Technische Universität München

Department of Mathematics

Information retrieval and cluster recognition of textual data obtained from Twitter using principal component analysis

Master's Thesis

by

Robert Hager

Supervisor:	Professor Claudia Czado, Ph.D.
Advisor:	Professor Claudia Czado, Ph.D. & Dominik Müller
Submission date:	June 24, 2015

I hereby declare that this thesis is my own work and no other sources have been used except those clearly indicated and referenced.

Garching, June 24, 2015

Acknowledgments

First and foremost I want to thank Professor Claudia Czado for giving me the opportunity to work on this very topic. Every discussion with her steered my work in the right direction and the given feedback and input on my already existing approaches always challenged me and improved my efforts as a result.
Moreover I want to especially thank my advisor Dominik Müller who was all time available for my questions and concerns and steadily helped to improve my thesis.
Lastly and most importantly, I want to thank my family, my girlfriend and my closest circle of friends for always supporting me and the decisions I made. Without you the difficult times would have been even more difficult and because of you I always see the positive aspects of life which definitely helped me to achieve my aims.

Abstract

The penetration of the internet and the social media platforms into the daily life of many people in the world reached a new summit in recent years. Therefore it is crucial for science, as well as for corporate use, to structure the huge amount of created data to gain valuable information. One of the most well-known and wide spread online platforms is the short message service Twitter. In order to cluster the extracted information from this data source and detect trends and former hidden patterns several approaches are investigated. Firstly the data source Twitter is introduced and shown how it can be exploited. Subsequently the basic definitions and the theory of principal component analysis, necessary to execute the investigation and the structuring of the data, is described. To demonstrate the developed methods in practical relevant cases two applications are introduced. In the first application we examine Twitter data treating the Islamic State and identify its clusters with the help of the leading principal components and a continuous assessment process of new data is set up. The results show that the developed algorithm recognizes meaningful and logical clusters and in addition classifies new data as expected. The second application briefly examines the dependency between appearances on Twitter and the respective development in the real world, concerning Ebola.

Zusammenfassung

Die weltweite Durchdringung und der Einfluss des Internets im Allgemeinen und von sozialen Netzwerken im Speziellen auf das Leben vieler Menschen, hat in den letzten Jahren einen neuen Höhepunkt erreicht. Die enormen, durch die permanente Nutzung entstehenden, Datenmengen sind sowohl für die Wissenschaft als auch für wirtschaftliche Unternehmen von höchster Relevanz. Um eine geeignete Auswertung zu ermöglichen, ist es unerlässlich die Daten zu filtern und zu strukturieren. Der Kurznachrichtendienst Twitter ist hierbei eine der bekanntesten und am weitesten verbreiteten Onlinekommunikationsplattformen. Die nachfolgende Arbeit beleuchtet zwei alternative Herangehensweisen um die Daten dieser Datenquelle zu gruppieren und Trends und versteckte Muster zu erkennen. Nach einer kurzen Einführung zur Struktur und Wirkungsweise von Twitter, wird erklärt wie daraus relevante Daten gewonnen werden. Anschließend werden die wichtigsten Definitionen und die Theorie der Hauptkomponentenanalyse eingeführt, welche die Grundlage für die Untersuchung und Datenstrukturierung darstellt. Zwei Anwendungen werden präsentiert um die erarbeiteten Vorgehensweisen zu veranschaulichen. In der ersten Anwendung werden aus Twitter Daten, welche das Thema „Islamischer Staat" behandeln, mit Hilfe der führenden Hauptkomponenten Cluster identifiziert und ein stetiger Zuordnungsprozess für neu auftretende Daten entwickelt. Die Resultate belegen hierbei, dass der entwickelte Algorithmus sinnvolle und logische Cluster entdeckt und neu auftretende Datensätze zuverlässig klassifiziert. Abschließend wird in der zweiten Anwendung im praxisrelevanten Fall die Abhängigkeit des Auftretens des Begriffes „Ebola" und damit verwandte Textbausteine mit der Verbreitung der Krankheit in der realen Welt untersucht.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

One of the most prominent expressions in the last years as well as in the public perception, as in the academic world and especially in the corporate context is the term Big Data. It arose as a consequence of the developments in the last years. These include the rising spread and use of the internet, low cost and big capacities of storage, the social media platforms and the growing mobile phone penetration. Almost every person in the developed countries uses the internet very frequently at home, at work or on their smart phone, which means all time availability at every location, revealing huge amounts of personal information. This leads to the massive accumulation of 2.5 Exabyte, which corresponds 2.5 million gigabytes of data per day as of the year 2012 [1].

In order to gain insight in this unstructured data we have to process it in a meaningful way so we can gain valuable information for research or for corporate use. For this purpose we want to detect factors which explain most of the textual data and dismiss the noise contained in every big and subjective data source. Among many others one of the most interesting sources for this kind of information is the social media platform Twitter, as it is a very wide spread social media platform on which numerous people around the world express their attitude, opinions and information regarding every imaginable topic. One goal is to cluster the information, contained in the short messages of Twitter automatically and continuously in order to detect patterns, developments and trends in the public opinion regarding certain topics, as simple, intuitive and logical as possible. Another intention is the prediction of certain developments in the real world, by comparing them with the reaction and behaviour of Twitter users regarding these events. The applications have a very broad range from meteorology to disease spread prediction and personalized advertisement to name just a few. For example [Generous (2014)] uses Wikipedia access logs in order to predict several diseases and [Paul (2014)] already uses Twitter as a data source to analyze developments of public health in the United States of America. The aim of this work is to give an overview of the different possibilities of analyzing, visualizing and structuring huge amounts of textual data, as well as introduce intelligent and sophisticated algorithms in order to assess the contained hidden patterns.

Therefore this thesis explains our data source and how to receive specific topic related short messages, so called tweets in Chapter 2. Then, the treatment of the extracted textual data and their transferral into a mathematical form will be stated in Chapter 3. Chapter 4 introduces the theory of the Principal Component Analysis and the underlying basics in order to detect hidden patterns and cluster in the data. In the last chapter we will state two applications, which are supposed to give an overview of how different topics obtained from Twitter can be examined and structured and how we can gain information from this huge subjective data source. Firstly, we analyze data treating the fundamentalist group of the Islamic State. We therefore examine time patterns, inspect times of high activity and the development of the most important terms. Finally we group the data into informative clusters with the help of the new factors obtained from the Principal Component Analysis and implement an algorithm which automatically allocates new data to the detected clusters. Secondly we explore Ebola related messages and verify them against official data, aiming to observe correlations and therefore predictions for the spread of this disease.

Chapter 2

Twitter

First of all, in order to understand the source of our data and the terminology, we will give a brief introduction to Twitter.

2.1 History

Twitter is an online social networking service, which was launched in 2006 and rapidly gained worldwide popularity since then. Nowadays it is not only known by internet affine people, but used and cited in newspapers, television shows and the daily life of many people. It is also not only a platform where people express their personal issues and beliefs or communicate and express theirselves, but prominent politicians as Barack Obama, as well as pop stars like Katy Perry, associations like UNICEF and enterprises, e.g. BMW among many others, use this social media platform to communicate, inform and undertake marketing to their followers or the public in general. As our daily life is more and more penetrated by smart phones, resulting in all-time availability of internet access everywhere, this way of communication got onto an even higher level in the recent years. Potentially, almost every person in the developed world can nowadays express their feelings, needs, knowledge and information in real time and without any constraints. The very successful stock market listing in September 2013 underlines the value and potential of the company seen by the financial markets, which have recognized the value of big data and personal information [2]. Obviously this bears huge opportunities for analysing certain events, behaviours and developments. Although the vast majority of Twitter users are under 35 years old and live in the western world, especially in the United States of America, see [3], one could gain significant, valid, real time and cost free analysis and predictions of certain topics by exploiting this data source.

2.2 Figures

As of July 2014, Twitter had over 270 million active users per month in average, who set off around 500 million tweets and execute around 2 billion search queries per day, see [4], [5] and [6]. This makes Twitter the top 8^{th} of the most visited websites [7]. But Twitter has also impressing offline figures with a revenue of 312 million USD just in the second quarter of 2014, which gives Twitter a rank in the Top 20 of the most valuable internet companies worldwide, see [4] and [8].

2.3 Properties and Functionality

A *tweet* is a short message consisting of a maximum of 140 characters which can be read by every person in the internet, but can only be posted by registered users. Tweets can also contain web links, pictures and videos. Every registered user has a unique user name with a precedent @. Via this user name every user can be addressed in a tweet by indexing the tweet with this unique name. Every user has a so called *timeline* which is the page of a user, on which one can see his personal information and the tweets he sent. Referring to a certain user is possible by writing @user in a tweet, therefore generating a redirect to his timeline. A *retweet* simply stands for a message of User A, User B thinks is so interesting, funny or complex he wants to share this tweet with his community. One can identify every retweet easily, as 'User A is retweeting:' is automatically written in the header of those shared messages. For example, a funny picture of some Hollywood actors during the 2013 Oscars tweeted by Ellen DeGeneres, was retweeted over 3 million times and showed up in diverse TV shows and even newspapers all over the world [9]. There is also the possibility to subscribe to certain users by *following* them. As a result you see the tweets of the users you follow on your news section. The most famous and most recognizable feature on Twitter is the #, the so called *hashtag*. It is used in the tweets as an index or reference to certain topics or keywords. If we create our own tweet, but we want to refer to an already existing topic or theme, we can include that in the message by referring to the topic via #topic. Therefore it is very interesting which hashtags, i.e. topics, arise every day and gain more popularity over time. This methodology triggers a self-reinforcing effect as more known hashtags are used more often in tweets and are also more often contained in retweets. A trending section, which shows uprising hashtags and terms to indicate topics which currently gain more influence and power in the tweets, already exists on the Twitter website. Unfortunately one can only see ten plain, unordered words without any further information or statistics.

2.4 Extraction from Twitter into R

In order to transfer our data, i.e. tweets from Twitter to R we exploit the R-package 'twitteR' [Gentry (2013)]. For that reason we use the function 'searchTwitter'. Before we call this function we have to go through an authentication process, which can be saved and then be called every time we want to fetch data from Twitter. One necessary requirement is a valid Twitter account. For more detailed information concerning this authentication process we refer to [10] and the authentication code in the Appendix B.1.

Example 2.1. *Extracting tweets from Twitter into R.*

We fetch 20 tweets of the account of the Technical University of Munich between two randomly chosen dates and have a closer look at one of them.

```
> # loading necessary library
> library(twitteR)
> # load verification standard of twitter
> load("twitteR_credentials")
> registerTwitterOAuth(twitCred)

[1] TRUE

> # obtain English tweets of the account of the TU Munich between the
> # given dates
> tum <- searchTwitter("@TU_Muenchen" , n = 20 , since = "2014-11-20" ,
+ until = "2014-11-24" , lang = "en" , cainfo =  "cacert.pem")
> # have a look at the 18th fetched tweet
> tum[[18]]$text

[1] "RT @SAP_UA: #SAP Big Data Truck comes to #Munich Nov.21 Experience
     #bigdata live at @TU_Muenchen or online http://t.co/EBG7MVLRWF"
```

After loading the authentication routine, we fetch 20 tweets of the account of the Technical University of Munich, addressed by @TU_Muenchen between November 20 and November 24, 2014 which are written in English. The parameter ‘cainfo‘ in the searchTwitter function reviews the authentication certificate one more time. By having a look at the 18th extracted tweet we see that it is a retweet, as ‘RT‘ is the first word and stands for retweet, to a tweet sent by the account of ‘SAP university alliances‘ concerning a marketing tour of SAP to Munich. We can see that the indices or references marked by the hashtags #SAP, #Munich and #bigdata already explain most of the message.
Table A.1 contains all elements of the list the ‘searchTwitter’ function returns and is given in the Appendix A.1.

2.5 Limitations and Opportunities

Two limitations arise when extracting tweets to R due to the API connection between Twitter and R. First of all it is only possible to get tweets of the last seven days. This makes it impossible to analyse arbitrary topics at an arbitrary time in the past. But we can build up a database by focusing on some keywords and extract the tweets containing them on a daily basis. By saving them every day, we get a database large enough to start analyzing those topics after some time. Secondly, if a tweet contains a non UTF-8 or non ASCII character, for example an Arabic or Cyrillic letter, the fetching routine aborts with an error and we have to work around this tweet.
Despite the above mentioned limitations it is easy to imagine what a broad range of applications could be developed through the analysis of tweets, by examining this huge data source. The extraction of tweets reveals a large amount of personal opinions, information and statements of individuals at a broad spectrum. Hidden patterns can even be found on topics were no official data is available as people may tweet things they would not tell, or would not bother to tell to officials and researchers. Due to the all-time availability of

internet through wireless networks, mobile internet and smart phones we are able to get a steady stream of information at neglible costs.

When talking about automatized extraction of personal data from social media platforms, the issue of privacy rights emerges automatically. This subject draws high attention especially in Germany in general and even more since the revelations of the NSA spy scandal. Although our approach is totally legal and every user agrees on the use of his data in the terms of service [11], we have to be aware that we are using private and maybe even sensitive data. Also moralist concerns have to be taken into account. As we obscure personal data and also do not use it for commercial use or even adjust actions, like marketing approaches to individual users, this should not be a problem.

So, Twitter is definitely a valuable data source worth examining in order to get interesting insights in peoples thoughts and discovering trending topics although we should not forget about privacy issues.

Chapter 3

The Document-Term Matrix

First of all, we have to bring the pure extracted textual data into a form it can be treated and analysed mathematically. From now on, we will say *document* instead of tweet for generalization purposes and *term* instead of word as not every character string in a tweet necessarily has to be a proper word. In the following we will convert our documents and the terms contained in them into an occurrence matrix M.

3.1 Pre-processing of raw data

At the very beginning we set up our *Corpus* which is the required data type of the R-package ‘tm‘ [Feinerer (2014)]. In our case we fill the Corpus with the actual text content of our extracted tweets. Before we have a closer look at the terms in the documents, we convert all upper case to lower case letters, as this is not an eminent point, by applying the function ‘tm_map(Corpus , tolower)‘. Furthermore we remove several unimportant characters like punctuation marks, special characters, numbers and URL's by exploiting the already implemented function ‘tm_map(Corpus , removeNumbers)‘ and regular expression routines like ‘gsub‘. Of course this procedure has to be adjusted to the special needs of every analysis. In our later applications for example, we will not exclude @ and # from the texts as they give essential information when looking at tweets. Our last step is the exclusion of several so called *stopwords*. Stopwords are words containing no important information like *the, a, and, or, then* and many more. A list of the standard English stopwords as used in the R-package ‘tm‘ is given in the Appendix A.2. We will expand this list later on for our special purposes. For example, we will also want to exclude the term ‘rt‘ which stands for retweet and just signals that the examined tweet is a retweet. This information is useless as there is the variable ‘isRetweet‘ in the extracted list containing exactly this information. For the time being we will leave the retweets in the Corpus because they stand for broader acceptance and higher spread of the message they contain. The data is then supposed to be clean and containing only meaningful terms.

3.2 Construction of the Document-Term Matrix

Suppose we have a data set of n documents and each of these documents contains m_i different terms, $i \in \{1, ..., n\}$. If we now depict this into a matrix form, we obtain a matrix M, which n rows are the documents and which $m' := m_1 + ... + m_n$ columns are all terms which appear in these documents. But suppose we have k columns which stand for the same l terms, as different documents can contain one or more equal terms. Then we combine those columns which describe the same terms, so that there is only one column left for each term. The final result are $m = m' - (k - l)$ columns, which are usually ordered alphabetically, therefore $M \in \mathbb{N}_0^{n \times m}$.

3.2.1 Term frequency - without weighting

The matrix M is now filled by a very simple and intuitive principle as shown in the following definition.

Definition 3.1: (Document-term matrix without weighting)

Suppose we have n documents containing m unique and alphabetically ordered terms t_j, $j \in \{1, ..., m\}$, corresponding to the j-th column of the matrix M. Let d_i, $i \in \{1, ..., n\}$ denote the i-th document, i.e. the i-th row of M, then with

$$m_{ij} := \begin{cases} h & \text{if} \quad t_j \in d_i \quad h\text{-times} \\ 0 & else, \end{cases}$$

$DTM := (m_{ij}) \in \mathbb{N}_0^{n \times m}$ is the *document-term matrix without weighting.*

So, if document $i \in \{1, ..., n\}$ contains term $j \in \{1, ..., m\}$ h times, the $ij-$th entry of DTM is h and zero if term j does not appear in document i. Because the overall vocabulary is going to be very large if we have many documents, but a single document only contains a few terms, the result will be a very sparse matrix as most of its entries will be zero.

