party they are intending to vote for at that moment. For example, the Dutch VVD party could convince PVV voters to vote for them by specifically targeting them with harder language on Islam and immigration, while convincing

¹ <https://dev.twitter.com/rest/reference/get/friends/ids>, 14-03-17

voters for the D66 party will need more focus on education. Also, if you can identify the political preference of popular Twitter users and convince them to vote for you, they might convince their followers to do the same.

The research question for my thesis is:

Which Machine Learning algorithm has the best performance in classifying Dutch Twitter users, based on their word use, friends and hashtags?

Which brings up these sub questions:

- *Which classification method has the best performance when looking at word use?*
- *How many N-grams for word use leads to the best performance?*
- *Which classification method has the best performance when looking at friends?*
- *Which classification method has the best performance when looking at hashtags?*
- *How accurate can you classify Twitter users to a political party?*

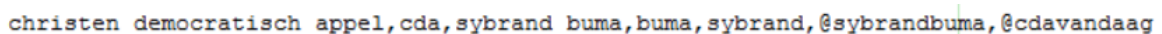
A research which is somewhat similar is that of [van Dalen \(2016\)](#). Van Dalen has already concluded that word use is an useful feature for classifying Twitter users to a political party using a Machine Learning algorithm based on Logistic Regression. He acquired an accuracy of 46 percentage point higher than the baseline. However, van Dalen did not use hashtags or friends as a feature, and chose for Logistic Regression while I'll use Support Vector Machines (SVM) and Naive Bayes (see chapter 3).

2 | DATA AND MATERIAL

2.1 COLLECTION

The Twitter data is gathered from two different sources: The Twitter Corpus of the University of Groningen and through Twitter's API, with the aid of the Tweepy library for Python (See Figure 4, Appendix A for a flowchart of how data was collected). Tweets from the first until the 17th of March 2017 were collected, as the elections were on the 15th of March, from the Twitter Corpus of the RUG. This is done with the `tweet2tab` function. A separate .txt file is created with all the tweets, and their corresponding tweeter, of each day, leading to a total of 1.4 GB of data. Tweets with the character combination 'stem' (*vote*) in it and the user name of the account who tweeted it, are then saved to a new .txt file. There were 532.084 tweets found with 'stem' in it. The character combination 'stem' must be in the tweet, as 'stem' is necessary to fit the definition of a party committing user. I use the following definition to determine if a user is committed to a party: "A user u commits to party p when u tweets that u will or has voted for p , or when u calls others to vote for p , whether by tweeting or retweeting". Retweets do not need to be excluded as [Conover et al. \(2011\)](#) concluded that retweets usually express agreement with the original tweet. I make the assumption that someone who expresses to commit him/herself to one party around election time will vote for that party during the elections. It is important to filter out the users who fit the definition of committing to multiple parties.

We can find these party committing tweets by searching for the characters 'stem' in tweets. With 'stem' we can not only find the word for *vote*, but also for *have voted*; 'heb gestemd', which is an approach similar to the approach [Makazhanov et al. \(2014\)](#) used in their research. We can link users to a political party by looking at the combination of 'stem' with the key words of a political party. The key words of a political party are according to this format: *party name, abbreviation of party name, full name of party leader, last name of party leader, first name of party leader, account of party leader, official account of party* (Figure 1).



```
christen democratisch appel,cda,sybrand buma,buma,sybrand,@sybrandbuma,@cdavandaag
```

Figure 1: Example of keywords of the CDA party

There were 9257 party committing users found out of 532.084 tweets with 'stem' (Table 1). For the parties D66 (social liberal, pro Europe), PVV (right-wing populist, anti-Europe, anti-Islam), GroenLinks (green liberalism) and CDA (christian centre party), there were enough voters identified (at least 1000). The reason I set the limit to 1000 is that I prefer fewer parties with a lot of data over all parties, with some having too little and some too much data (which results in too skewed data sets). For other parties were not a lot voters identified. This does make sense though, as there were a lot of small parties competing in the elections. It is however remarkable that the voters of the biggest party, the VVD (conservative liberalism), were only found in small numbers. They apparently do not express their political preference as much or in the same style as voters of other large parties in the Netherlands.

1000 users have been picked randomly for each of the selected four parties. Their party committing tweet and the party they are committed to, are the values of a dictionary with their user name as key. I chose to use the same amount of users for every party because the parties ended up close in the elections (between 9.1 and 13.1% of the votes), and Twitter users probably do not correctly resemble the voters'

population (Sang and Bos, 2012). Because each party has the same size, and there are four parties, the baseline is set to 25%.

	Users
VVD	190
PVV	2464
CDA	1529
D66	2926
GroenLinks	1218
SP	441
PvdA	244
PvdD	106
50PLUS	20
SGP	32

Table 1: Users found per party¹.

Next, we have to gather more data of these 4000 users, as we only have their party committing tweets at this stage. To gather more data, I have used the Tweepy library for Python to connect with Twitter’s API. The goal was to download x tweets and all friends per user. Unfortunately, Twitter has strict limits on its API use. Because of this, I download up to 200 tweets and 200 friends per user, and was not able to download more than one user per minute, leading to download times over 65 hours. An average of 183 tweets and 166 friends per user was reached after the downloading was done. Data was immediately written to a pickle file after downloading to prevent data getting lost if something went wrong during the 65 hours. The data was saved in the form of a dictionary, with user name as key, and `[[tweets],[friends],party committed to]` as value.