3.2.2 Term-frequency inverse document-frequency weighting

Another possibility to construct the document-term matrix is to apply the so-called *term-frequency inverse document-frequency weighting (TfIdf).* This procedure takes more into account than just pure counting, which only gives information how often a certain term appears in a certain document. We will get a better insight into this approach by looking at the definition of the weighting method.

Definition 3.2: (Term-frequency inverse document-frequency)

Suppose we have n documents containing m unique and alphabetically ordered terms t_j, $j \in \{1, ..., m\}$. Let d_i, $i \in \{1, ..., n\}$ denote the i-th document and D the set of all documents.

We define the *normalized term frequency* of term j in document i as

$$tf_{ij} := \frac{m_{ij}}{\sum\limits_{k=1}^{m} m_{ik}},$$

where m_{ij} is the number of occurrences of a term t_j in document d_i.

Further, let

$$idf_j := \log_2 \left(\frac{|D|}{|\{d \in D | t_j \in d\}|} \right)$$

denote the *inverse document frequency* of term j, where $|D|$ denotes the total number of documents.

Finally, the *term-frequency inverse document-frequency* is calculated by

$$TfIdf_{ij} := tf_{ij} \cdot idf_j.$$

Then, the document-term matrix with term-frequency inverse document-frequency weighting is defined by

$$DTMw := (TfIdf_{ij}) \in \mathbb{R}^{n \times m}.$$

The normalized term frequency tf_{ij} counts the number of occurrences of term j in document i and normalizes it by the overall number of terms contained in this document. The normalization has the effect that the more terms appear in a document the less important one single term of them is. The inverse document frequency idf_j takes into account, in how many percent of all documents term j appears. So the inverse document frequency rates a term lower the more often it appears in all documents. If term t_j would appear in every document, idf_j would be zero as the term has obviously no specific information at all. The choice of the binary logarithm results from information theory as the amount of self-information and information entropy is often expressed by using this logarithm, see [Lubbe (1997)] p. 3. The term-frequency inverse document-frequency weighting $TfIdf_{ij}$ is simply the combination of tf_{ij} and idf_j and therefore unites both of the above described weighting properties. Therefore, we obtain a matrix reflecting the documents and terms in the extracted tweets and the weighted importance of the contained terms as their entries.

3.3 Example

To illustrate the above described construction of the two different document-term matrices we give a short made up example.

Example 3.1. *Constructing different document-term matrices.*

1. Set up

```
> # load necessary library
> library(tm)
> # creating made up example consisting of five fictional tweets
> # from the tu munich account
> tweets <- vector(mode = "character" , length = 5)
> tweets[1] <- "being math #student @TU_Muenchen is great"
> tweets[2] <- "#bigdata @TU_Muenchen"
> tweets[3] <- "#student record @TU_Muenchen"
> tweets[4] <- "RT: being math #student @TU_Muenchen is great"
> tweets[5] <- "writing #masterthesis @TU_Muenchen about #bigdata"
```

2. Building the Corpus

In the next steps we resolve the texts to get clean ones containing only informative terms.

```
> # building a corpus containing the tweet texts
> Corpus <- Corpus(VectorSource(tweets))
> # convert to lower case
> Corpus <- tm_map(Corpus , tolower)
> # remove punctuation but not # and @
> removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
> Corpus <- tm_map(Corpus , removesomepunct)
> # remove numbers
> Corpus <- tm_map(Corpus , removeNumbers)
> # remove URLs
> removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
> Corpus <- tm_map(Corpus , removeURL)
> # remove stopwords adjusted where necessary
> # (including rt which stands for retweet)
> myStopwords <- c(stopwords("english") , "rt")
> Corpus <- tm_map(Corpus , removeWords , myStopwords)
> # inspect corpus
> inspect(Corpus)
```

The resulting cleaned Corpus now looks as follows:

```
A corpus with 5 text documents
```

```
The metadata consists of 2 tag-value pairs and a data frame
Available tags are:
  create_date creator
Available variables in the data frame are:
  MetaID

[[1]]
 math #student @tumuenchen  great

[[2]]
#bigdata @tumuenchen

[[3]]
#student record @tumuenchen

[[4]]
  math #student @tumuenchen  great

[[5]]
writing #masterthesis @tumuenchen  #bigdata
```

3. Constructing document-term matrix

In this case, we have $n = 5$ documents and $m_1 = 4$, $m_2 = 2$, $m_3 = 3$, $m_4 = 4$, $m_5 = 4$ resulting in $m' = 17$. Because *@tumuenchen* appears in every document, *#student* in three documents and *math*, *great* and *#bigdata* in two documents, $k = 14$ and $l = 5$. Therefore, we have $m = 17 - (14 - 5) = 8$ unique terms resulting in $M \in \mathbb{N}_0^{5\times 8}$. Now, we can construct a document-term matrix as described above, once without weighting and once with TfIdf weighting. The calculation is done by the function 'DocumentTermMatrix(Corpus , weighting = ...)'.

```
> # building a term document matrix without weighting
> DTM <- DocumentTermMatrix(Corpus , control =
+                                        list(wordLengths = c(2 , Inf)))
> DTM <- as.matrix(DTM)
> # building a term document matrix with TfIdf weighting
> DTMw <- DocumentTermMatrix(Corpus , control =
+              list(wordLengths = c(2 , Inf) , weighting = weightTfIdf))
> DTMw
```

```
A document-term matrix (5 documents, 8 terms)

Non-/sparse entries: 12/28
Sparsity           : 70%
Maximal term length: 13
Weighting          : term frequency - inverse document frequency
                                      (normalized) (tf-idf)
```

```
> DTMw <- as.matrix(DTMw)
```

The resulting document-term matrix without weighting is given below:

```
> show(DTM)
Docs/Terms  #bigdata  #masterthesis  #student  @tumuenchen  great  math  record  writing
    1          0            0            1          1          1     1      0        0
    2          1            0            0          1          0     0      0        0
    3          0            0            1          1          0     0      1        0
    4          0            0            1          1          1     1      0        0
    5          1            1            0          1          0     0      0        1
```

From this basic document-term matrix one can easily see which terms are in which document or in which documents a certain term appears vice versa. For example, we see that the term *#student* appears in document 1, 3 and 4 and that document 2 contains the terms *#bigdata* and *@tumuenchen*. For comparison, we now examine the document-term matrix with TfIdf weighting.

```
> options(digits = 3)
> show(DTMw)
Docs/Terms  #bigdata  #masterthesis  #student  @tumuenchen  great  math   record  writing
    1        0.000       0.000        0.184        0       0.331  0.331  0.000   0.000
    2        0.661       0.000        0.000        0       0.000  0.000  0.000   0.000
    3        0.000       0.000        0.246        0       0.000  0.000  0.774   0.000
    4        0.000       0.000        0.184        0       0.331  0.331  0.000   0.000
    5        0.331       0.581        0.000        0       0.000  0.000  0.000   0.581
```

An interpretation of this document-term matrix is not as easy as above as this matrix does not just count occurrences but puts a weighting on it. Larger weights now mean higher importance of this term for this document. In other words, it shows how important a specific term is for describing a specific document. As we can see the term *@tumuenchen* has only zero entries due to the fact that it appears in every document because it was our search term and is therefore irrelevant for explaining the meanings of the documents. On the contrary, the term *record* has the highest weight of all in document 3, as this term only appears in this document and also is one of only three terms it contains. Referring to Definition 3.2, this particular weight is calculated by

$$TfIdf_{37} := tf_{37} \cdot idf_7 = \frac{m_{37}}{\sum_{k=1}^{8} m_{3k}} \cdot \log_2 \left(\frac{|D|}{|\left\{d \in D | t_7 \in d\right\}|} \right) = \frac{1}{3} \cdot \log_2 \left(\frac{5}{1} \right) \approx 0.774,$$

indicating the importance of this term for this document.

If we look for example at the term *#student* in document one we get a lower weight as this term is just one out of four words in this document and also appears in the documents 3 and 4.

$$TfIdf_{13} := tf_{13} \cdot idf_3 = \frac{m_{13}}{\sum_{k=1}^{8} m_{1k}} \cdot \log_2 \left(\frac{|D|}{|\left\{d \in D | t_3 \in d\right\}|} \right) = \frac{1}{4} \cdot \log_2 \left(\frac{5}{3} \right) \approx 0.184.$$

We have seen how to depict textual data into a numeric document-term matrix and can now mathematically analyse it.

Chapter 4

Principal Component Analysis

Assume we extracted n tweets from Twitter each containing a specific keyword. In this case we have n observations and m variables, i.e. all the terms contained in the tweets. After applying the steps described in Chapter 3, we obtain a document-term matrix $Y \in \mathbb{R}^{n \times m}$. In order to structure the given data, construct new variables, reduce their number and discover hidden patterns we want to apply Principal Component Analysis. The underlying model, assumptions and theoretical background as well as the approach itself is described in the following.

4.1 Foundations

First of all we state some definitions which will be used later on.

Definition 4.1: (Expected value)

Let X_1 be a random variable on the probability space $(\Omega, \mathcal{F}, P)$ then

$$\mathbb{E}(X_1) = \mu_1 := \int_\Omega X_1(\omega) P(d\omega)$$

is called *expected value* of X_1 if the Lebesgue integral exists.

Definition 4.2: (Variance and Covariance)

Let X_1 be a random variable, then the *variance* of X_1 is calculated by

$$Var(X_1) = \sigma_1^2 := \mathbb{E}((X_1 - \mathbb{E}(X_1))^2).$$

Let X_2 be another random variable, the *covariance* between X_1 and X_2 is then calculated via

$$Cov(X_1, X_2) = \mathbb{E}((X_1 - \mathbb{E}(X_1))(X_2 - \mathbb{E}(X_2))).$$

Furthermore, let

$$\Sigma := \begin{pmatrix} Cov(X_1, X_1) = \sigma_1^2 & ... & Cov(X_1, X_m) \\ . & & . \\ . & & . \\ . & & . \\ Cov(X_m, X_1) & ... & Cov(X_m, X_m) = \sigma_m^2 \end{pmatrix} \in \mathbb{R}^{m \times m}$$

denote the *covariance matrix* of the random variables $X_1, ..., X_m$, $m \in \mathbb{N}$.

The two precedent definitions are analogous to [Klenke (2006)], Chapter 5.

Definition 4.3: (Skewness and Kurtosis)

Let X_1 be a random variable with $\mu_1 = \mathbb{E}(X_1)$ and $\sigma_1^2 = Var(X_1)$, then

$$Skew(X_1) := \mathbb{E}\left(\left(\frac{X_1 - \mu_1}{\sigma_1}\right)^3\right)$$

denotes the *skewness* of X_1 and

$$Kurt(X_1) := \frac{\mathbb{E}\left((X_1 - \mu_1)^4\right)}{\left(\mathbb{E}\left((X_1 - \mu_1)^2\right)\right)^2}$$

the *kurtosis* of X_1, see [Bai (2005)].

Definition 4.4: (Multivariate Skewness and Kurtosis)

Let $\mathbf{X} = (X_1, ..., X_m)'$ be a random vector with mean vector $\boldsymbol{\mu} = (\mu_1, ..., \mu_m)'$ and covariance matrix Σ. Furthermore let $\mathbf{Y}$ be a random vector which has the same distribution as $\mathbf{X}$ but is independent from $\mathbf{X}$, then

$$\beta_{1,m} := \mathbb{E}(((\mathbf{X} - \mu)'\Sigma^{-1}(\mathbf{Y} - \mu))^3)$$

denotes the *multivariate extension of the skewness* of $\mathbf{X}$ and

$$\beta_{2,m} := \mathbb{E}(((\mathbf{X} - \mu)'\Sigma^{-1}(\mathbf{X} - \mu))^2)$$

the *multivariate extension of the kurtosis* of $\mathbf{X}$, see [Mardia (1974)].

If $\boldsymbol{X}$ follows a symmetric distribution, for example a normal distribution, the multivariate extension of the skewness $\beta_{1,m}$ is zero. The multivariate extension of the kurtosis $\beta_{2,m}$ of $\mathbf{X}$ is $m(m+2)$ under the assumption of multivariate normal distribution, see [Mardia (1970)] p. 527.

Definition 4.5: (Empirical distribution function)

Let X_1 be a random variable. Furthermore let $x_{11}, ..., x_{1n}$ be n observed realizations of X_1. Then we denote

$$F_{X_{1_n}}(x) := \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\{x \leq x_{1i}\}}(x),$$

as the *empirical distribution function*, see [Klenke (2006)] p. 111. $\mathbb{1}_{\{x \leq x_{1i}\}}(x)$ stands for the indicator function which is 1 if $x \leq x_{1i}$ and zero else.

Definition 4.6: (Median)

The *median* m of a data sample $x_1, ..., x_n$, $n \in \mathbb{N}$ is the number which divides the sample such that,

$$P(x_i \leq m) \geq 0.5 \text{ and } P(x_i \geq m) \leq 0.5 \quad \forall i \in \{1, ..., n\}\,.$$

Definition 4.7: (Box-Whisker-Plot and Outlier)

The *Box-Whisker-Plot* shows a box which lower bound marks the first and which upper bound marks the third quartile of the data. Additionally the median of the data is shown inside the box. The two *whiskers* of a boxplot mark the maximum, respectively the minimum of the data as long as they do not exceed 1.5 times the interquartile range of the upper, respectively the lower quartile from the median. A certain data point is called *outlier* if its value lies beyond the extremes of the whiskers.

Definition 4.8: (Eigenvalue and eigenvector)

Let $A \in \mathbb{R}^{n \times n}$. Any $\lambda \in \mathbb{R}$ satisfying

$$A\boldsymbol{t} = \lambda \boldsymbol{t}$$

for a non-zero vector $\boldsymbol{t} \in \mathbb{R}^n$ is called *eigenvalue* of A corresponding to the *eigenvector* $\boldsymbol{t}$, see [Axler (1997)] p. 77.

One can guarantee uniqueness of λ and $\boldsymbol{t}$ by normalizing the eigenvectors via

$$\boldsymbol{t}' = \frac{\boldsymbol{t}}{||\boldsymbol{t}||_2},$$

where $||.||_2$ denotes the Euclidean norm.
If $A \in \mathbb{R}^{n \times n}$ is symmetric, the eigenvectors of different eigenvalues are orthogonal to each

other and, as we normalize them, orthonormal. This is a direct consequence of the Spectral Theorem, see [Axler (1997)] p. 136.

Definition 4.9: (Rank of a matrix)

Let $A \in \mathbb{R}^{n \times m}$, the maximum number of linearly independent column vectors of A is then called the *rank* of the matrix A, $rank(A)$.

Definition 4.10: (Best linear approximation of a random variable)

Let $X_1, ..., X_m$ be random variables and $\boldsymbol{Z} = (X_1, ..., X_m) \in \mathbb{R}^m$ the corresponding random vector. Then a *linear approximation* of X_i, $i \in \{1, ..., m\}$ based on the variables $\boldsymbol{Z} \setminus X_i$ is defined as

$$\hat{X}_{i;\{1,...,m\}\setminus\{i\}} := a_1X_1 + ... + a_{i-1}X_{i-1} + a_{i+1}X_{i+1} + ... + a_mX_m + b_i = \sum_{\substack{j=1 \\ j \neq i}}^{m} a_jX_j + b_i,$$

where $a_j, b_i \in \mathbb{R} \; \forall \; i, j \in \{1, ..., m\}$.

The *best linear approximation* $X^*_{i;\{1,...,m\}\setminus\{i\}}$ of X_i is then the linear approximation which minimizes the mean squared error

$$e_{ms_i} := \mathbb{E}((X_i - \hat{X}_{i;\{1,...,m\}\setminus\{i\}})^2).$$

The last definition is based on [Congfeng] Chapter 3.

Definition 4.11: (Linear regression model)

A *linear regression model* assumes a linear relationship between a random variable Y and p other random variables $X_1, ..., X_p$ of the form

$$Y = \beta_0 + \beta_1X_1 + ... + \beta_pX_p + \epsilon, \quad \mathbb{E}(\epsilon) = 0.$$

Y is called *response variable* whereas $X_1, ..., X_p$ are called the *covariates*.

Definition 4.12: (Statistical significance of a covariate)

A certain covariate X_1 of a linear regression is said to be *statistically significant at level* α if the test statistics

$$H_0 : \beta_1 = 0 \text{ vs. } H_1 : \beta_1 \neq 0$$

is rejected at the significance level α. Therefore this covariate has an explaining character for the response variable.

For further details regarding linear regression and statistical significance, see [Fahrmeir (1996)] Chapter 4.

4.2 Factor model

The following two chapters are based on Chapter 11.1 of [Fahrmeir (1996)].

Consider a random vector $\boldsymbol{Y} = (Y_1, ..., Y_m)'$ consisting of the m random variables $Y_1, ..., Y_m$. Furthermore $\boldsymbol{\mu} = \mathbb{E}(\boldsymbol{Y}) = (\mu_1, ..., \mu_m)'$, where $\mu_i = \mathbb{E}(Y_i)\ \forall\ i \in \{1, ..., m\}$ and Σ is the covariance matrix of $\boldsymbol{Y}$.

Definition 4.13: (Factor model)

If there exists a linear relationship between the Y_i's and certain hidden random variables $F_1, ..., F_k$, $k < m$, called *common factors* and m additional random variables $E_1, ..., E_m$ so called *specific factors*, the *factor model* expresses the random variables through the hidden factors:

$$\begin{aligned} Y_1 &= \mu_1 + l_{11}F_1 + l_{12}F_2 + ... + l_{1k}F_k + E_1 \\ Y_2 &= \mu_2 + l_{21}F_1 + l_{22}F_2 + ... + l_{2k}F_k + E_2 \\ &\vdots \\ Y_m &= \mu_m + l_{m1}F_1 + l_{m2}F_2 + ... + l_{mk}F_k + E_m. \end{aligned}$$

The coefficients $l_{ij} \in \mathbb{R}$, $i \in \{1, ..., m\}$, $j \in \{1, ..., k\}$ signal the influence of the factor F_i on the variable Y_i. These coefficients are called *loadings*.

E_i summarizes all influences which solely have an effect on the corresponding Y_i and measuring errors.

Definition 4.14: (Matrix notation of the factor model)

In matrix notation the factor model can be written as:

$$\boldsymbol{Y} - \boldsymbol{\mu} = L\boldsymbol{F} + \boldsymbol{E},$$

where $\boldsymbol{F} = (F_1, ..., F_k)'$, $\boldsymbol{E} = (E_1, ..., E_m)'$ and $L = (l_{ij}) \in \mathbb{R}^{m \times k}$ is the so called *loading matrix*.

The factor model is based on the following assumptions.

Definition 4.15: (Assumptions of the factor model)

1. The factor variables as well as the specific factors are not observable, meaning they are random variables, i.e. we assume
$$\mathbb{E}(\boldsymbol{F}) = 0 \text{ and } \mathbb{E}(\boldsymbol{E}) = 0.$$
2. The factor variables $\boldsymbol{F}$ are standardized, i.e.
$$Var(F_i) = 1 \; \forall \; i \in \{1, ..., k\}$$
and their correlation with the specific factors $\boldsymbol{E}$ is zero, i.e.
$$Cov(\boldsymbol{F}, \boldsymbol{E}) = \mathbb{E}(\boldsymbol{F}\boldsymbol{E}') = 0.$$
3. The specific factors are uncorrelated, i.e.
$$\forall \; i, j \in \{1, ..., m\}, \; i \neq j : \;\; Cov(E_i, E_j) = 0.$$

Definition 4.16: (Residual variance matrix)

Let $\boldsymbol{E} = (E_1, ..., E_m)'$ be the specific factor vector from the factor model.
Definition 4.13 assumes that its elements are uncorrelated but as they can have arbitrary variances, we define
$$V := \mathbb{E}(\boldsymbol{E} \cdot \boldsymbol{E}') = diag\left\{v_1^2, ..., v_m^2\right\},$$
where $V \in \mathbb{R}^{m \times m}$ is called the *residual variance matrix.*
In particular we have
$$Var(E_j) = v_j^2 \; \forall \; j \in \{1, ..., m\}.$$

4.3 Factor model for observed data

We now have a look at observed realizations of the random vector $\boldsymbol{Y}$, i.e. the realization of the Y_i's, $\forall \; i \in \{1, ..., m\}$. If we have n independent observations of each Y_i, $\forall \; i \in \{1, ..., m\}$ denoted as
$$\boldsymbol{Y_i} := (Y_{1i}, ..., Y_{ni})' \in \mathbb{R}^n, \;\; \forall \; i \in \{1, ..., m\}$$
we can transfer these into the so called *data matrix* Y,
$$Y := (\boldsymbol{Y_1}, ..., \boldsymbol{Y_m}) = \begin{pmatrix} Y_{11} & ... & Y_{1m} \\ . & & . \\ . & & . \\ . & & . \\ Y_{n1} & ... & Y_{nm} \end{pmatrix} \in \mathbb{R}^{n \times m}.$$

Therefore, there are k factor variables

$$(F_{j1}, ..., F_{jk})' =: \tilde{\boldsymbol{F_j}} \in \mathbb{R}^k$$

and m specific factors

$$(E_{j1}, ..., E_{jm})' =: \tilde{\boldsymbol{E_j}} \in \mathbb{R}^m$$

for each observation of $\boldsymbol{Y}$, i.e. for each row of the data matrix Y, denoted by

$$\tilde{\boldsymbol{Y_j}} := (Y_{j1}, ..., Y_{jm})' \in \mathbb{R}^m, \quad \forall\ j \in \{1, ..., n\}\,.$$

Furthermore denote

$$\boldsymbol{F_i} := (F_{1i}, ..., F_{ni}) \in \mathbb{R}^n, \quad \forall\ i \in \{1, ..., k\} \text{ and } \boldsymbol{E_i} := (E_{1i}, ..., E_{ni}) \in \mathbb{R}^n, \quad \forall\ i \in \{1, ..., m\}\,.$$

This results in the matrix of factor variables

$$F := (\tilde{\boldsymbol{F_1}}, ..., \tilde{\boldsymbol{F_n}})' = (\boldsymbol{F_1}, ..., \boldsymbol{F_k}) = \begin{pmatrix} F_{11} & ... & F_{1k} \\ . & & . \\ . & & . \\ . & & . \\ F_{n1} & ... & F_{nk} \end{pmatrix} \in \mathbb{R}^{n \times k}$$

and the matrix of specific factors

$$E := (\tilde{\boldsymbol{E_1}}, ..., \tilde{\boldsymbol{E_n}})' = (\boldsymbol{E_1}, ..., \boldsymbol{E_m}) = \begin{pmatrix} E_{11} & ... & E_{1m} \\ . & & . \\ . & & . \\ . & & . \\ E_{n1} & ... & E_{nm} \end{pmatrix} \in \mathbb{R}^{n \times m}.$$

Define

$$M := \begin{pmatrix} \mu_1 & ... & \mu_m \\ . & & . \\ . & & . \\ . & & . \\ \mu_1 & ... & \mu_m \end{pmatrix} \in \mathbb{R}^{n \times m},$$

then we obtain the factor model for n observations and k factors

$$Y - M = FL' + E.$$

Let $Y \in \mathbb{R}^{n \times m}$ be the observed data matrix. We assume that

$$\forall\ j \in \{1, ..., m\} : Y_{ij} \sim \mathcal{N}(\mu_j, \sigma_j^2) \quad \forall\ i \in \{1, ..., n\}\, i.i.d.,$$

which stands for independent and identically distributed. In fact the Y_{ij}'s do not have to be normally distributed. The important point is that they are assumed to be independent and identically distributed with an arbitrary underlying distribution. Therefore the matrix contains n observations of m variables. In order to scale the observations, we calculate the *standardized data matrix* $Z \in \mathbb{R}^{n \times m}$, as follows:

$$Z = (Z_{ij}) \in \mathbb{R}^{n \times m},$$

where

$$Z_{ij} := \frac{Y_{ij} - \bar{Y}_j}{\sqrt{n-1} \cdot s_j}, \quad \bar{Y}_j := \frac{1}{n} \sum_{i=1}^{n} Y_{ij}, \quad s_j^2 := \frac{1}{n-1} \sum_{i=1}^{n} (Y_{ij} - \bar{Y}_j)^2. \tag{4.1}$$

Now we can derive the *empirical correlation matrix*:

$$R = Z'Z \in \mathbb{R}^{m \times m}. \tag{4.2}$$

4.4 Assumptions

Before we show how the Principal Components are constructed and interpreted we state the assumptions, which have to hold to successfully carry out a Principal Component Analysis.

Definition 4.17: (Assumptions for the Principal Component Analysis)

1. As mentioned above $\forall\ j \in \{1, ..., m\}$ Y_{ij} have to be independent and identical normally distributed with $\mathbb{E}(Y_{ij}) = \mu_j$ and $Var(Y_{ij}) = \sigma_j^2$, $\forall\ i \in \{1, ..., n\}$.
2. Z_j have to be linearly independent $\forall\ j \in \{1, ..., m\} \Leftrightarrow rank(Z) = m$.
3. $(Y_1, ..., Y_m)$ have to be multivariate normally distributed.
4. $R \neq I$ must hold for the empirical correlation matrix, where $I \in \mathbb{R}^{m \times m}$ is the identity matrix, in order to obtain interpretable Principal Components.

The Y_{ij}'s do not necessarily have to be normally distributed as stated in Assumption 1. The underlying distribution can be arbitrary which is one of the strengths of the Principal Component Analysis as it makes no suppositions on the distribution of the data, but is only of data manipulative nature. However, normally distributed data would make the solution more stable and stronger. Assumption 3 only has to hold in order to be able to perform the test which tests Assumption 4. In fact, Assumption 4 is not necessary to perform a Principal Component Analysis but it tests if such an analysis makes sense at all.

4.5 Applicability Check

In order to justify the application of the Principal Component Analysis on our data we have to check the assumptions stated in Definition 4.17.
First of all we examine if $\boldsymbol{Y_j}$ is normally distributed $\forall\ j \in \{1, ..., m\}$. Therefore we apply the following test.

Definition 4.18: (Kolmogorov-Smirnov test)

Let X_1 be an observed random variable with corresponding empirical distribution function $F_{X_{1_n}}$ and let F_0 be the suspected distribution function which should be tested against, for example the cumulative distribution function of a standard normal distribution Φ. The *Kolmogorov-Smirnov test* tests the null hypothesis

$$H_0 : F_{X_{1_n}}(x) = F_0(x) \text{ vs. } H_1 : F_{X_{1_n}}(x) \neq F_0(x)$$

by examining the test statistic

$$d_n := \|F_{X_{1_n}} - F_0\|_\infty = \sup_x |F_{X_{1_n}}(x) - F_0(x)|,$$

where $\sup\limits_x$ stands for the supremum over all x, see [Lilliefors (1967)].

We reject H_0 at significance level α if

$$d_n > d_\alpha := \frac{\sqrt{\ln(\frac{2}{\alpha})}}{\sqrt{2n}}. \tag{4.3}$$

Due to the Glivenko-Cantelli Theorem the empirical distribution converges almost surely to F_0 under H_0 for $n \to \infty$, see [Klenke (2006)] p. 111. So if our observed random variable would have the distribution with distribution function F_0, d_n should converge to zero for large n. That is why we reject the null hypothesis at significance level α if d_n exceeds the specific threshold d_α. The Kolmogorov-Smirnov test is a large sample test and therefore only has approximate α-level.
Next we should verify if the observed data vectors of our m variables are linear independent, in other words if $rank(Z) = m$ holds. If $rank(Z) = \tilde{m} < m$ we can always exclude the linear dependent columns from the variables to be examined until the rank of the new matrix $\tilde{Z}$ is equal to the number of its columns, i.e. terms and therefore Assumption 2 of Definition 4.17 holds. In other words

$$rank(\tilde{Z}) = \tilde{m} \text{ for } m > \tilde{m} \in \mathbb{N} \text{ and } \tilde{Z} \in \mathbb{R}^{n \times \tilde{m}}.$$

For the rest of this Chapter we assume that $rank(Z) = m$ and therefore $Z = \tilde{Z}$ and $R = Z'Z \in \mathbb{R}^{m \times m}$.

4.5.1 Bartlett's sphericity test

Eventually we can check if our data has the potential to be reduced to less factors than the number of variables we have observed, via Bartlett's sphericity test. Before we do this we have to verify Assumption 3 to check if our data is multivariate normally distributed.

Definition 4.19: (Mardia's test)

Let $\boldsymbol{Z} = (Z_1, ..., Z_m)' \in \mathbb{R}^m$ be a m-dimensional random vector with expectation vector $\boldsymbol{\mu} = (\mu_1, ..., \mu_m)'$ and covariance matrix Σ.

Mardia's test tests the null hypothesis

$$H_0 : \boldsymbol{Z} \sim \mathcal{N}_m(\boldsymbol{\mu}, \Sigma) \text{ vs. } H_1 : \boldsymbol{Z} \nsim \mathcal{N}_m(\boldsymbol{\mu}, \Sigma).$$

Let

$$\tilde{\boldsymbol{Z}}_i = (Z_{i1}, ..., Z_{im}) \in \mathbb{R}^m \quad \forall\, i \in \{1, ..., n\}$$

be the vector of the i-th observed random variables and

$$\bar{\boldsymbol{Z}} = (\bar{Z}_1, ..., \bar{Z}_m) \in \mathbb{R}^m \text{ where } \bar{Z}_j = \frac{1}{n}\sum_{i=1}^{n} Z_{ij} \quad \forall\, j \in \{1, ..., m\}\,.$$

Furthermore we define the *empirical covariance matrix*

$$\hat{\Sigma} := \frac{1}{n}\sum_{i=1}^{n}(\tilde{\boldsymbol{Z}}_i - \bar{\boldsymbol{Z}})(\tilde{\boldsymbol{Z}}_i - \bar{\boldsymbol{Z}})'.$$

Then Mardia's sample measure of multivariate skewness [Mardia (1970)]

$$A := \frac{1}{6n}\sum_{i=1}^{n}\sum_{j=1}^{n}((\tilde{\boldsymbol{Z}}_i - \bar{\boldsymbol{Z}})'\hat{\Sigma}^{-1}(\tilde{\boldsymbol{Z}}_j - \bar{\boldsymbol{Z}}))^3,$$

and Mardia's sample measure of multivariate kurtosis [Mardia (1970)]

$$B := \sqrt{\frac{n}{8m(m+2)}}\left(\frac{1}{n}\sum_{i=1}^{n}((\tilde{\boldsymbol{Z}}_i - \bar{\boldsymbol{Z}})'\hat{\Sigma}^{-1}(\tilde{\boldsymbol{Z}}_i - \bar{\boldsymbol{Z}}))^2 - m(m+2)\right)$$

can be calculated.

Under H_0,

$$A \sim \chi^2_{\frac{1}{6}m(m+1)(m+2)} \text{ and } B \sim \mathcal{N}(0, 1),$$

see [Mardia (1970)] p. 523 and p. 527.

So we reject H_0 if $A \sim \chi^2_{\frac{1}{6}m(m+1)(m+2)}$ or $B \sim \mathcal{N}(0,1)$ is rejected, for example by the Kolmogorov-Smirnov test, i.e. in this case the empirical multivariate skewness and kurtosis.

Definition 4.20: (Bartlett's sphericity test)

Let $R \in \mathbb{R}^{m\times m}$ and I be the $m \times m$ identity matrix.
Bartlett's sphericity test tests, under the crucial assumption of multivariate normal distribution of the data, the null hypothesis

$$H_0 : R = I \text{ vs. } H_1 : R \neq I.$$

The null hypothesis will be rejected at significance level α if

$$X^2 = -\left(n - 1 - \frac{2m+5}{6}\right)\log(\det(R)) > \chi^2_{\frac{1}{2}m(m-1),\alpha}, \tag{4.4}$$

since X^2 is approximately $\chi^2_{\frac{1}{2}m(m-1)}$- distributed, as $n \to \infty$, see [Cochran (1989)].

As

$$\boldsymbol{Z_i} \sim \mathcal{N}(0,1) \quad i.i.d. \quad \forall\, i \in \{1,...,n\}$$

and

$$R = Z'Z = (\boldsymbol{Z_1},...,\boldsymbol{Z_m})'(\boldsymbol{Z_1},...\boldsymbol{Z_m}) = \sum_{i=1}^{n} \boldsymbol{Z_i}^2 \quad \Rightarrow R \sim \chi^2_m.$$

As the logarithm just increases the convergence to a normal distribution and from

$$X_i \sim \chi^2_{m_i}, \quad \forall\, i \in \{1,...,n\} \text{ and independent } \Rightarrow \sum_{i=1}^{n} \chi^2_{m_i} \sim \chi^2_{\sum_{i=1}^{n} m_i}$$

it follows that X^2 is approximately $\chi^2_{\frac{1}{2}m(m-1)}$- distributed, see Definition 4.20.
We therefore test if the correlation matrix R equals the identity matrix. If that is the case, meaning that the variables are totally uncorrelated and therefore we need as many factors as variables, a Principal Component Analysis is not useful at all. The other extreme case would be if all variables are perfectly correlated, meaning just one factor suffices to explain all of the data. The major drawback of this test is that it tends to always reject the null hypothesis if n gets very large, which will be the case in our applications as we extract thousands of tweets.

4.5.2 Kaiser-Mayer-Olkin (KMO) measure

Because of the above mentioned drawback of Bartlett's sphericity test we will focus on the KMO measure. The goal of the KMO measure is also testing if a PCA can be effectively carried out. But in contrast to Bartlett's test it does not only take into account the correlations but the partial correlations.