2.2 ANNOTATION

I use five different features to classify a Twitter user to a political party:

- The unigrams found in their tweets
- The bigrams found in their tweets
- The trigrams found in their tweets
- Their friends
- Their hashtags

The labels are the names of one of the four political parties: PVV, CDA, D66 and GroenLinks. These labels are determined in a *distant supervision* manner, namely, when a user has tweeted a tweet which confined to the definition described in 2.1. However, because this is *distant supervision*, an annotator is needed to determine the accuracy of the labels. Thijs Kuipers, a student with years of Twitter experience, has annotated 100 random party committing tweets to determine the accuracy of the labels. Out of 100 tweets, 90 were annotated as correct, while 10 labels were annotated as wrong or uncertain (Table 2). All parties except D66 have an accuracy over 95%. The reason why the party committing tweets of D66 are 72% is that one sarcastic tweet about D66 was retweeted a lot (see second tweet in Figure 5, Appendix A). Six of the seven wrong labeled tweets of D66 are because of this specific retweet.

¹ Table sorted on amount of seats in the Dutch House of Representatives after the elections of 2017, with VVD as biggest party.

	Correct	Wrong	Total	% Correct
CDA	25	1	26	96.2
D66	18	7	25	72.0
GroenLinks	25	1	26	96.2
PVV	22	1	23	95.7
Total	90	10	100	90.0

Table 2: Results of annotation of 100 random party committing tweets

2.3 PROCESSING

At this point, we have the tweets, friends and favorite party of 4000 users. Now we have to process this data before we can send it to the classifier. This process is done three times: for all tweets, for personal tweets² only and for retweets only (why is explained in chapter 3).

Two main paths of processing are used: one process for each user, and one process for each tweet of each user.

1. For each user, its friends are appended as one string, with white spaces between each friend, to a list, and their party to another list. Also all friends are individually appended to a list where each item is a single friend (two left paths in Figure 7, Appendix A).
2. This process is done for each user, and each of their tweets. First, the tweet is lower cased and tokenized using the TweetTokenizer function of NLTK. The advantage of TweetTokenizer is that it does convert *doei!* to *['doei','!']*, but does not convert *@pieter* to *['@','pieter']*, and does not convert *#cda* to *['#','cda']*, thus leaving import Twitter characteristics like mentions and hashtags unchanged. The tweet is compared with the user's party committing tweet (which is also lowercased and TweetTokenized) and is excluded from the research if they are identical, to prevent bias. At the same time, the tweet is checked if it is a retweet or not, unless all tweets are allowed to pass. Thus the party committing tweets and tweets of the wrong type have been filtered out at this point. I decided to not delete punctuation, as there might be a slight difference in punctuation use between supporters of different parties. I also chose to not remove predefined stop words, because even here might be a slight difference between different parties' supporters. Besides, most predefined stop word removal functions are based on 'normal' texts and not on Twitter text.

The hashtags are split from words and punctuation, as I use hashtags as a separate feature. Of the words and punctuation, bigrams and trigrams are made with their corresponding NLTK functions; *bigrams()* and *trigrams()*. The bigrams and trigrams form a tuple in the format *(unigram, unigram)* or *(unigram, unigram, unigram)*. To fit the right format for Scikit-Learn, they are concatenated with three underscores in between, so that *(unigram, unigram)* becomes *'unigram__unigram'*.

Each unigram, bigram, trigram and hashtag are appended to 'total' lists with all their occurrences. Thus, each item in that list is a single vector item *[unigram, unigram,...]* (middle path in Figure 7, Appendix A).

All the unigrams, bigrams, trigrams and hashtags of a user are also appended to 'X' lists, where each item in that list is the feature vector of a single user: *[[unigrams of user x],[unigrams of user y],...]* (right path in Figure 7, Appendix A). The *counter()* function is ran on the 'total' lists, so that each item is now a key of a dictionary, with its number of occurrences as value. The 'X' lists, of friends, hashtags, uni-, bi- and trigrams remain intact, while also new types

² Personal tweets are tweets which are composed by the Twitter user himself, and thus not a retweet

of list will be created out of them. The feature vectors of each user are compared with the corresponding counter type. If an item in a user's feature vector is an unique one (occurrences in counter equal to one), its classifying value is none, and this item can be discarded³. The reason to do this is reducing the data, otherwise most computers will not be able to classify it with NLTK. This filtered feature vector of a user is then appended to a 'feats' list: [(bag_of_words(*filtered feature vector*),party),(...)] (middle path in Figure 7, Appendix A). Toral provided the bag_of_words function, which turns a list into a dictionary, with true as value for each item present in that list (Figure 8, Appendix A). This type of format is necessary when classifying with NLTK instead of Scikit-Learn.

At last, the 'X' lists, the label list, and the 'feats' lists are combined in a 'total data' list, which is written to a pickle file.

³ This tip was provided by dr. A. Toral Ruiz

3 | METHOD

I chose to use the ScikitLearn library for Python for the classifying process. I have had previous experience with classifying, using the NLTK library, but as dr. L.M. Bosveld-de Smet and M. Nissim advised the Scikit-Learn library, I decided to follow their advice. There is a map on the website of Scikit-Learn which helps you choose the right classifying algorithms (Figure 2). Each feature in this research has over 100.000 samples, so there is no need for a SGD Classifier. Although the map advises to use Naive Bayes if Linear SVC did not work, I used both instantly, so it is possible to compare the results of the two different classifying algorithms. A disadvantage of Scikit-Learn is that the library hardly provides any options to analyze the results they have given as output. The library is simple to use, but does not satisfy the answer why certain results came out. To do get an answer how a Twitter user of, for example the CDA party, is recognizable, NLTK does provide an useful function: `show_most_informative_features(x)`. x is the amount of informative features NLTK will show. For this research, x has been set to 50, so that the most important examples for each party are visible without having a overload of data needed to be displayed.