Definition 4.21: (Partial Correlation)

Let $X_1, ..., X_m$ be random variables and $\boldsymbol{Z} = (X_1, ..., X_m) \in \mathbb{R}^m$ the corresponding random vector.
Furthermore let $X^*_{i;\{1,...,m\}\setminus\{i,j\}}$ and $X^*_{j;\{1,...,m\}\setminus\{i,j\}}$ be the best linear approximations for the random variables X_i and X_j based on the random variables $\boldsymbol{Z} \setminus \{X_i, X_j\}$, $\forall\ i,j \in \{1,...,m\}$.
If we then define

$$R_i := X_i - X^*_{i;\{1,...,m\}\setminus\{i,j\}} \text{ and } R_j := X_j - X^*_{j;\{1,...,m\}\setminus\{i,j\}},$$

the *partial correlation* between X_i and X_j, $\forall\ i,j \in \{1,...,m\}$ is defined as

$$a_{ij} = Cor(X_i, X_j|\boldsymbol{Z} \setminus \{X_i, X_j\}) := \frac{\mathbb{E}(R_i - \mathbb{E}(R_i))(R_j - \mathbb{E}(R_j))}{\sqrt{Var(R_i) \cdot Var(R_j)}}, \quad \text{see [12].}$$

Corollary 4.22: (Calculation of partial correlation)

If X_i is a normally distributed random variable $\forall\ i \in \{1,...,m\}$ and $R \in \mathbb{R}^{m\times m}$ is the respective correlation matrix then the partial correlation matrix $A := (a_{ij})$ can be computed as follows:

$$a_{ij} = -\frac{v_{ij}}{\sqrt{v_{ii}v_{jj}}} \quad \forall\ i,j \in \{1,...,m\}\,, \quad i \neq j \text{ and } a_{ii} = 1 - \frac{1}{v_{ii}} \quad \forall\ i \in \{1,...,m\}\,,$$

whereas $R^{-1} := (v_{ij})$.

Proof. [Eichler (2007)]. □

The partial correlation between two variables measures the pure correlation between them, after removing the effect of the other variables on it.

Definition 4.23: (KMO index)

The *KMO index* is calculated by:

$$KMO := \frac{\sum\limits_i \sum\limits_{j\neq i} r_{ij}^2}{\sum\limits_i \sum\limits_{j\neq i} r_{ij}^2 + \sum\limits_i \sum\limits_{j\neq i} a_{ij}^2},$$

where $R = (r_{ij})$, see [13].

Therefore, the KMO index gets close to one if there is almost no partial correlation, which means that all the variables have influence on each other and a Principal Component Analysis makes sense. On the other hand if the KMO index gets below the Kaiser-Rice critical value of 0.5, see [Cureton (1983)] p. 391, meaning partial correlations are high, so most of the correlation is driven by the influence of solely two variables on each other, not many advantages will arise from a Principal Component Analysis.

4.6 Construction of the Principal Components

We now want to decompose the standardized data matrix Z into a matrix with orthonormal columns, which are ordered decreasingly by the share of the overall variance they explain and the loading matrix L as follows:

$$Z = FL', \quad F = (\boldsymbol{F_1}, ..., \boldsymbol{F_m}) \in \mathbb{R}^{n\times m}, \quad L \in \mathbb{R}^{m\times m},$$

whereas the first k normalized principal axis $\boldsymbol{F_1}, ..., \boldsymbol{F_k}$ will later on be the *Principal Components*.

The following theorem states the steps to calculate these Principal Components under Assumption 2 of Definition 4.17.

Theorem 4.24: (Construction of the Principal Components)

Let $Z \in \mathbb{R}^{n\times m}$ with $rank(Z) = m$, then one gets the orthogonal and normed factors $\boldsymbol{F_i}$, $F = (\boldsymbol{F_1}, ..., \boldsymbol{F_m})$, which are called Principal Components through

$$F := H\Lambda^{-\frac{1}{2}} \in \mathbb{R}^{n\times m},$$

where

$$H := (\boldsymbol{H_1}, ..., \boldsymbol{H_m}) = ZT \in \mathbb{R}^{n\times m}$$

is the matrix of the principal axis of Z and $T \in \mathbb{R}^{m\times m}$ is the orthonormal matrix of eigenvectors to the corresponding ordered eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_m > 0$ of $R = Z'Z$. Furthermore

$$\Lambda := diag\left\{\lambda_1, ..., \lambda_m\right\} \in \mathbb{R}^{m\times m}$$

and the loading matrix can be calculated by

$$L := T\Lambda^{\frac{1}{2}} \in \mathbb{R}^{m\times m}.$$

Another representation of F can be derived via

$$F = ZR^{-1}L.$$

Proof. The fact that the principal axis $\boldsymbol{H_i}$ can be calculated as described above is proven in [Fahrmeir (1996)] p. 663 f. Then

$$H'H = (ZT)'ZT = T'Z'ZT = T'RT = T'\boldsymbol{\lambda}T = \boldsymbol{\lambda}I = \Lambda \in \mathbb{R}^{m\times m}$$

$$\boldsymbol{\lambda} := (\lambda_1, ..., \lambda_m)' \in \mathbb{R}^m$$

holds, so the principal axis $\boldsymbol{H_i}$ are orthogonal. Furthermore

$$HT = ZTT' = Z \text{ and } R = Z'Z = (HT')'HT' = TH'HT' = T\Lambda T'.$$

It follows that

$$F'F = (H\Lambda^{-\frac{1}{2}})'H\Lambda^{-\frac{1}{2}} = \Lambda^{-\frac{1}{2}}H'H\Lambda^{-\frac{1}{2}} = \Lambda^{-\frac{1}{2}}\Lambda\Lambda^{-\frac{1}{2}} = I$$

proving $\boldsymbol{F_i}$ are orthonormal and

$$FL' = H\Lambda^{-\frac{1}{2}}(T\Lambda^{\frac{1}{2}})' = H\Lambda^{-\frac{1}{2}}\Lambda^{\frac{1}{2}}T' = HT' = Z.$$

By looking at the decomposition of

$$Z = FL' = F(T\Lambda^{\frac{1}{2}})' = F\Lambda^{\frac{1}{2}}T',$$

and solving it for F one gets:

$$F = ZT\Lambda^{-\frac{1}{2}} = ZT\Lambda^{-1}T'T\Lambda^{\frac{1}{2}} = ZR^{-1}L.$$

□

4.7 Interpretation and consequences

Assume we constructed the matrix of the Principal Components F, so that $Z = FL'$ holds. We then have to decide for the number of components to keep.
A simple possibility to find k is choosing

$$\max_{k\in\{1,...,m\}} \lambda_k \geq 1.$$

But we can also apply a graphical procedure called the Scree test, see [Fahrmeir (1996)] p. 669.

Definition 4.25: (Scree test)

The basic assumption is that, in contrast to correlated data, the eigenvalues of random data are typically nearly constant. So the *Scree test* looks at the plot of decreasingly ordered eigenvalues and determines at which eigenvalue the drop in value ceases and the curve makes an elbow and becomes less steep. The number of this eigenvalue should be chosen to be k.

Of course this criterion is very subjective, but if coordinated with the first criterion it can give valid justification for the choice of k. In many cases the two criteria coincide.
Assume we decided to continue with k Principal Components. So

$$Z = F^k L^{k'} + E^k, \quad E^k = \boldsymbol{F_{k+1}l'_{k+1}} + ... + \boldsymbol{F_m l'_m}$$

holds, where

$$F^k = (\boldsymbol{F_1}, ..., \boldsymbol{F_k}) \text{ and } L^k = (\boldsymbol{l_1}, ..., \boldsymbol{l_k}).$$

By only considering $F^k L^{k'}$ we get an approximation of Z which explains

$$\frac{\lambda_1 + ... + \lambda_k}{trace(R)}$$

of the overall variance of the data. If the choice of k is suitable this will suffice to explain the data adequately. So, without losing any significant information we reduced the amount of variables from m to k.

The matrix $F^k \in \mathbb{R}^{n \times k}$ contains the transformed and dimension reduced data, now represented by k variables, let's call them factor 1 to factor k. By looking at the entries of the loading matrix L, or L^k in the reduced case, we can identify meaningful statements and names for the factors. Usually if

$$|l_{ij}| > 0.5 \text{ for } i, j \in \{1, ..., k\},$$

we would consider this loading as high and assume that the variable i has large influence on factor j.

We detected new meaningful variables, i.e. factors which are linear combinations of the original variables and also reduced the number of variables. We assume that the original data as well as new data can be classified better via the extracted factors than via the original variables, as they are classified with regard to formerly hidden patterns in the data and because the majority of the noise contained in it was removed. In addition it is easier to visualize the data as we reduced dimensions. Reduction to two dimensions would be optimal as we could plot the transformed data in a two dimensional coordinate system. This means we plot the original data projected onto the two dimensional plane with the basis $\{factor1, factor2\}$. But also a reduction to three dimensions is already helpful, as we could look at the transformed data in three different ways. Once projected onto the factor 1 - factor 2 plane, once onto the factor 1 - factor 3 plane and once onto the factor 2 - factor 3 plane, respectively, or by simply plotting a 3D plot if the data has a clear structure. This makes a clustering of the data much easier and more convenient.

In order to assign every tweet to its cluster we introduce the k-means algorithm in the following. The aim of this algorithm is the partition of the data into K clusters. The reason we choose the capital K for the number of clusters is that it is not confused with the number of components to keep, which already is denoted by k. The major drawback is that we have to decide on the number of clusters K to build upfront by our own. The algorithm then minimizes the sum of squares within the K clusters by reassigning the data points until no further improvement is possible. The most common algorithm to achieve this is the Lloyd's algorithm:

Algorithm 1 Lloyd's algorithm

```
1: Choose desired number of clusters $K$.
2: Choose $K$ random means from the data $m_1^{(1)}, ..., m_K^{(K)}$.
3: Assign each observation to the cluster whose mean yields the least sum of squares
   within the respective cluster:
       $S_i^{(t)} = \left\{ x_p : \|x_p - m_i^{(t)}\|^2 \leq \|x_p - m_j^{(t)}\|^2 \quad \forall j, 1 \leq j \leq K \right\}.$
4: Calculate the new cluster means as the centroids of the cluster:
       $m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j.$
Repeat 3 and 4 until assignments of the data to the clusters are not changing anymore.
```

For more details see [MacKay (2003)] p. 284-292.

Later on, we will also use a more robust clustering method called k-medoids. This method is very similar to the k-means clustering, but instead of random means, the k-medoids method uses data points as centers, called medoids. After choosing the desired number of clusters K, we calculate the Euclidean distance between all pairs of data points

$$d_{ij} = \sqrt{\sum_{l=1}^{m} (x_{il} - x_{jl})^2}, \quad \forall i, j \in \{1, ...n\} \, i < j$$

given n data points $x_1, ..., x_n \in \mathbb{R}^m$. We then calculate

$$v_i = \sum_{j=1}^{n} \frac{d_{ij}}{\sum\limits_{l=1}^{n} d_{il}}$$

for every data point x_i. Then the K smallest values of v_i are determined and their respective data points are chosen as the initial medoids. After assigning each data point to their corresponding, i.e. nearest, medoid we calculate the sum of distances from all objects to their medoids. Update the medoids by minimizing the total distance to other objects in its cluster and reassign the data points to the updated clusters as long as the sum of distances decreases. For more details see [Park (2009)]. It is also possible to determine the number of clusters automatically via the so called silhouette technique. This technique compares different clustering results with a dissimilarity measure which evaluates how well the certain data points match to their respective cluster and then decides in favour of the clustering result with the lowest sum of these measures. For more details and the construction of the dissimilarity measure see [Rousseeuw (1987)].

We are now abled to assign tweets to clusters and then examine these clusters in a second step, in order to structure and get an insight in the extracted data, with the help of this algorithm.

4.8 Example

In the following we will see an example of how Principal Component Analysis can be applied to textual data. The several steps as well as the dimension reduction and the graphical clustering will get clearer by this example.

Example 4.1. *Principal Component Analysis applied to textual data.*

We therefore pick up the document-term matrix with TfIdf weighting from Example 3.1.

Docs/Terms	#bigdata	#masterthesis	#student	@tumuenchen	great	math	record	writing
1	0.000	0.000	0.184	0	0.331	0.331	0.000	0.000
2	0.661	0.000	0.000	0	0.000	0.000	0.000	0.000
3	0.000	0.000	0.246	0	0.000	0.000	0.774	0.000
4	0.000	0.000	0.184	0	0.331	0.331	0.000	0.000
5	0.331	0.581	0.000	0	0.000	0.000	0.000	0.581

1. Calculation of Z

Now we calculate the standardized data matrix Z from the document-term matrix $DTMw$ through (4.1).

```
> # calculate the standardized data matrix Z
> n <- dim(DTMw)[1]
> m <- dim(DTMw)[2]
> # calculate the empirical mean of the variables
> mean <- colMeans(DTMw)
> # calculate the empirical variance of the variables
> s <- vector(mode = "numeric" , length = m)
> for (i in 1:m){
+   s[i] <- sum((DTMw[ , i] - mean[i])^2)/(n-1)
+ }
> Z <- matrix(nrow = n , ncol = m)
> for (i in 1:m){
+   for (j in 1:n){
+     Z[j , i] <- (DTMw[j , i] - mean[i])/(sqrt(n-1)*sqrt(s[i]))
+   }
+ }
> colnames(Z) <- colnames(DTMw)
> # remove columns with NaN entries
> Z <- Z[ , complete.cases(t(Z))]
> show(Z)
```

Docs/Terms	#bigdata	#masterthesis	#student	great	math	record	writing
1	−0.335	−0.224	0.267	0.548	0.548	−0.224	−0.224
2	0.783	−0.224	−0.535	−0.365	−0.365	−0.224	−0.224
3	−0.335	−0.224	0.535	−0.365	−0.365	0.894	−0.224
4	−0.335	−0.224	0.267	0.548	0.548	−0.224	−0.224
5	0.224	0.894	−0.535	−0.365	−0.365	−0.224	0.894

As we can see the columns have been standardized and the term *@tumuenchen* was removed as it had only zeros as entries.

2. Test linear independence of columns of Z

In the next step we exclude columns from Z until the rank of the new matirx $\tilde{Z}$ is equal to the number of its columns, to ensure linear independence of the variables.

```
> # find linear dependent variables and delete them
> rankMatrix(Z)

[1] 3
attr(,"method")
[1] "tolNorm2"
attr(,"useGrad")
[1] FALSE
attr(,"tol")
[1] 1.554312e-15

> Ztilde <- Z
> if(rankMatrix(Ztilde) < m){
+   for (i in m:1){
+     if(rankMatrix(Ztilde[ , -i]) == rankMatrix(Ztilde)){
+       Ztilde <- Ztilde[ , -i]
+     }
+   }
+ }
> rankMatrix(Ztilde)

[1] 3
attr(,"method")
[1] "tolNorm2"
attr(,"useGrad")
[1] FALSE
attr(,"tol")
[1] 1.110223e-15

> m <- dim(Ztilde)[2]
> show(Ztilde)
```

Docs/Terms	#bigdata	#masterthesis	#student
1	−0.335	−0.224	0.267
2	0.783	−0.224	−0.535
3	−0.335	−0.224	0.535
4	−0.335	−0.224	0.267
5	0.224	0.894	−0.535

As we can see the rank of Z is three, so we excluded five linear dependent columns, resulting in the new matrix $\tilde{Z}$ consisting only of the three columns *#bigdata*, *#masterthesis* and *#student*, which are now linear independent.

3. Test if columns of $\tilde{Z}$ are normally distributed

In the next step we test if the columns of $\tilde{Z}$ are normally distributed via the Kolmogorov-Smirnov test according to Definition 4.18.

```
> # test if columns of Ztilde are normally distributed
> for(i in 1:m){
+   if(ks.test(Ztilde[ , i] , y = 'pnorm' , alternative = 'two.sided')$p.value
+                                                            < 0.1){
+     print(ks.test(Ztilde[ , i] , y = 'pnorm' , alternative = 'two.sided'))
+     print("NO")
+   }else{
+     print(ks.test(Ztilde[ , i] , y = 'pnorm' , alternative = 'two.sided'))
+     print("YES")
+   }
+ }

        One-sample Kolmogorov-Smirnov test

data:  Ztilde[, i]
D = 0.3687, p-value = 0.5051
alternative hypothesis: two-sided

[1] "YES"

        One-sample Kolmogorov-Smirnov test

data:  Ztilde[, i]
D = 0.4115, p-value = 0.3654
alternative hypothesis: two-sided

[1] "YES"

        One-sample Kolmogorov-Smirnov test

data:  Ztilde[, i]
D = 0.2965, p-value = 0.7717
alternative hypothesis: two-sided

[1] "YES"
```

As

$$d_{0.1} = \frac{\sqrt{\ln(\frac{2}{0.1})}}{\sqrt{2 \cdot 5}} \approx 0.5473 \text{ and } D < d_{0.1}$$

for all three tests, due to equation (4.3) the null hypothesis cannot be rejected at significance level $\alpha = 0.1$ for all three columns. Therefore we can assume that the columns of $\tilde{Z}$ are normally distributed.

4. Test of multivariate normal distribution

Now we test if $(\tilde{\boldsymbol{Z}}_{\mathbf{1}}, \tilde{\boldsymbol{Z}}_{\mathbf{2}}, \tilde{\boldsymbol{Z}}_{\mathbf{3}})$ is multivariate normally distributed via Mardia's test which was introduced in Definition 4.19.

```
> # test if Ztilde is multivariate normally distributed
> mardiaTest(Ztilde , cov = TRUE , qqplot = TRUE)
```

```
   Mardia's Multivariate Normality Test
---------------------------------------
   data : Ztilde

   g1p            : 7.5
   chi.skew       : 6.25
   p.value.skew   : 0.7938401

   g2p            : 10.5
   z.kurtosis     : -0.9185587
   p.value.kurt   : 0.3583265

   chi.small.skew : 13.33333
   p.value.small  : 0.2056272

   Result         : Data is multivariate normal.
---------------------------------------
```

As both of the p-values of the sample measure of multivariate skewness $g1p = A$ and of Mardia's multivariate kurtosis $g2p = B$ are higher than $\alpha = 0.05$ the null hypothesis cannot be rejected at significance level α and we therefore assume that $(\tilde{\boldsymbol{Z}}_{\mathbf{1}}, \tilde{\boldsymbol{Z}}_{\mathbf{2}}, \tilde{\boldsymbol{Z}}_{\mathbf{3}})$ is multivariate normally distributed.

5. Bartlett's sphericity test

Next we calculate the empirical correlation matrix R as in equation (4.2) and apply Bartlett's sphericity test analogously to Definition 4.20, in order to find out if a Principal Component Analysis will be useful.
The latter is applicable as we showed multivariate normal distribution of $\tilde{Z}$ before.

```
> # calculate empirical correlation matrix
> R <- t(Ztilde)%*%Ztilde
> rankMatrix(R)
```

```
[1] 3
attr(,"method")
[1] "tolNorm2"
attr(,"useGrad")
[1] FALSE
attr(,"tol")
[1] 6.661338e-16
```

```
> show(R)
```

Docs/Terms	#bigdata	#masterthesis	#student
#bigdata	1.000	0.250	−0.896
#masterthesis	0.250	1.000	−0.598
#student	−0.896	−0.598	1.000

```
> # Bartlett's test
> chi <- -(n-1-(2*m+5)/6)*log(det(R))
> chi
```

```
[1] 6.736299
```

```
> dof <- m*(m-1)/2
> dof
```

```
[1] 3
```

```
> pchisq(chi , dof , lower.tail = FALSE)
```

```
[1] 0.08079508
```

```
> qchisq(0.1 , dof , lower.tail = FALSE)
```

```
[1] 6.251389
```

```
> chi > qchisq(0.1 , dof , lower.tail = FALSE)
```

```
[1] TRUE
```

As $0.0808 < 0.1$ and $6.7363 > 6.2514$ we reject the null hypothesis at significance level $\alpha = 0.1$ due to equation (4.4) and conclude that a Principal Component Analysis should be carried out.

6. KMO measure

Although Bartlett's sphericity test suggests a Principal Component Analysis and n is not large, meaning we can rely on this test, we will calculate the KMO index, as stated in Definition 4.23 as well.

```
> # KMO index
> # calculate inverse of R
> V <- solve(R)
> # calculate partial correlation matrix due to Corollary \ref{partialcorrelation}
> A <- matrix(nrow = m , ncol = m)
> for(i in 1:m){
+   for(j in 1:m){
+     if(i == j){
+       A[i , j] <- 1-1/V[i , j]
+     }else{
+       A[i , j] <- -V[i , j]/(sqrt(V[i , i]*V[j , j]))
+     }
+   }
+ }
> # calculate KMO index itself
> qusur <- vector(mode = "numeric" , length = m)
> qusua <- vector(mode = "numeric" , length = m)
> for(i in 1:m){
+   for(j in 1:m){
+     if(i != j){
+       qusur[i] <- qusur[i] + R[i , j]^2
+       qusua[i] <- qusua[i] + A[i , j]^2
+     }
+   }
+ }
> qusur <- sum(qusur)
> qusua <- sum(qusua)
> KMO <- qusur/(qusur+qusua)
> KMO
```

```
[1] 0.344259
```

```
> KMO > 0.5
```

```
[1] FALSE
```

As the KMO index is smaller than the Kaiser-Rice critical value of 0.5 we doubt if a Principal Component Analysis makes sense. We conduct it anyway as Bartlett's test suggests this.

7. Principal Component Analysis

First of all we calculate the loading matrix L and the matrix of Principal Components F according to Theorem 4.24.

```
> # PCA
> # calculation of the eigenvalues
> lambda <- eigen(R)$value
> lambda
```

```
[1] 2.202 0.771 0.026
```

```
> Lambda <- diag(lambda)
> # calculation of the eigenvectors
> T <- eigen(R)$vectors
> # compute loading matrix
> L <- T %*% sqrt(Lambda)
> show(L)
```

$$\begin{array}{lrrr}
 & [,1] & [,2] & [,3] \\
[1,] & -0.877 & -0.470 & 0.098 \\
[2,] & -0.674 & 0.737 & 0.048 \\
[3,] & 0.990 & 0.086 & 0.120
\end{array}$$

```
> # compute principal components
> F <- Ztilde%*%(solve(R))%*%L
> show(F)
```

$$\begin{array}{lrrr}
 & [,1] & [,2] & [,3] \\
[1,] & 0.322 & 0.020 & -0.443 \\
[2,] & -0.483 & -0.750 & 0.066 \\
[3,] & 0.442 & 0.050 & 0.776 \\
[4,] & 0.322 & 0.020 & -0.443 \\
[5,] & -0.603 & 0.659 & 0.043
\end{array}$$

8. Choice of k

In the next step we want to decide how many components we want to keep. For this we examine the eigenvalues of R.

```
> # plot eigenvalues to choose k
> plot(1:m , lambda , xlab = "index" , ylab = "eigenvalues" ,
+  xaxt = "n" , main = "Eigenvalues of R")
> axis(side = 1 , at = 1:m)
> abline(1 , 0  ,lty = "dotted")
> lines(lambda , col = "red")
```

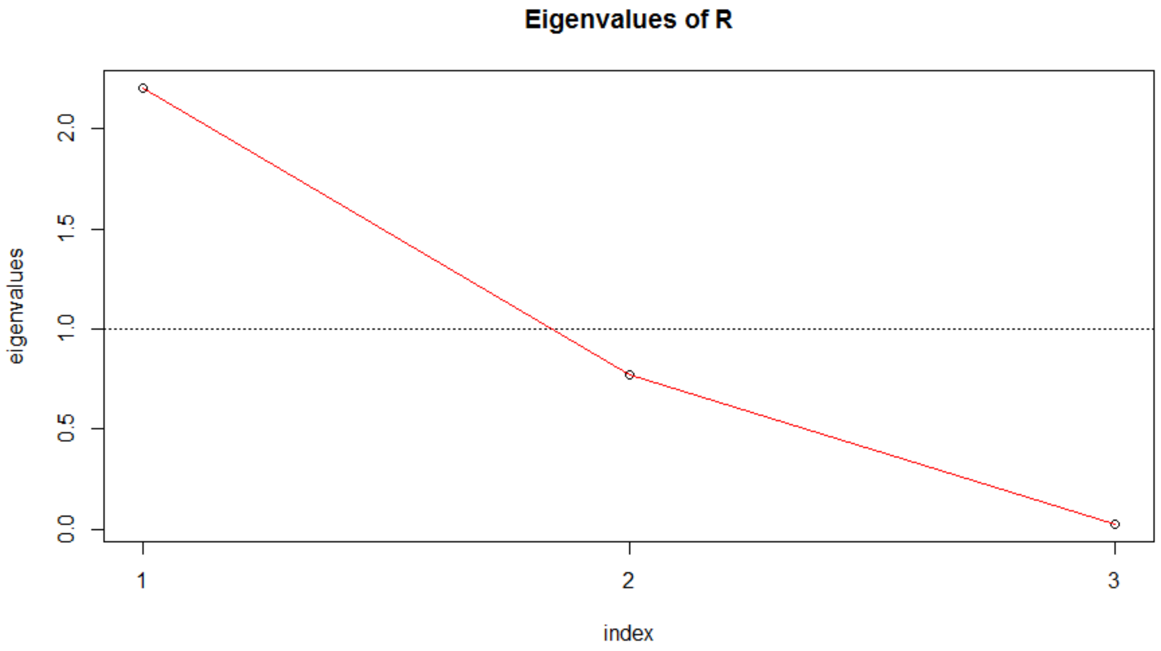


Figure 4.1: Eigenvalues of R

```
> # which eigenvalues are bigger than 1
> k <- length(which(lambda >= 1))
> k

[1] 1
```

Although only the first eigenvalue has a value larger than one, we decide to keep two dimensions as the Scree test, see Definition 4.25 would suggest so and also graphical visualisation will be clearer.

9. Reduction of dimensions

So now we reduce to two dimensions and look at the high loadings of the reduced loading matrix L^k to identify the terms which have the most influence on the two remaining factors.

```
> k <- 2
> # consider only first k components
> LHK <- L[ , 1:k]
> rownames(LHK) <- c(colnames(Ztilde))
> colnames(LHK) <- c(1:k)
> show(LHK)
```

	1	2
#bigdata	−0.877	−0.470
#masterthesis	−0.674	0.737
#student	0.990	0.086

```
> ind <- list()
> high<- list()
> for (i in 1:k){
+   ind[[i]] <- which(abs(LHK[ , i]) > 0.5)
+   high[[i]] <- rownames(LHK)[ind[[i]]]
+ }
> high

[[1]]
[1] "#bigdata"      "#masterthesis" "#student"

[[2]]
[1] "#masterthesis"

> FHK <- F[ , 1:k]
> show(FHK)
```

	[, 1]	[, 2]
[1,]	0.322	0.020
[2,]	−0.483	−0.750
[3,]	0.442	0.050
[4,]	0.322	0.020
[5,]	−0.603	0.659

```
> colnames(FHK) <- c(1:k)
> # how much of the variance is captured
> # with the first k principal components
> lambda[1:k]/tr(R)

[1] 0.9912394
```

As we see all three remaining terms have a high influence on the first factor, whereas only the term '#masterthesis' has a high loading on the second factor. This result is very hard to interpret as simply all terms load high on the first factor. This is due to the low rank of our minimal made up tweet example. Nevertheless we will try to cluster the five documents and verify if this clustering done by the Principal Component Analysis makes sense. Furthermore 99.12 % of the overall variance is captured through the first two factors, signalling a very good approximation through these two factors. At this point, we once again add, that one factor would probably suffice, as it already explains 73.42 % of the overall variance.

10. Graphical Clustering

In the next step we plot the documents in their new reduced coordinates. This can be seen as the projection of the original coordinates of our documents onto the factor 1 - factor 2 plane.

```
> # plot documents in new coordinates
> set.seed(9876)
> names <- c("d1" , "d2" , "d3" , "d4" , "d5")
> textplot(FHK[ , 1] , FHK[ , 2] , names , main = "Document Clustering",
+          xaxt = "n" , yaxt = "n" , xlab = "Factor 1"  ,
+          ylab = "Factor 2" , xlim = c(min(FHK[ , 1]) ,
+          max(FHK[ , 1])) , ylim = c(min(FHK[ , 2]) , max(FHK[ , 2])))
```

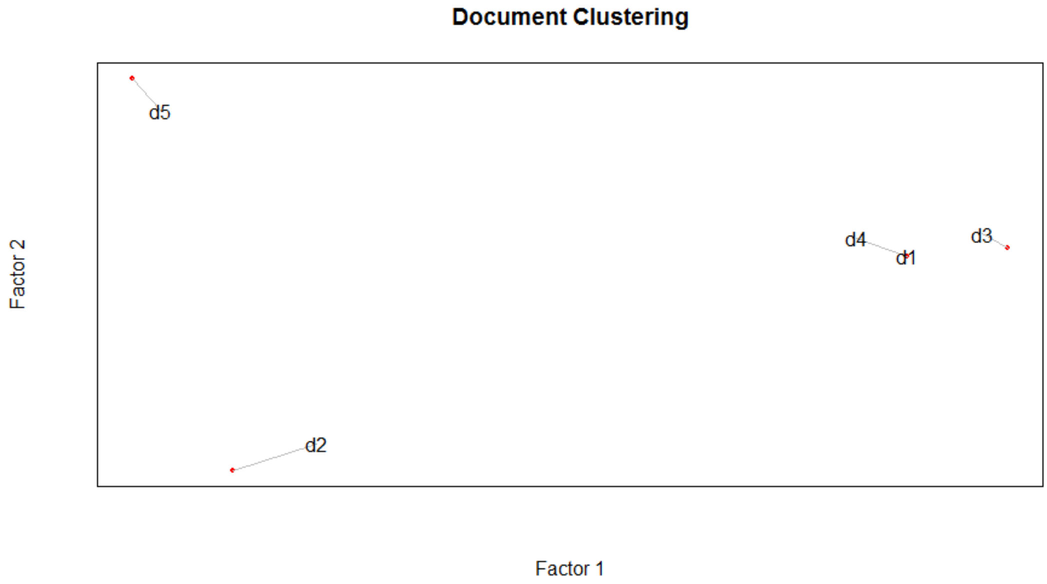


Figure 4.2: Document Clustering

Although our little example is not perfect for carrying out a Principal Component Analysis, we see some very convincing results in Figure 4.2.

Document 1 and Document 4 have exactly the same coordinates, which is clear since the latter was a retweet of Document 1. Furthermore we can see that Document 3 is plotted nearby those two. That also makes sense since those three documents contain information regarding the students of the TU Munich. Although Document 2 and Document 5 are not situated nearby each other at a first glance, they have very similar factor 1-values which might suggest to assign them to one cluster. By looking at the content this would definitely be reasonable as they both treat the examination of big data at the TU Munich.

11. K-means clustering

If we apply the k-means algorithm with $K = 3$ to the five observations and their new coordinates we get the following clusters:

```
> # clustering via k-means
> clus <- 3
> set.seed(8796)
> kmeansResult <- kmeans(F[ , 1:2] , clus)
> clusterkmeans <- kmeansResult$cluster
> textplot(F[ , 1] , F[ , 2] , names , main = "Document Clustering k-means" ,
+      xaxt = "n" , yaxt = "n" , xlab = "Factor 1"  , ylab = "Factor 2" ,
+      xlim = c(min(F[ , 1]) , max(F[ , 1])) ,
+      ylim = c(min(F[ , 2]) , max(F[ , 2])) ,
+      col = clusterkmeans)
```

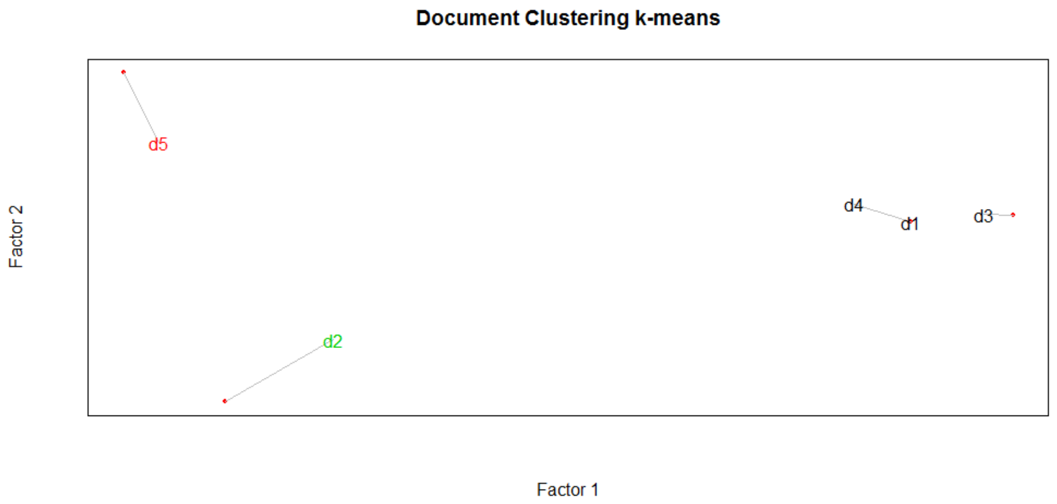


Figure 4.3: Document Clustering with k-means

As we see the first cluster includes the Documents 1, 3 and 4, the second cluster Document 5 and the third Document 2.