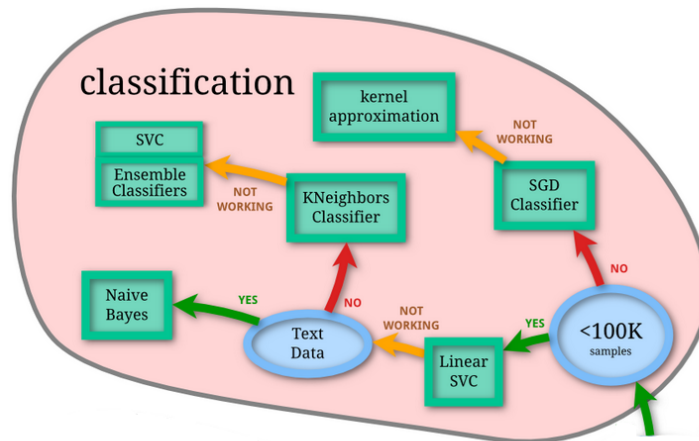


Figure 2: Help map on how to decide classification algorithm.¹

There are five lists of features, containing 4000 items, with each item being the vector of a feature of a single user. The sixth list is a list where each item is the political party the user supports, and thus the list of labels. The process of making this lists have been described in the previous chapter. Each of the feature 'X' lists is sent together with the label list to the classify function. Here, the 'X' and label list are first split into 90% training and 10% test data (20% of the training data was used as a development set). Then these lists are sent to a Naive Bayes and SVM classifier algorithm of the Scikit-Learn library². The development set indicated that Naive Bayes works better with a count vectorizer, while SVM works better with a td-idf vectorizer (Table 4, Appendix B). Thus Naive Bayes is combined with a count vectorizer and SVM with a td-idf vectorizer. The classifier is trained and tested and returns the results in a table-wise format (Figure 3).

Now the question is how did this classifier come to these results. As mentioned be-

¹ http://scikit-learn.org/stable/tutorial/machine_learning_map/ (31-05-17)

² The basis of the code needed for this has been provided by dr. L.M. Bosveld-de Smet and M. Nissim

```

Feature: unigrams
Classification: Naive Bayes
Vectorizer: Count
Accuracy: 0.75

              precision    recall  f1-score   support

   CDA         0.75         0.90         0.82         20
   D66         0.67         0.40         0.50         15
GroenLinks     0.79         0.65         0.71         17
   PVV         0.76         0.95         0.84         20

 avg / total     0.74         0.75         0.73         72

```

Figure 3: Example of output of the results of a classifier on the development set, with both personal and retweets.

fore, the `show_informative_features()` of NLTK can provide some answers. But as this data was classified by Scikit-Learn and not NLTK, classifying again with NLTK was needed to get these informative features. That is where the 'feats' lists are for. NLTK classifies these with Naive Bayes, and then the `show_informative_features()` function is able to run on the classifier. After running the `show_informative_features()` on the development set, which included both retweets and personal tweets, almost all informative features were coming from retweets (Figure 9, Appendix B). This is probably because retweets are very political segregated (Conover et al., 2011), and thus very informative for classifiers. However, I wanted to find out if it is possible to classify users to a political party on their word use, and a retweet is not in their own words, but in the words of somebody else. Therefore, I decided to repeat the classifying algorithm three times: for all tweets, for personal tweets only and for retweets only. Comparing those results makes conclusions possible on how important retweets are for classifying.

4

RESULTS AND DISCUSSION

The results are based on a training set of 3600 and a test set of 400 tweets. The Support Vector Machine classification based on unigrams from retweets only had the highest accuracy score: 90%, while hashtags of personal tweets classified with Naive Bayes had the lowest score: 61% (Table 3). With a baseline of 25%, all classifications varieties have performed far above the baseline score. As expected, the accuracy scores of SVM are almost always higher than the accuracy scores of the same type classified Naive Bayes wise: SVM scored higher on 14 out of the 15 variances of accuracy scores. However, several times Naive Bayes resulted in higher precision, recall and/or f1 score for a party than SVM (see Table 6 Table 7 Table 8, Appendix C). The D66 party is the worst performing party on recall, which probably is effected by their lower party committing tweet annotation score in comparison to the other parties.

The accuracy score for friends is hardly of influence by the classification method (87 versus 88%).

	Naive Bayes			SVM		
	Personal	Retweets	All Tweets	Personal	Retweets	All Tweets
Unigrams	0.71	0.85	0.84	0.81	0.90	0.89
Bigrams	0.74	0.86	0.86	0.79	0.89	0.89
Trigrams	0.72	0.86	0.87	0.76	0.88	0.88
Friends	0.87	0.87	0.87	0.88	0.88	0.88
Hashtags	0.61	0.88	0.85	0.70	0.87	0.87
Average	0.73	0.86	0.86	0.79	0.88	0.88

Table 3: Accuracy Scores

The hypothesis that voters for GroenLinks talk more about climate and voters for PVV more about islam and immigration can be confirmed by, for instance, looking at the output of the `most_informative_features()` function of NLTK on hashtags found in personal tweets (Figure 11, Appendix C). Examples of informative hashtags found for GroenLinks are `#klimaat` (*climate*) and `#klimaatverandering` (*climate-change*). Examples of informative hashtags for PVV voters are `moslims` (*muslims*), `#islam` and `#grenzendicht` (*close the borders*).

My expectation that official Twitter accounts of political parties and their party leaders were the most important friends for classifying was wrong. The official party account and party leader of none of the four parties were found in the top 50 most informative friends. The top three most informative friends of CDA (Esther_de_Lange, JAWCJanssens, AgnesMulderCDA), GroenLinks (rikgrashoff, SybrenKooistra, destembus2017) and D66 (LettyDemmers, tRadicaleMidden, JanTerlouw) all were Twitter accounts with a focus on politics, while the top three most informative friends of PVV voters (JoostNiemoller, V_of_Europe, StuivenbergInfo) were spreaders of (alternative) news. That the news accounts a user follows indicates which party the user will vote for is not unexpected though, as Golbeck and Hansen (2014) already concluded that in their research. It is remarkable though, that all three most informative friends of the PVV are accounts which mostly spread news, while for the other parties the top three informative friends were political related accounts. Retweets are far more important in determining someones' political preference than personal tweets, as the accuracy scores of retweets are always higher than those of personal tweets (Table 3). This is especially true for hashtags.

5 | CONCLUSION

Classifying Twitter data with Naive Bayes works best when using a count vectorizer, while classifying Twitter data with SVM works best with a tf-idf vectorizer. If you want to classify a Twitter user based on their word use, whether using only personal tweets, only retweets, or both, classifying unigrams with a Support Vector Machine algorithm produces the highest results: 81% for personal tweets, 90% for retweets and 89% for all tweets.

There is barely a difference in classification scores on friends between Naive Bayes and Support Vector Machine: 87% versus 88%. Hashtags also require SVM to get the highest results.

The best way to classify Twitter accounts to a political party is by classifying their unigrams, found in their retweets, with a SVM algorithm. This scores with 90% 65 percentage point above the baseline of 25%. Besides that, both Naive Bayes and SVM perform for each used feature far above the baseline of 25%. Thus, both word use, friends and hashtags are useful features for classifying a user to a political party.

There are some limitations to this research as a distant supervision annotation manner was used. First, the accuracy of the labels was 90%, which means 10% of the labels potentially had a wrong label, and thus the training and test set were not completely accurate. A better data set might lead to even better results. Second, every user must have tweeted at least once their political preference (or at least the algorithm must thought they did), so all users in this research are outspoken of their political preference, and are thus easier to classify than completely random users. Also, all parties used in this research have quite a distinct political viewpoint, which makes their voters easier to discriminate.

There is so many future work possible based on this research. Even though this research provided the answers I was searching for, it also raised a lot of new questions. Does including more parties with more similar political viewpoints reduce the scores? Do the scores improve for example in the political landscape of the United States of America which is split into just two main competing parties: the Democrats and the Republicans? Do the scores improve if more data per user is used? Does the length of a hashtag says something about its importance for classifying?

These are just a few examples of question that can be answered with minimal effort, with just a few adaptations to the Python programs I made for this research. It was beyond the scope of this research to answer all these questions, especially with the limited time given.

BIBLIOGRAPHY

- Burnap, P., R. Gibson, L. Sloan, R. Southern, and M. Williams (2016). 140 characters to victory?: Using twitter to predict the {UK} 2015 general election. *Electoral Studies* 41, 230 – 233.
- Conover, M., J. Ratkiewicz, M. R. Francisco, B. Gonçalves, F. Menczer, and A. Flammini (2011). Political polarization on twitter. *ICWSM* 133, 89–96.
- Ghosh, A., G. Li, T. Veale, P. Rosso, E. Shutova, J. Barnden, and A. Reyes (2015, June). Semeval-2015 task 11: Sentiment analysis of figurative language in twitter. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, Denver, Colorado, pp. 470–478. Association for Computational Linguistics.
- Golbeck, J. and D. Hansen (2014). A method for computing political preference among twitter followers. *Social Networks* 36, 177–184.
- Makazhanov, A., D. Rafiei, and M. Waqar (2014). Predicting political preference of twitter users. *Social Network Analysis and Mining* 4(1), 193.
- Nederhoed, P. (2010). *Helder rapporteren: Een handleiding voor het opzetten en schrijven van rapporten, scripties, nota's en artikelen*. Bohn Stafleu van Loghum.
- Sang, E. T. K. and J. Bos (2012). Predicting the 2011 dutch senate election results with twitter. In *Proceedings of the Workshop on Semantic Analysis in Social Media*, Stroudsburg, PA, USA, pp. 53–60. Association for Computational Linguistics.
- Tumasjan, A., T. O. Sprenger, P. G. Sandner, and I. M. Welp (2010). Predicting elections with twitter: What 140 characters reveal about political sentiment. *ICWSM* 10(1), 178–185.
- van Dalen, R. G. (2016). Automatic classification of dutch twitter users based on political preference using their tweets. *Bachelor Thesis, Faculty of Arts, University of Groningen*.

Appendices

Appendix not entirely complete yet.