So we got a clustering similar to the one a human being would have made through the Principal Component Analysis. Therefore the application on bigger data sets, which cannot be ordered and grouped manually anymore, will be very interesting.

Chapter 5

Application 1 - Topic: Islamic State

We now want to investigate the similarities, meanings and interrelations of tweets containing a specific keyword. We also want to detect hidden patterns and structures in these documents. To sum up, we want to get the big picture of the tweets treating our subject. The goal will be to cluster the tweets in certain groups to identify the main opinions and points of views to a certain subject. Also the ratio and the development over time of these clusters could be observed and analyzed. Furthermore new incoming tweets could be assigned to already existing clusters in order to obtain a continuous real time assessment of the extracted tweets.
Our first example are tweets containing the keyword Islamic State, which was a hot topic in the autumn of 2014. The whole world looked at the development of this fundamentalist state and wondered about the next move of this cruel movement, as well as the next steps of the western world and how they act against it.

5.1 General Time Patterns

First of all, we examine general time patterns in the tweets by reading in all tweets containing the keyword Islamic State in November 2014 and aggregate them for every minute. We have to take into account that the extraction routine starts at the end of each day, i.e. at 23:59:59, and ends at its beginning, i.e. at 0:00:00, in the case that no cancellation error occurred. So, for each minute, a maximum of 1440 time points per day, we yield the amount of tweets containing our keyword. The code which reads the tweets and aggregates them for every minute of every day of November 2014 is given in the Appendix B.2. For every day in November 2014 the frequency of tweets over the day is plotted to visualize and identify the most active times of the respective day. Therefore, the date is plotted at the maximum of each curve which represent the days of November 2014.

The smoothed tweet curves are given below:

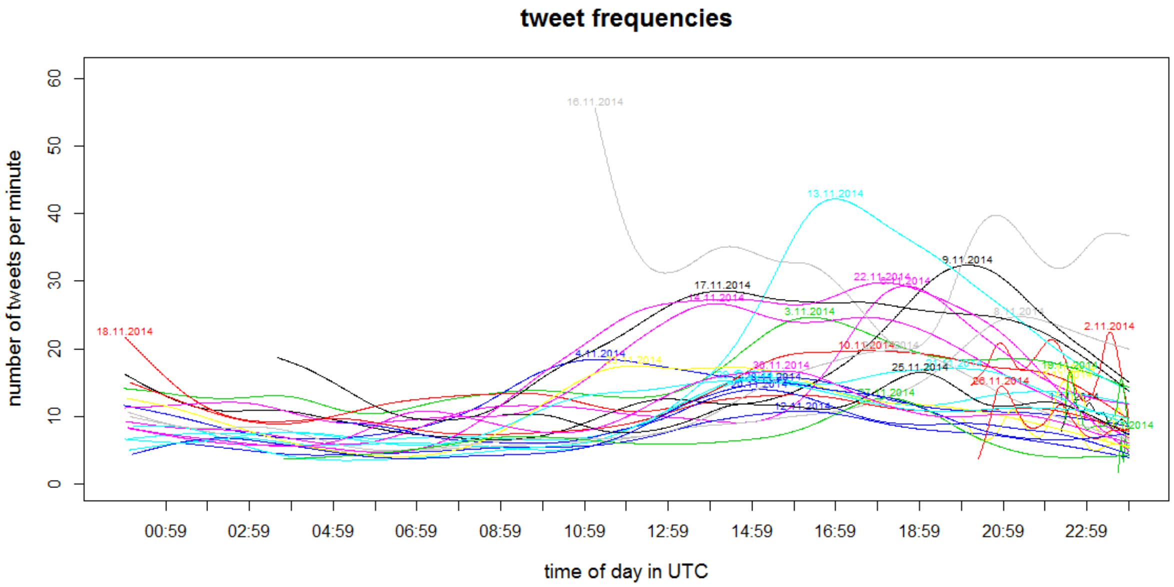


Figure 5.1: Daytime Patterns of Islamic State tweets for all days of November 2014

First of all that we see that not all days are covered around the clock, by looking at Figure 5.1. That is because of the cancellation error in the fetching routine described in Section 2.5. By looking at Figure 5.2 we see the number of extracted tweets per day in November 2014. There are several days with only 1,000 extracted tweets and on November 11, 2014 only 80 tweets could be extracted. This is a result of the abortion error and we should exclude these days from the analysis, as the underlying data for them is insufficient. Therefore, we suspend the days which had less than a 12 hour coverage, i.e. we remove days on which the cancellation error occurred between 12:00:00 and 23:59:59. Table 5.1 shows the excluded days and their respective coverage of the day.

date	01.11.14	02.11.14	07.11.14	11.11.14	15.11.14	19.11.14	26.11.14
coverage [hours:minutes]	1:38	3:47	1:35	0:16	4:01	1:33	3:36

Table 5.1: Insufficient coverage in November 2014

After excluding the days stated above we take a look at Figure 5.3 which shows the tweet patterns without the insufficient days.

The second finding is that the daily peak of the tweets is between 15:00 and 21:00 UTC, which stands for Coordinated Universal Time and corresponds to the time of Central Europe. This looks reasonable as people are more active in the internet in the afternoon and in the evening. One could also conclude that although the majority of tweets in general is set off by users located in the United States, most tweets concerning this topics are situated in Europe or rather in the Middle East as these are the mainly affected regions. This would explain why the peak is in the evening at UTC, or even earlier, which coincides with evening in Europe and the Middle East respectively and not the *American evening*.

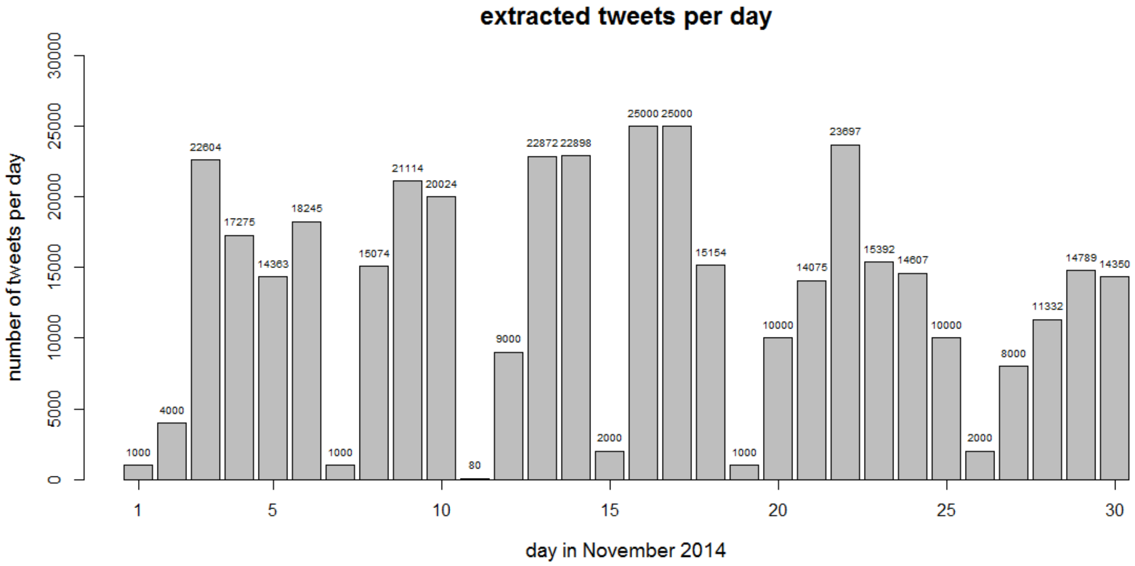


Figure 5.2: Number of extracted tweets in November 2014

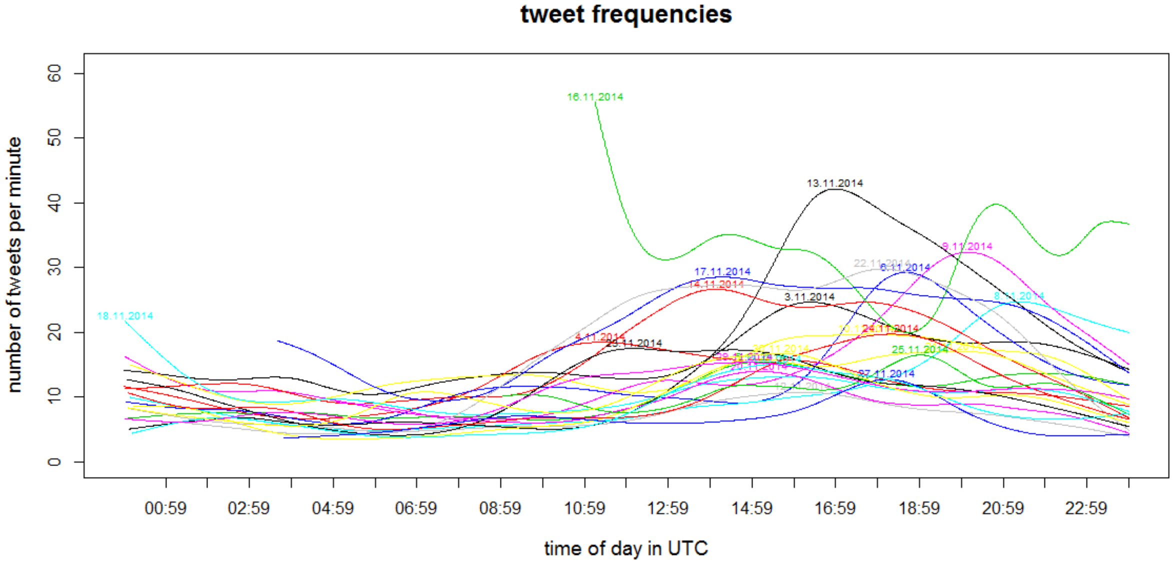


Figure 5.3: Daytime Patterns of Islamic State tweets for complete days of November 2014

5.2 Tweet Activity Outliers

We now have a closer look at the single days in order to discover time points of extreme tweet activity and verify the terms responsible for the peaks against online headlines. First, we detect the outliers in the tweet activity of each day as introduced in Definition 4.7 and then treat the tweets of these time points with the text mining routines described in Chapter 3. The resulting unweighted document-term matrix gives us the most frequent terms of the time points with high tweet activity. The code which was implemented for this purpose is given in the Appendix B.3. We now examine two exemplary days to see if our approach is promising.

Example 5.1. *Tweet Activity Outliers on November 16, 2014*

By looking at the boxplot 5.4 of the tweet frequencies per minute on November 16, 2014, we see several outliers, two extreme ones among them with a tweet activity of 200 per minute.

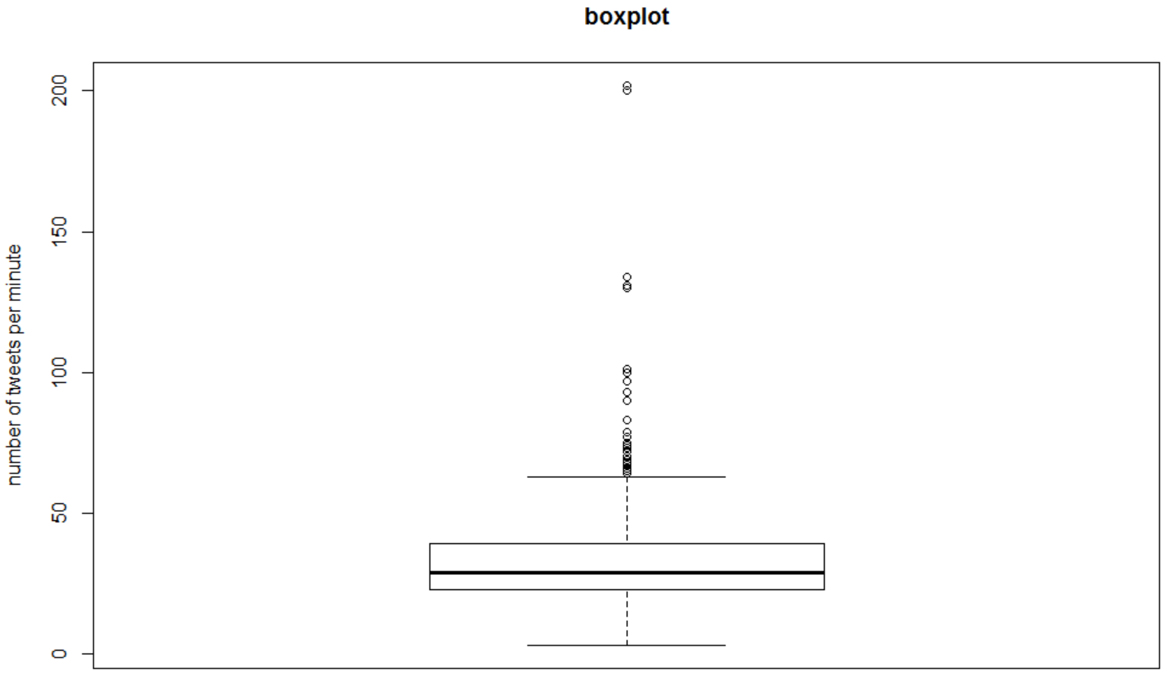


Figure 5.4: Boxplot of Islamic State tweets per minute on November 16, 2014

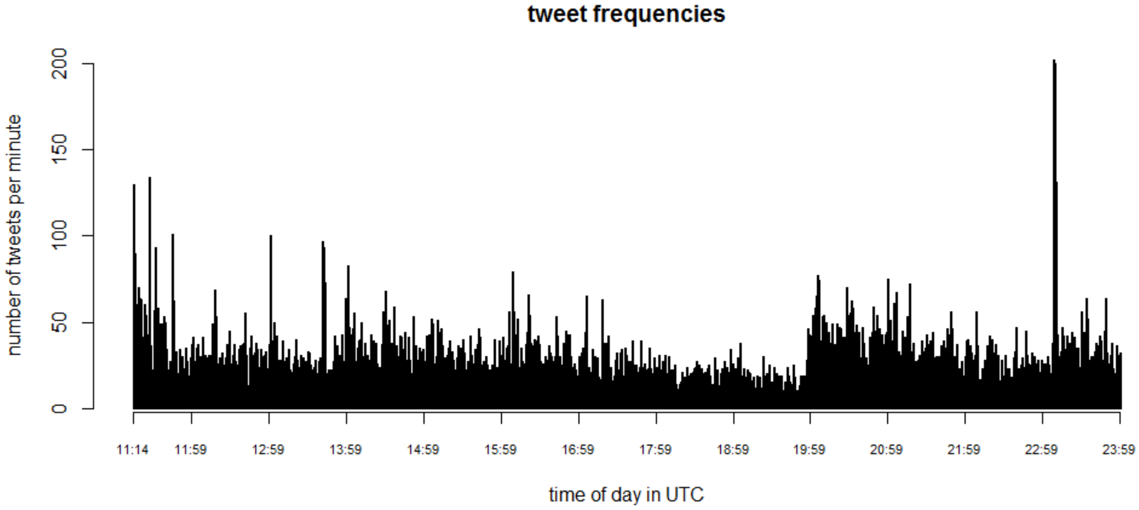


Figure 5.5: Frequency of Islamic State tweets per minute on November 16, 2014

```
> # get time of outliers
> sum[[16]][ind[[16]]]
```

```
11:14 11:15 11:18 11:19 11:26 11:31 11:44 12:17 13:00 13:41
  130    90    70    64   134    93   101    69   100    97
13:42 13:43 13:59 14:00 14:30 16:08 16:20 17:05 20:04 20:05
   93    73    64    83    68    79    66    65    65    77
20:06 20:27 20:59 21:06 21:16 23:08 23:09 23:10 23:33 23:48
   74    70    75    67    72   202   200   131    64    64
```

After examining the bar plot and the times of the outliers we see that the extreme outliers happened in the late hours of this day, between 23:08 and 23:10 UTC. By looking at the most frequent terms contained in the tweets which were set off in the outlier minutes we can see which terms triggered them. The ten most frequent terms in the outlier minutes of November 16, 2014 are given below.

```
> toptermfreq[[16]]
```

```
   kassig     video        us     peter    claims  beheaded
     1963      1903      1588      1118       983       897
      aid    worker beheading  released
      876       857       749       612
```

We now try to verify what happened via online headlines by searching the most frequent terms *kassig, video, us, peter, claims, beheaded, aid, worker, beheading* and *released* and find, among others, an article on the online presence of CNN [14] and one on CBS [15] both verifying the beheading of the U.S. citizen and aid worker Peter Kassig carried out by followers of the Islamic State. Both articles contain all the top ten terms stated above.