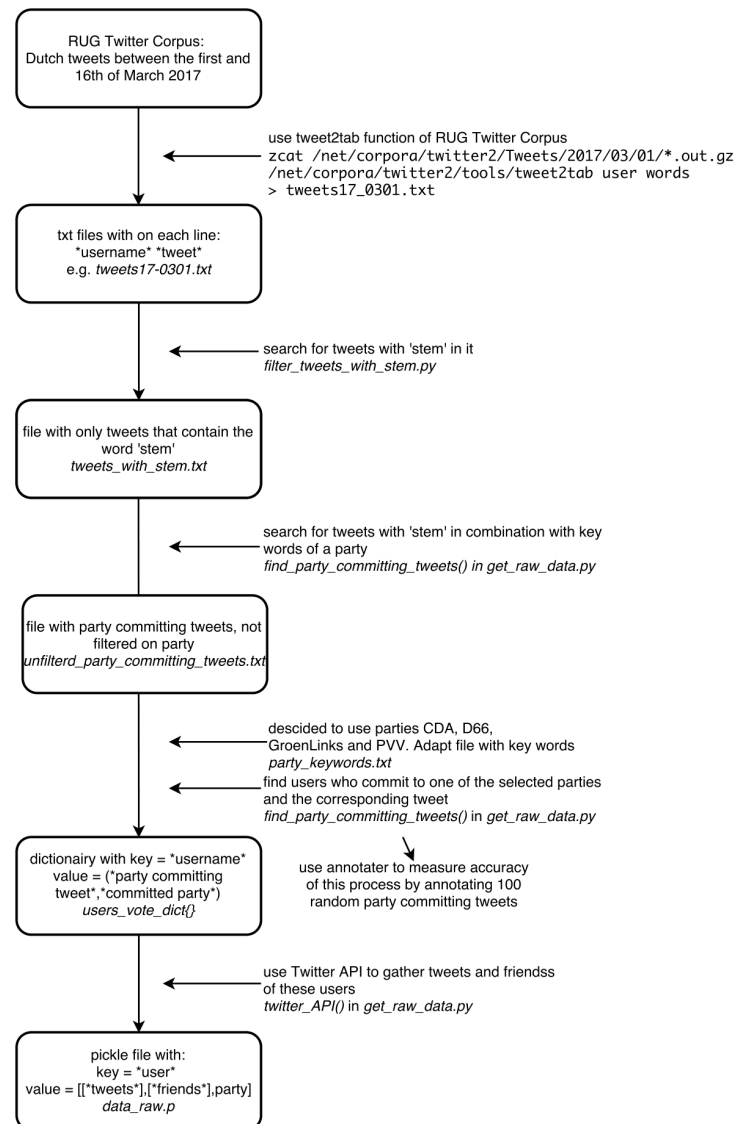


Figure 4: Flowchart of data collection

ik heb gestemd op @groenlinks op sophie schers ! tijd voor verandering en daarmee ook transgender representatie in de kamer #ikstem
 i have voted for @groenlinks for sophie schers ! time for change and with that also transgender representation in the house [of representatives] #ivote
 rt @pvvstrijders : het wordt tijd dat wij intolerant gaan worden tegen intolerantie ! #stoperdogan #stopislam #pvv #stempvv #tk2017
 rt @pvvstrijders : it's time that we become intolerant against intolerance ! #stoperdogan #stopislam #pvv #votevv #tk2017

Figure 5: Example of tweets which were annotated as having the correct label assigned to.

@markrutte als het populisme inderdaad is gestopt wordt het tijd verschillen te verkleinen ! #ikstem @jesseklaver @apechtold @sybrandbuma
 @markrutte if populism has indeed been stopt it's time to reduce the differences ! #ivote @jesseklaver @apechtold @sybrandbuma

rt @ewdevlieger : goed dat u vier jaar geleden geen premier was meneer @apechtold een vooruitziende blik van nul , komma , nul #stemd66
 rt @ewdevlieger : good you weren't prime minister four years ago mr. @apechtold a foresighted view of zero , comma , zero #voted66

Figure 6: Example of tweets which were annotated as having a wrong label assigned to

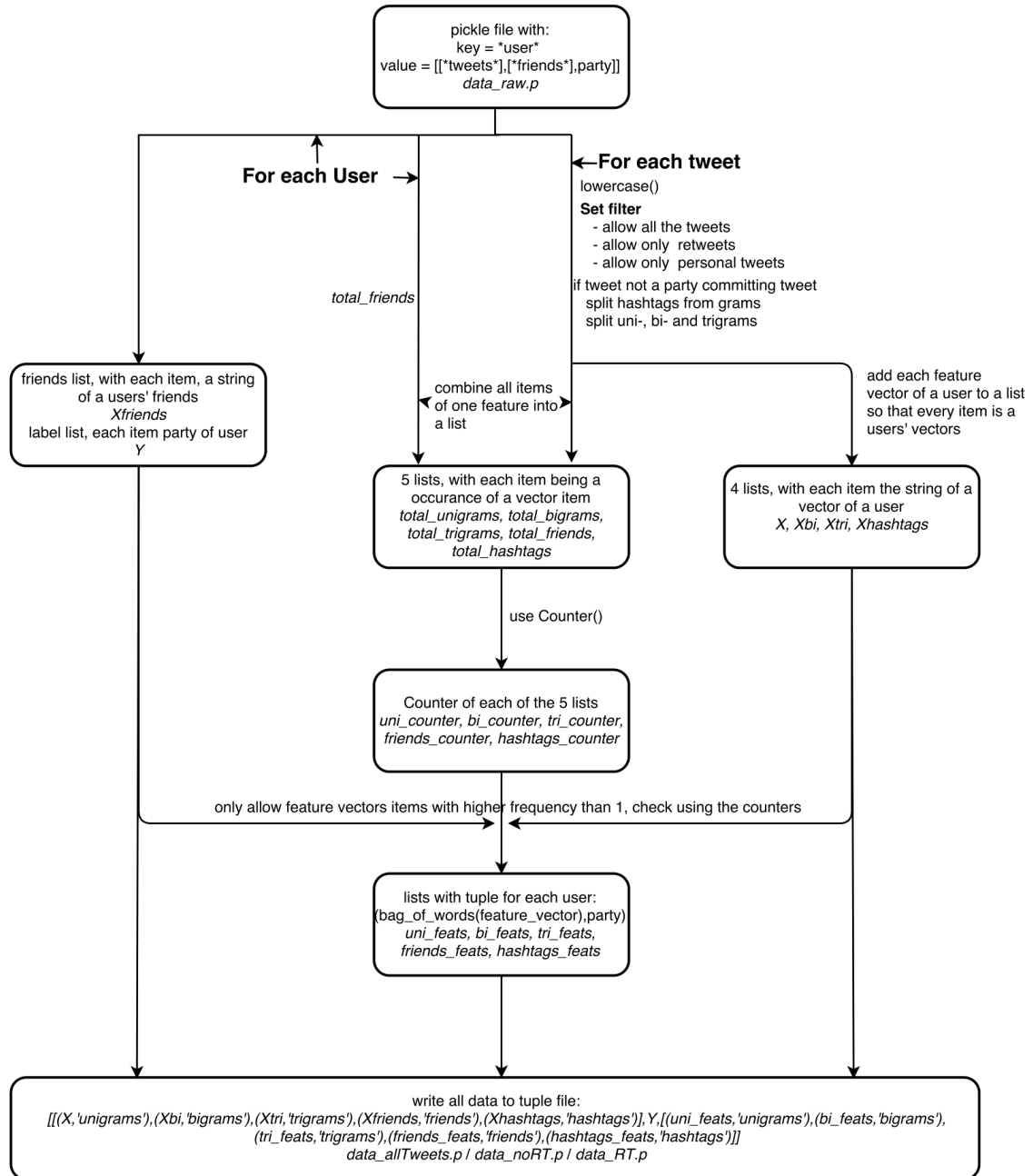


Figure 7: Flowchart of processing the data.

```
bag_of_words(['the', 'quick', 'brown', 'fox'])  
{'quick': True, 'brown': True, 'the': True, 'fox': True}
```

Figure 8: Example of Toral's `bag_of_words` function.

B | METHOD

	Naive Bayes		SVM	
	count	td-idf	count	td-idf
Unigrams	0.75	0.61	0.81	0.85
Bigrams	0.78	0.75	0.79	0.83
Trigrams	0.76	0.75	0.79	0.78
Friends	0.83	0.83	0.77	0.88
Hashtags	0.86	0.83	0.88	0.86
Average	0.80	0.75	0.81	0.84

Table 4: Accuracy scores for all tweets in development set

@martinbosma_pvv__ : = True	PVV : CDA =	67.6 : 1.0
rt__@martinbosma_pvv = True	PVV : CDA =	67.6 : 1.0
rt__@blondemevrouw1 = True	PVV : CDA =	64.8 : 1.0
@blondemevrouw1__ : = True	PVV : CDA =	64.8 : 1.0
@pieteromtzig__ : = True	CDA : GroenL =	62.6 : 1.0
rt__@groenlinks = True	GroenL : CDA =	61.9 : 1.0
@groenlinks__ : = True	GroenL : CDA =	61.9 : 1.0
@frenkie4allll__ : = True	PVV : CDA =	61.9 : 1.0
rt__@frenkie4allll = True	PVV : CDA =	61.9 : 1.0
rt__@pieteromtzig = True	CDA : GroenL =	60.7 : 1.0
@lisahenegauwen__ : = True	PVV : CDA =	58.3 : 1.0
rt__@lisahenegauwen = True	PVV : CDA =	58.3 : 1.0
rt__@jawcjanssens = True	CDA : PVV =	55.6 : 1.0
@jawcjanssens__ : = True	CDA : PVV =	55.6 : 1.0
rt__@geertwilderspvpv = True	PVV : GroenL =	55.4 : 1.0
@apechtold__ : = True	D66 : PVV =	53.3 : 1.0
rt__@cdavandaag = True	CDA : D66 =	53.2 : 1.0
rt__@percolator_hnj = True	PVV : GroenL =	51.5 : 1.0

Figure 9: Top 20 of the output of the most_informative_features() on bigrams on development set, with personal and retweets.

C | RESULTS

Table 5: Numbers of feature vectors

		Personal	Retweets	All Tweets
Unigrams	Total	4.839.825	8.382.136	13.221.961
	Unique	8.0%	5.1%	5.3%
Bigrams	Total	4.427.313	7.780.736	12.208.049
	Unique	37.2%	23.2%	24.7%
Trigrams	Total	4.057.121	723.5447	11.292.568
	Unique	72.5%	45.9%	51.8%
Friends	Total	664.770	664.770	664.770
	Unique	39.0%	39.0%	39.0%
Hashtags	Total	167.089	288.854	455.943
	Unique	26.9%	17.1%	17.5%

		Naive Bayes			SVM			Support
		Prec	Recall	f1	Prec	Recall	f1	
Unigrams	CDA	0.67	0.77	0.72	0.88	0.81	0.84	97
	D66	0.73	0.43	0.54	0.80	0.62	0.70	102
	GL	0.58	0.79	0.67	0.73	0.91	0.81	90
	PVV	0.87	0.83	0.85	0.85	0.90	0.87	111
	Average	0.72	0.70	0.70	0.81	0.81	0.81	100
Bigrams	CDA	0.80	0.77	0.79	0.86	0.79	0.82	97
	D66	0.80	0.46	0.58	0.84	0.55	0.66	102
	GL	0.57	0.92	0.70	0.69	0.91	0.78	90
	PVV	0.88	0.80	0.84	0.81	0.90	0.85	111
	Average	0.77	0.73	.73	0.80	0.79	0.78	100
Trigrams	CDA	0.85	0.75	0.80	0.82	0.74	0.78	97
	D66	0.81	0.45	0.58	0.79	0.55	0.65	102
	GL	0.53	0.96	0.68	0.68	0.87	0.76	90
	PVV	0.88	0.75	0.81	0.76	0.86	0.81	111
	Average	0.78	0.75	0.81	0.76	0.76	0.75	100
Friends	CDA	0.96	0.88	0.91	0.96	0.91	0.93	97
	D66	0.82	0.75	0.78	0.87	0.77	0.82	102
	GL	0.81	0.92	0.86	0.80	0.94	0.87	90
	PVV	0.89	0.92	0.90	0.91	0.91	0.91	111
	Average	0.87	0.86	0.86	0.89	0.88	0.88	100
Hashtags	CDA	0.84	0.72	0.78	0.84	0.67	0.75	97
	D66	0.64	0.53	0.58	0.65	0.55	0.60	102
	GL	0.41	0.68	0.51	0.71	0.67	0.69	90
	PVV	0.69	0.53	0.60	0.65	0.90	0.76	111
	Average	0.65	0.61	0.62	0.71	0.70	0.70	100