The articles were published on November 17 at 3:06 UTC and November 16 at 21:54 UTC respectively, verifying the content of the tweets and showing the very fast spread of information on Twitter.

Example 5.2. *Tweet Activity Outliers on November 8, 2014*

On November 8, 2014 we see again several outliers we want to examine.

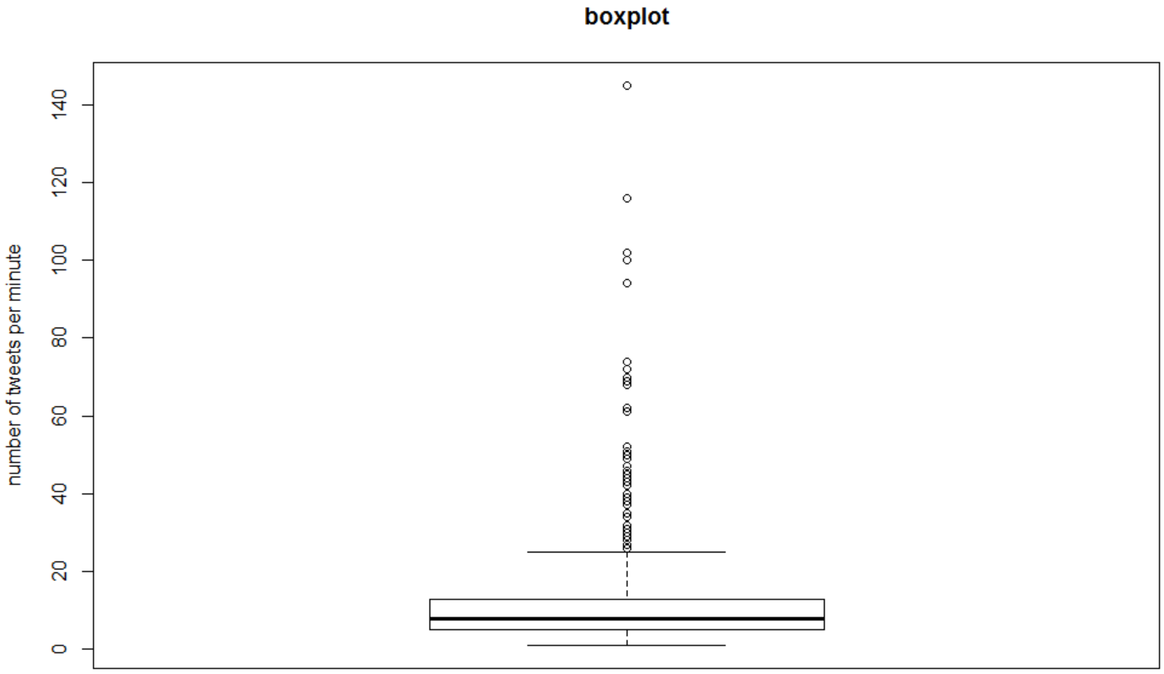


Figure 5.6: Boxplot of Islamic State tweets per minute on November 8, 2014

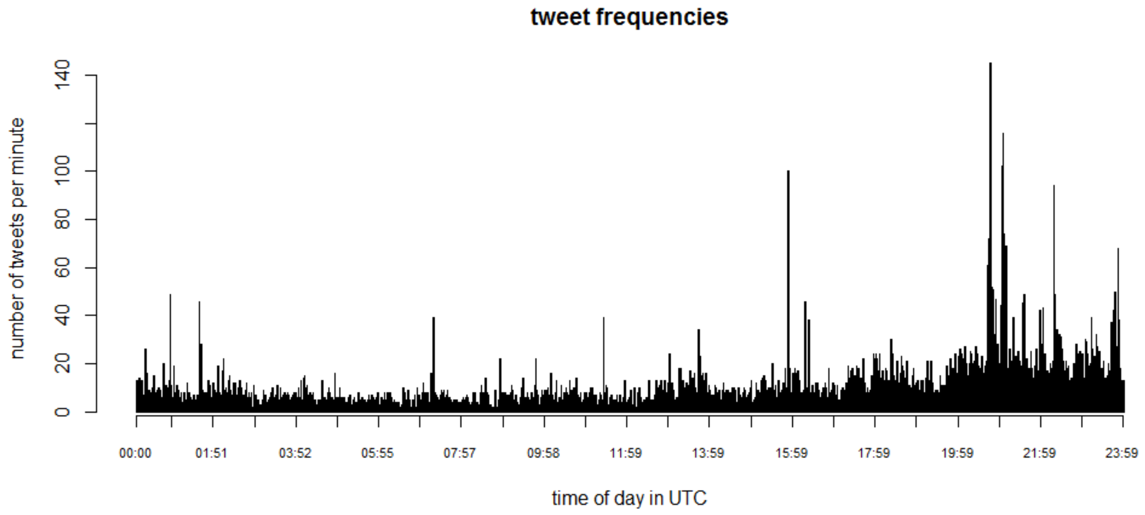


Figure 5.7: Frequency of Islamic State tweets per minute on November 8, 2014

```
> # get time of outliers
> sum[[8]][ind[[8]]]
00:13 00:49 01:31 01:34 07:17 11:25 13:43 15:53 16:18 16:23
   26    49    46    28    39    39    34   100    46    38
18:22 20:02 20:03 20:09 20:24 20:25 20:42 20:43 20:44 20:45
   30    26    26    27    26    27    35    61    72    70
20:46 20:47 20:48 20:49 20:50 20:51 20:52 20:53 20:54 20:55
  145    61    52    46    51    35    32    27    47    26
20:57 21:02 21:03 21:04 21:05 21:06 21:07 21:08 21:09 21:14
   28    44    35   102   116    74    38    62    69    26
21:20 21:33 21:34 21:35 21:36 21:53 21:58 21:59 22:01 22:03
   39    38    45    49    40    26    42    35    28    43
22:18 22:19 22:20 22:21 22:23 22:26 22:27 22:28 22:29 22:30
   94    49    29    26    34    32    29    31    27    26
22:51 23:04 23:05 23:13 23:15 23:20 23:21 23:41 23:42 23:45
   28    30    29    39    26    32    27    28    37    42
23:47 23:48 23:50 23:51 23:52 23:53
   50    26    27    68    31    38
```

We see that most of the high activity of this day happened in the last hours of the day, starting at 20:00 UTC. By looking at the most frequent terms in those hours we can see which terms were responsible for the increase of tweets in this time interval.

```
> toptermfreq[[8]]
  strikes       air        us    target      iraq gathering
     2593      2070      1931      1591      1522      1220
  leaders     mosul      near    convoy
     1053       936       927       861
```

We find an article on the online presence of The Guardian that confirms that U.S. air strikes near Mosul destroyed a convoy of the Islamic State and that it is possible that high leaders of the movement were killed, see [16]. The article was published on November 9 at 9:38 UTC, several hours after the news were spread on Twitter. The most frequent terms of the time points of high activity of every day and their corresponding online article with the time it was published, is given in the Appendix A.3.

So already from this first analysis we get an insight which topics and news are most interesting for the majority of the people on Twitter. We are also able to verify the validity of those statements by double-checking with online articles of big newspapers. From this we can conclude that the information gained from Twitter is valid and very topical, although an uniform database for news, like Reuters or Bloomberg would be optimal in order to certify the results from Twitter and is addressed to further research.

5.3 Development of most frequent terms

We can also examine the most frequent terms of every day and have a look on when they arose and vanished. This is a good method to see what the hot topics have been and for how long they had the attention of the public interest.

Example 5.3. *Example 5.1 continued - Development of the most frequent terms of November 16, 2014*

For this reason we have a look again at the days the Kassig incident happened.

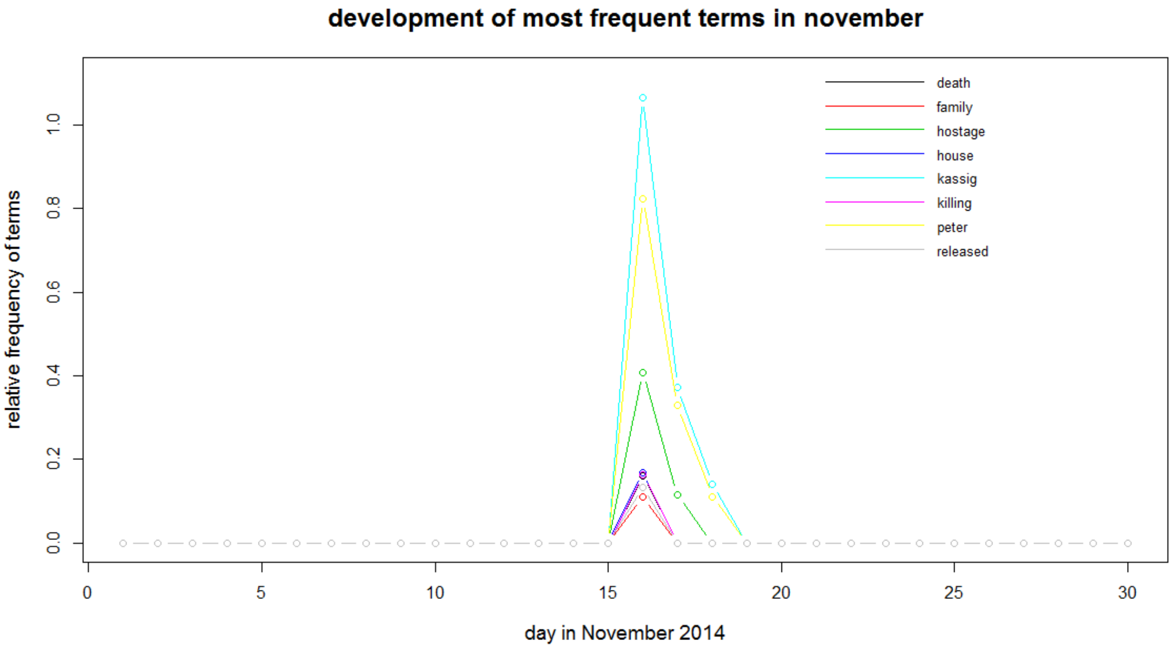


Figure 5.8: Development of several top terms of Islamic State tweets in November 2014 I

The y-axis shows the occurrence of the terms in relation to all Islamic State tweets of the respective day. As we can see the term *kassig* has a relative frequency larger than one. This is possible as a term can occur more than once in a tweet. Like in Section 5.2 we can see how the drama of the beheaded U.S. hostage Peter Kassig got spread over the world. But even such shocking news only stay in the focus of people for a few days, as we can see that the Kassig incidence disappears, at least relatively to other news, after two days of high attention on November 19.

Example 5.4. *Development of the most frequent terms of November 2014*

But there are also examples of terms which are mentioned very often over the whole month, which can be seen in Figure 5.9. General words like *children, iraq* and *group* seem never to lose their importance. As Iraq is the country in which the majority of the Islamic State was founded and many children suffer from the actions of this radical group, these words describe general concerns and situations and are therefore mentioned all the time in the tweets.

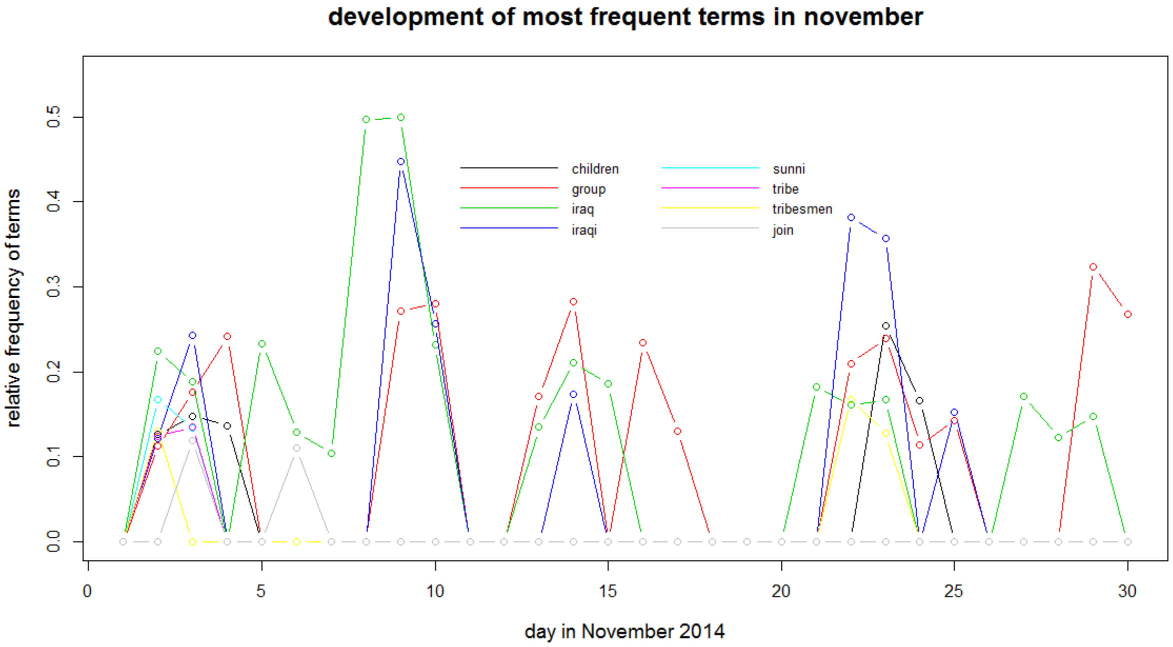


Figure 5.9: Development of several top terms of Islamic State tweets in November 2014 II

One can argue that most of the top terms are generated via retweets or simply the repetition of an already existing tweet and not from independent sources and people. But that should not be a problem because first of all a retweet is not necessarily a wrong information and secondly we verify the validity of the information via online headlines on the respective days. Furthermore, a retweet still indicates information of high interest.
The plots of all frequent terms and their development over the days of November 2014 are given in the Appendix C.1. The code which treats all tweets with the text mining routine described in Chapter 3, finds the most frequent terms and plots their development in

November 2014 is given in Appendix B.4.
Also after excluding the retweets from our data, the majority of the tweets of a certain day treat only a few subjects and therefore have many terms in common. The following summary states how much of the original data, where the retweets were excluded, is necessary to cover 95 % of the terms when all of the data is taken into account.

```
> summary(95%terms)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.06623 0.15930 0.27210 0.32620 0.37820 1.00000
```

We see that, under the assumption of a uniform distribution of the tweets over the daytime, in average 33 % of the data is sufficient to cover most of the terms of a certain day. This is a very promising result when we want to classify new tweets, as we can stop collecting data after eight hours, perform the Principal Component Analysis with the data collected so far and then categorize new tweets sufficiently accurate for the rest of the day.
The code which analyzes how much of the data is needed to cover 95 % of all terms on this day is given in the Appendix B.5.

5.4 Principal Component Analysis

In this section we will leave the classical and straight forward methods to analyze data behind us and set the focus on more advanced methods like the Principal Component Analysis introduced in Chapter 4. We want to construct new variables, reduce their quantity, find hidden patterns and create informative clusters from the extracted tweets. Furthermore, we want to classify new tweets with the help of the before calculated components and the detected clusters.
For this reason, we take all available tweets containing the words Islamic State of every day in November 2014 and then proceed analogously to Example 4.1. Before we apply the text mining routines, we remove all retweets as they do not contain any additional information. After retrieving the document-term matrix Y, we calculate the standardized data matrix Z from it and test the columns, e.g. terms, on independence. To ensure independence we delete columns till the new matrix $\tilde{Z}$ has full rank. Then the empirical correlation matrix $R = \tilde{Z}'\tilde{Z}$, $rank(R) = rank(\tilde{Z})$, is calculated for each of the thirty days of November 2014. The overview in Figure 5.10 shows the different steps and matrices in order to clarify the procedure.