Table 6: Result table of classification of personal tweets

		Naive Bayes			SVM			Support
		Prec	Recall	f1	Prec	Recall	f1	
Unigrams	CDA	0.91	0.84	0.87	0.94	0.88	0.91	97
	D66	0.84	0.66	0.74	0.91	0.76	0.83	102
	GL	0.80	0.91	0.85	0.80	0.97	0.87	90
	PVV	0.84	0.96	0.90	0.92	0.95	0.94	111
	Average	0.85	0.84	0.84	0.90	0.89	0.89	100
Bigrams	CDA	0.94	0.82	0.88	0.98	0.87	0.92	97
	D66	0.85	0.71	0.77	0.84	0.78	0.81	102
	GL	0.82	0.93	0.87	0.81	0.92	0.86	90
	PVV	0.84	0.97	0.90	0.91	0.96	0.94	111
	Average	0.86	0.86	0.86	0.89	0.89	0.88	100
Trigrams	CDA	0.96	0.85	0.90	1.00	0.86	0.92	97
	D66	0.83	0.74	0.78	0.82	0.78	0.80	102
	GL	0.86	0.90	0.88	0.81	0.91	0.86	90
	PVV	0.83	0.98	0.90	0.89	0.95	0.92	111
	Average	0.87	0.87	0.87	0.88	0.88	0.88	100
Friends	CDA	0.96	0.88	0.91	0.96	0.91	0.93	97
	D66	0.82	0.75	0.78	0.87	0.77	0.82	102
	GL	0.81	0.92	0.86	0.80	0.94	0.87	90
	PVV	0.89	0.92	0.90	0.91	0.91	0.91	111
	Average	0.87	0.86	0.86	0.89	0.88	0.88	100
Hashtags	CDA	0.95	0.84	0.89	0.94	0.86	0.90	97
	D66	0.81	0.71	0.75	0.83	0.75	0.79	102
	GL	0.83	0.88	0.85	0.77	0.97	0.86	90
	PVV	0.82	0.96	0.88	0.93	0.89	0.91	111
	Average	0.85	0.85	0.85	0.87	0.86	0.87	100

Table 7: Result table of classification of all tweets

		Naive Bayes			SVM			Support
		Prec	Recall	f1	Prec	Recall	f1	
Unigrams	CDA	0.94	0.85	0.89	0.98	0.88	0.92	97
	D66	0.81	0.75	0.78	0.91	0.81	0.86	102
	GL	0.86	0.84	0.85	0.81	0.99	0.89	90
	PVV	0.79	0.94	0.86	0.93	0.94	0.93	111
	Average	0.85	0.84	0.84	0.91	0.90	0.90	100
Bigrams	CDA	0.98	0.85	0.91	0.98	0.85	0.91	97
	D66	0.78	0.78	0.78	0.85	0.82	0.84	102
	GL	0.90	0.84	0.87	0.83	0.93	0.88	90
	PVV	0.81	0.95	0.87	0.91	0.95	0.93	111
	Average	0.86	0.86	0.86	0.89	0.89	0.89	100
Trigrams	CDA	0.98	0.85	0.91	0.99	0.88	0.93	97
	D66	0.77	0.80	0.78	0.82	0.82	0.82	102
	GL	0.93	0.84	0.88	0.83	0.89	0.86	90
	PVV	0.83	0.95	0.88	0.89	0.93	0.91	111
	Average	0.87	0.86	0.86	0.88	0.88	0.88	100
Friends	CDA	0.96	0.88	0.91	0.96	0.91	0.93	97
	D66	0.82	0.75	0.78	0.87	0.77	0.82	102
	GL	0.81	0.92	0.86	0.80	0.94	0.87	90
	PVV	0.89	0.92	0.90	0.91	0.91	0.91	111
	Average	0.87	0.86	0.86	0.89	0.88	0.88	100
Hashtags	CDA	0.94	0.85	0.89	0.94	0.85	0.89	97
	D66	0.83	0.75	0.78	0.86	0.76	0.81	102
	GL	0.80	0.81	0.81	0.75	0.96	0.84	90
	PVV	0.81	0.95	0.87	0.93	0.90	0.92	111
	Average	0.84	0.84	0.84	0.87	0.86	0.87	100

Table 8: Result table of classification of retweets

#stemvoorverandering = True	GroenL : D66 = 196.4 : 1.0
#stemd66 = True	D66 : GroenL = 131.4 : 1.0
#cdacongres = True	CDA : D66 = 79.9 : 1.0
#teamcda = True	CDA : PVV = 78.4 : 1.0
#tijdvoorverandering = True	GroenL : CDA = 72.4 : 1.0
#stempvv = True	PVV : CDA = 59.5 : 1.0
#d66congres = True	D66 : GroenL = 57.1 : 1.0
#nexit = True	PVV : CDA = 55.9 : 1.0
#ingesprekmet = True	D66 : CDA = 52.0 : 1.0
#stemcda = True	CDA : PVV = 43.0 : 1.0
#beternederland = True	CDA : D66 = 38.1 : 1.0
#onderwijs = True	D66 : PVV = 36.0 : 1.0
#turken = True	PVV : CDA = 32.2 : 1.0
#nl doet = True	CDA : PVV = 31.8 : 1.0
#nietschreeuwenmaarpraten = True	GroenL : CDA = 31.3 : 1.0
#klimaat = True	GroenL : PVV = 30.9 : 1.0
#islam = True	PVV : CDA = 29.5 : 1.0
#demeetup = True	GroenL : CDA = 29.4 : 1.0
#nederlandweervanons = True	PVV : GroenL = 26.3 : 1.0
#meetup = True	GroenL : CDA = 24.8 : 1.0
#bumor = True	CDA : GroenL = 20.5 : 1.0
#slotdebat = True	GroenL : D66 = 19.2 : 1.0
#vrijwilligers = True	CDA : GroenL = 18.5 : 1.0
#wietwet = True	D66 : GroenL = 17.9 : 1.0
#vnl = True	PVV : GroenL = 17.4 : 1.0
#widm = True	GroenL : PVV = 16.9 : 1.0
#turkse = True	PVV : GroenL = 16.7 : 1.0
#glcongres = True	GroenL : PVV = 16.6 : 1.0
#twibbon = True	D66 : PVV = 16.2 : 1.0
#moslims = True	PVV : GroenL = 16.0 : 1.0
#pvvâ = True	PVV : CDA = 15.9 : 1.0
#zinin = True	CDA : PVV = 15.6 : 1.0
#kiesklimaat = True	GroenL : CDA = 15.5 : 1.0
#cdaâ = True	CDA : PVV = 15.4 : 1.0
#moltalk = True	GroenL : PVV = 15.3 : 1.0
#fd = True	D66 : PVV = 15.2 : 1.0
#pvv2017 = True	PVV : GroenL = 14.7 : 1.0
#markrutte = True	PVV : CDA = 14.6 : 1.0
#geenpeil = True	PVV : CDA = 14.6 : 1.0
#worldtotrump = True	GroenL : CDA = 13.6 : 1.0
#nijmegen = True	GroenL : PVV = 13.4 : 1.0
#ceta = True	GroenL : PVV = 12.7 : 1.0
#d66 = True	D66 : GroenL = 12.4 : 1.0
#klimaatverandering = True	GroenL : CDA = 12.2 : 1.0
#jessemeetup = True	GroenL : D66 = 12.2 : 1.0
#zwolle = True	CDA : PVV = 12.1 : 1.0
#grenzendicht = True	PVV : D66 = 11.9 : 1.0
#ikdoemee = True	CDA : GroenL = 11.8 : 1.0
#paay = True	PVV : GroenL = 11.3 : 1.0
#vrouw = True	PVV : D66 = 11.1 : 1.0

Figure 10: Output of `most_informative_features()` on hashtags in data set with only personal tweets.

JoostNiemoller = True	PVV : GroenL =	132.7 : 1.0
V_of_Europe = True	PVV : CDA =	108.0 : 1.0
Esther_de_Lange = True	CDA : GroenL =	107.2 : 1.0
JAWCJanssens = True	CDA : PVV =	106.0 : 1.0
StuivenbergInfo = True	PVV : GroenL =	101.3 : 1.0
AgnesMulderCDA = True	CDA : PVV =	98.1 : 1.0
RealDutchRoots = True	PVV : GroenL =	91.1 : 1.0
RobertKnijff = True	PVV : GroenL =	91.1 : 1.0
PeterSweden7 = True	PVV : CDA =	87.7 : 1.0
frenkie4allll = True	PVV : GroenL =	87.0 : 1.0
LettyDemmers = True	D66 : GroenL =	82.1 : 1.0
waziman15 = True	PVV : CDA =	81.6 : 1.0
MartijnCDA = True	CDA : PVV =	80.4 : 1.0
RijnPatrick = True	PVV : GroenL =	80.2 : 1.0
ruthpeetoom = True	CDA : GroenL =	78.8 : 1.0
d_e_mol = True	PVV : GroenL =	77.5 : 1.0
rikgrashoff = True	GroenL : PVV =	75.9 : 1.0
VictorKortijjs = True	PVV : CDA =	75.5 : 1.0
tRadicaleMidden = True	D66 : CDA =	75.4 : 1.0
Hannesz1956 = True	PVV : CDA =	74.8 : 1.0
Veerman_Edward = True	PVV : GroenL =	74.7 : 1.0
michildijkman = True	CDA : PVV =	74.5 : 1.0
Fub_Fub = True	PVV : GroenL =	74.0 : 1.0
JanTerlouw = True	D66 : PVV =	73.6 : 1.0
2dekamerpvv = True	PVV : GroenL =	72.0 : 1.0
cdaeuropa = True	CDA : GroenL =	70.2 : 1.0
Mirjam152 = True	PVV : CDA =	68.1 : 1.0
SalimaBelhaj = True	D66 : PVV =	67.6 : 1.0
truusdemier = True	PVV : GroenL =	67.2 : 1.0
hulswoud = True	PVV : CDA =	66.4 : 1.0
ErikRennes = True	CDA : GroenL =	66.2 : 1.0
ovcned = True	PVV : D66 =	65.0 : 1.0
Blondemevrouw1 = True	PVV : CDA =	64.8 : 1.0
neeRUTTEnee = True	PVV : CDA =	64.7 : 1.0
charlottel01987 = True	PVV : CDA =	64.7 : 1.0
MLP_officiel = True	PVV : CDA =	64.0 : 1.0
horentastenzien = True	PVV : GroenL =	63.8 : 1.0
jpaternotte = True	D66 : PVV =	63.7 : 1.0
SybrenKooistra = True	GroenL : CDA =	63.2 : 1.0
DFranksen = True	PVV : GroenL =	63.1 : 1.0
PVVstrijders = True	PVV : CDA =	62.6 : 1.0
SamvanRooy1 = True	PVV : CDA =	61.3 : 1.0
Jan_Pijnenburg = True	PVV : GroenL =	61.2 : 1.0
Little_one_Ann = True	PVV : GroenL =	60.4 : 1.0
HitradioNL = True	PVV : GroenL =	60.4 : 1.0
VMStichting = True	D66 : GroenL =	59.8 : 1.0
consumentnieuws = True	PVV : GroenL =	59.7 : 1.0
Real_Pantheist = True	PVV : GroenL =	59.7 : 1.0
destembus2017 = True	GroenL : PVV =	58.9 : 1.0
Wegaandiep = True	PVV : CDA =	58.6 : 1.0

Figure 11: Output of most_informative_features() on friends.