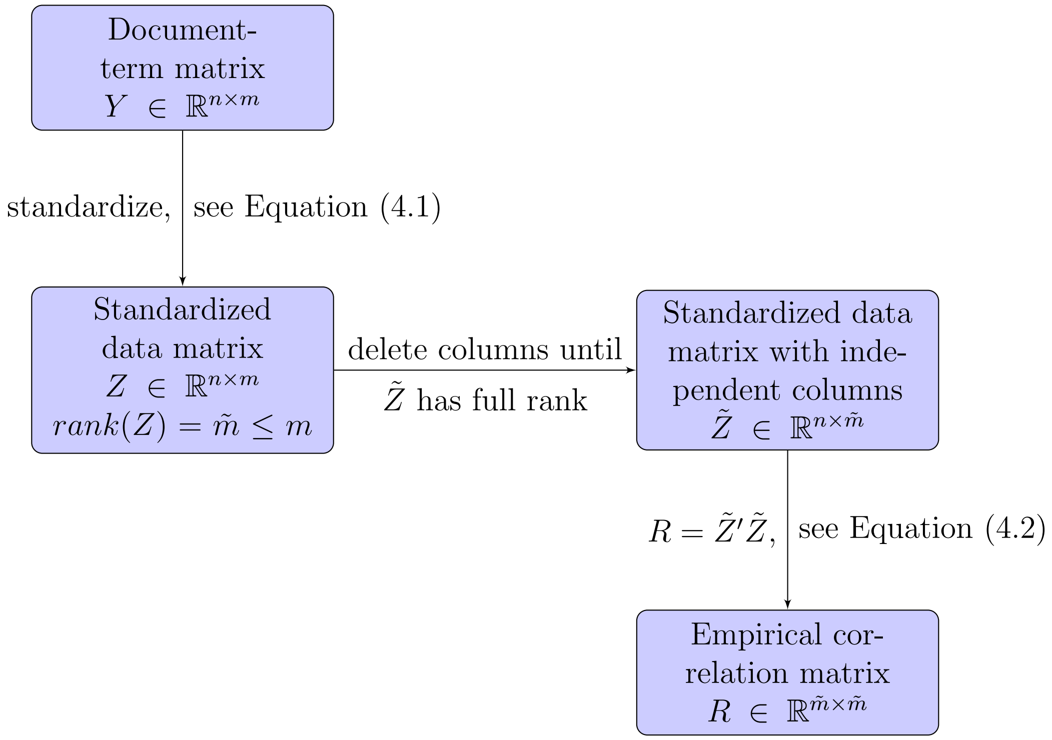

Document-term matrix $Y \in \mathbb{R}^{n\times m}$
standardize, see Equation (4.1)
Standardized data matrix $Z \in \mathbb{R}^{n\times m}$ $rank(Z) = \tilde{m} \leq m$
delete columns until $\tilde{Z}$ has full rank
Standardized data matrix with independent columns $\tilde{Z} \in \mathbb{R}^{n\times \tilde{m}}$
$R = \tilde{Z}'\tilde{Z}$, see Equation (4.2)
Empirical correlation matrix $R \in \mathbb{R}^{\tilde{m}\times \tilde{m}}$

Figure 5.10: Procedure of the data matrix treatment

Table 5.2 and Figure 5.11 show m the number of columns of Z, i.e. the number of terms on this day and $\tilde{m}$ the number of columns of $\tilde{Z}$, i.e. the number of independent terms of this day.

date	11/03/14	11/04/14	11/05/14	11/06/14	11/08/14	11/09/14	11/10/14	11/12/14
m	129	133	147	141	121	127	153	170
$\tilde{m}$	129	133	144	140	121	124	153	162

date	11/13/14	11/14/14	11/16/14	11/17/14	11/18/14	11/20/14	11/21/14	11/22/14
m	173	164	109	144	150	140	135	135
$\tilde{m}$	172	164	109	144	149	139	135	133

date	11/23/14	11/24/14	11/25/14	11/27/14	11/28/14	11/29/14	11/30/14
m	134	143	128	115	136	145	164
$\tilde{m}$	133	138	126	112	136	142	160

Table 5.2: Dimensions of Z and $\tilde{Z}$

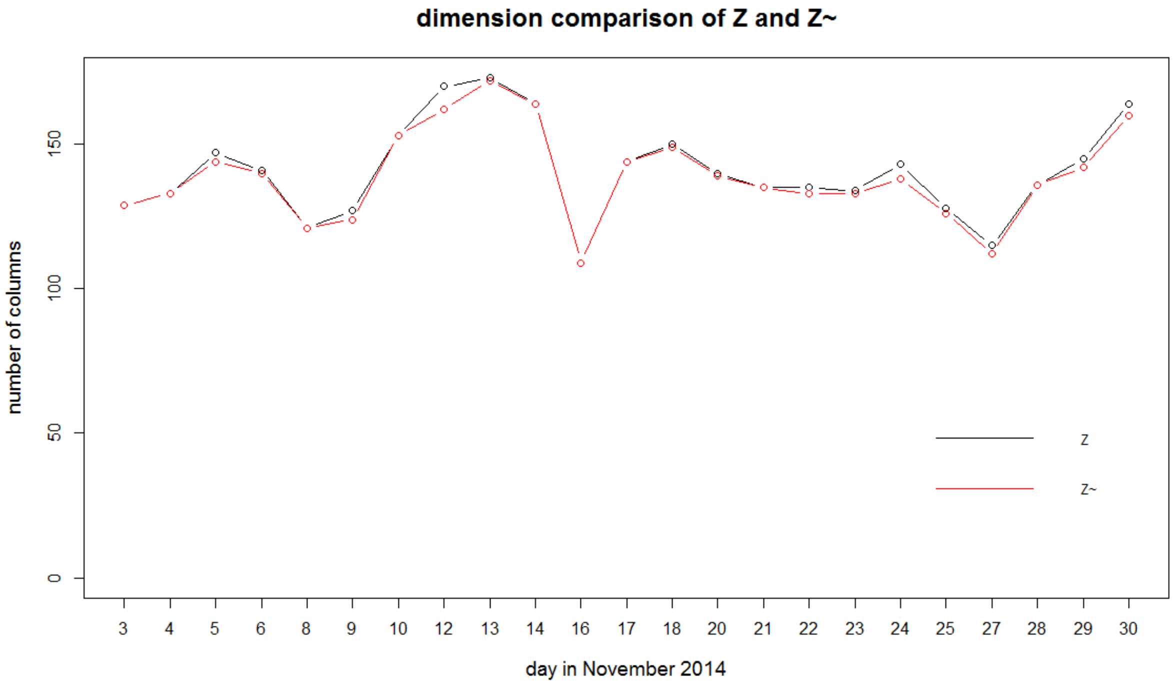


Figure 5.11: Dimension of the columns of Z and $\tilde{Z}$

The difference $m - \tilde{m}$ of the number of columns between Z and $\tilde{Z}$ is very small. This means that most of the terms of one day are linearly independent. As the summary below states, only 1 % of the columns had to be excluded in average, in order to achieve linear independent columns.

```
> summary(m~/m)

   Min.  1st Qu.  Median    Mean  3rd Qu.    Max.
 0.9529  0.9795  0.9929  0.9886  1.0000  1.0000
```

Then we test if the columns of $\tilde{Z}$ are normally distributed via the Kolmogorov-Smirnov test introduced in Definition 4.18, which would be optimal but not necessary to perform a Principal Component Analysis and furthermore test if $\tilde{Z}$ is multivariate normally distributed via Mardia's test, refer to Definition 4.19, as this is an assumption for Bartlett's sphericity test defined in Definition 4.20. As for none of the thirty days of November 2014 the columns of $\tilde{Z}$ are normally distributed, nor is the standardized data matrix $\tilde{Z}$ multivariate normally distributed and in addition our sample size is very large, we focus on the KMO index, as stated in Definition 4.23 and not on Bartlett's sphericity test in order to assess if a Principal Component Analysis makes sense.
Table 5.3 and Figure 5.12 show the KMO index for each day. As the index exceeds 0.5 for every day, we suggest that performing a Principal Component Analysis on our data could be useful.

date	11/03/14	11/04/14	11/05/14	11/06/14	11/08/14	11/09/14	11/10/14	11/12/14
KMO	0.818	0.809	0.808	0.831	0.808	0.746	0.792	0.788

date	11/13/14	11/14/14	11/16/14	11/17/14	11/18/14	11/20/14	11/21/14	11/22/14
KMO	0.778	0.802	0.765	0.722	0.800	0.819	0.791	0.785

date	11/23/14	11/24/14	11/25/14	11/27/14	11/28/14	11/29/14	11/30/14
KMO	0.794	0.801	0.795	0.822	0.818	0.778	0.822

Table 5.3: KMO index of each day of November 2014

```
> summary(KMO)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.7217  0.7869  0.7996  0.7953  0.8134  0.8313
```

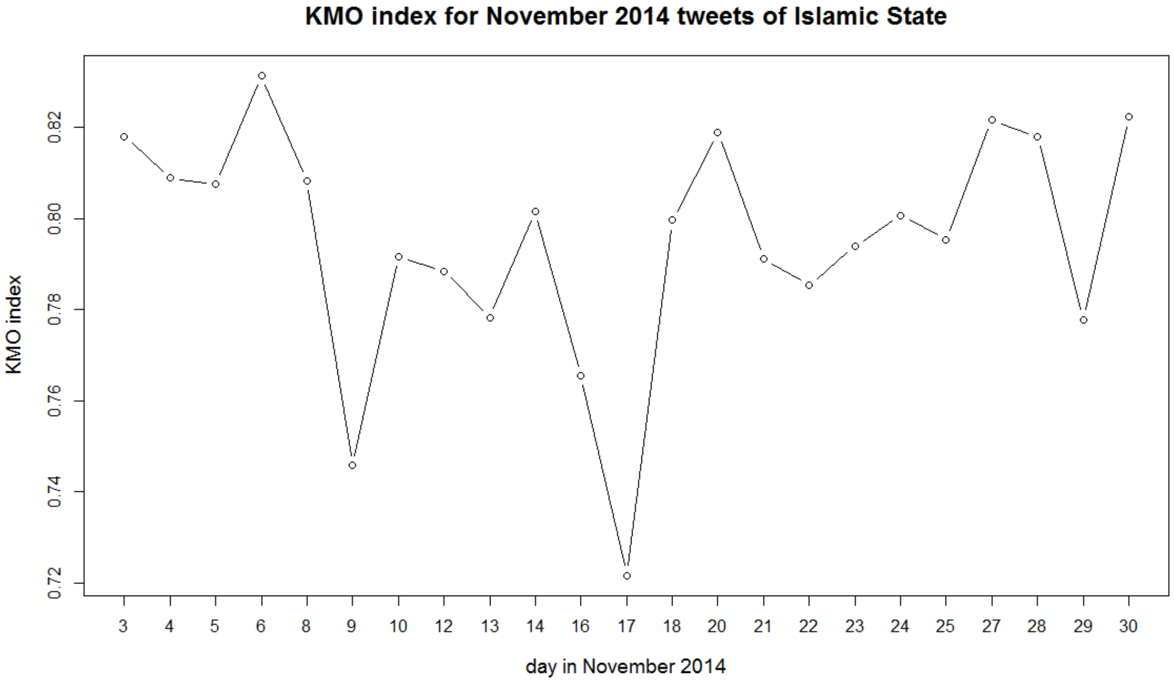


Figure 5.12: KMO Index of Islamic State tweets in November 2014

We then calculate the matrix of principal components F and the loading matrix L as described in Chapter 4.

Finally, we have to find k, the number of components to keep. By choosing

$$\max_{k\in\{1,\dots,\tilde{m}\}} \lambda_k \geq 1,$$

where $\lambda_1, ..., \lambda_{\tilde{m}}$ are the eigenvalues of R, we yield k. We also apply the Scree test as defined in Definition 4.25 to obtain a second proposal for k and then decide based on these two suggestions.

Example 5.5. *Example 5.3 continued - Choice of k on November 16, 2014*

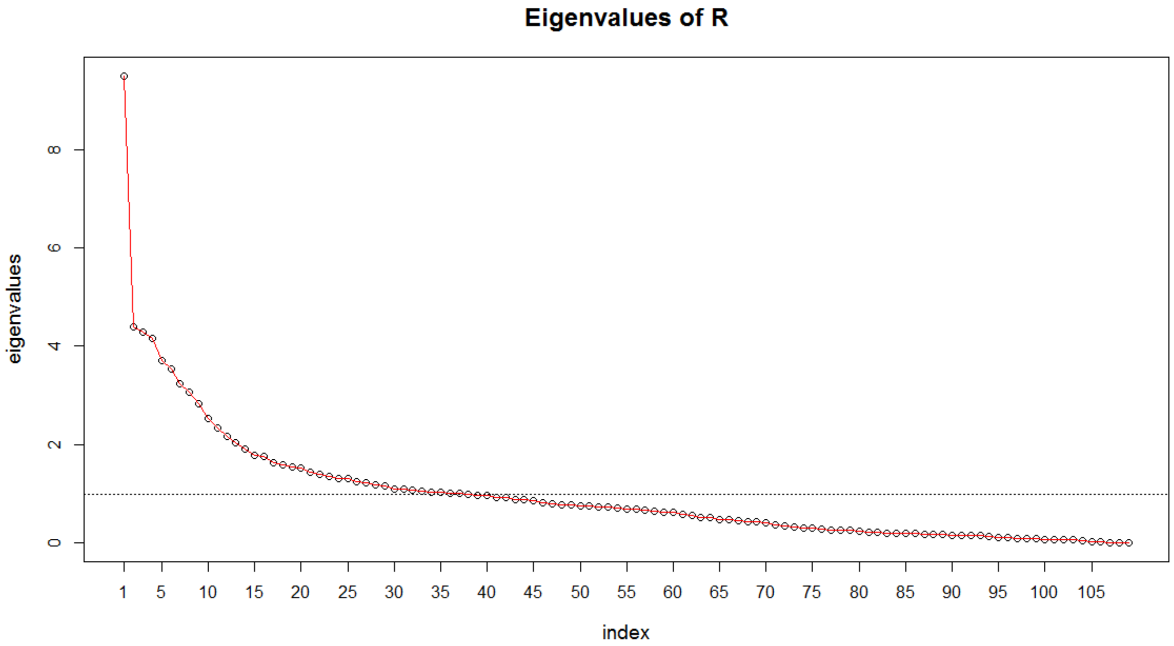


Figure 5.13: Ordered eigenvalues of empirical correlation matrix R of November 16, 2014

Figure 5.13 shows the ordered eigenvalues of the empirical correlation matrix R of November 16, 2014, where 37 of them are larger than 1, but we choose $k = 15$ after considering the Scree test. The 16. of the eigenvalues is 1.760 and therefore just slightly larger than one.

Table 5.4 shows k, the number of eigenvalues larger than 1, Scree which stands for k suggested by the Scree test and $\frac{k}{\tilde{m}}$, so the ratio of k to the number of independent terms $\tilde{m}$ on this particular day, for each day of November 2014.

```
> summary(k/m~)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.2786  0.2868  0.2946  0.2986  0.3060  0.3394
```

date	11/03/14	11/04/14	11/05/14	11/06/14	11/08/14	11/09/14	11/10/14	11/12/14	11/13/14	11/14/14	11/16/14	11/17/14	11/18/14	11/20/14	11/21/14	11/22/14	11/23/14	11/24/14	11/25/14	11/27/14	11/28/14	11/29/14	11/30/14
k	37	40	41	39	35	40	45	51	50	47	37	46	45	43	41	41	40	40	38	33	39	40	45
Scree	29	28	30	20	19	25	30	35	34	30	15	37	32	24	28	27	22	20	19	17	20	22	20
$\frac{k}{\tilde{m}}$	0.287	0.301	0.285	0.279	0.289	0.323	0.294	0.315	0.291	0.287	0.339	0.319	0.302	0.309	0.304	0.308	0.301	0.290	0.302	0.295	0.287	0.282	0.281

Table 5.4: Determining the number of components to keep k

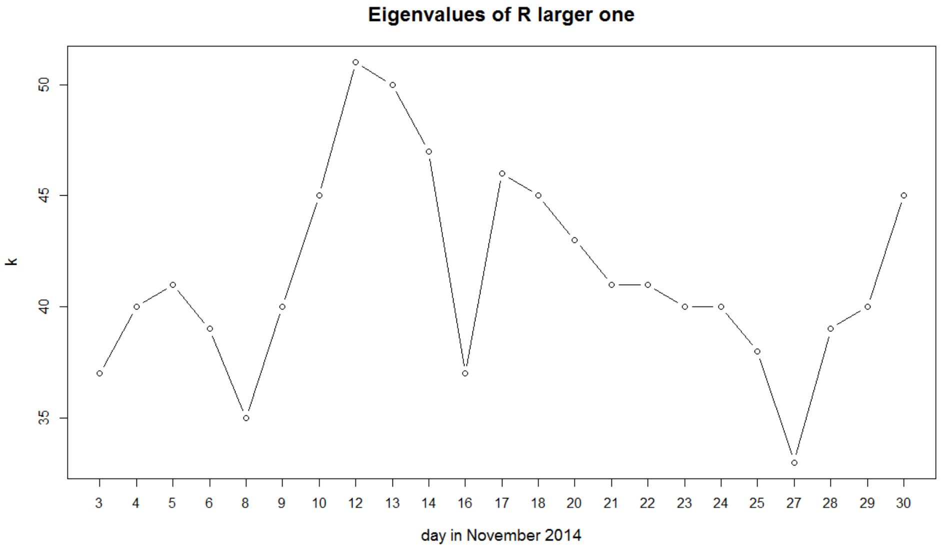


Figure 5.14: Eigenvalues of R larger one in November 2014

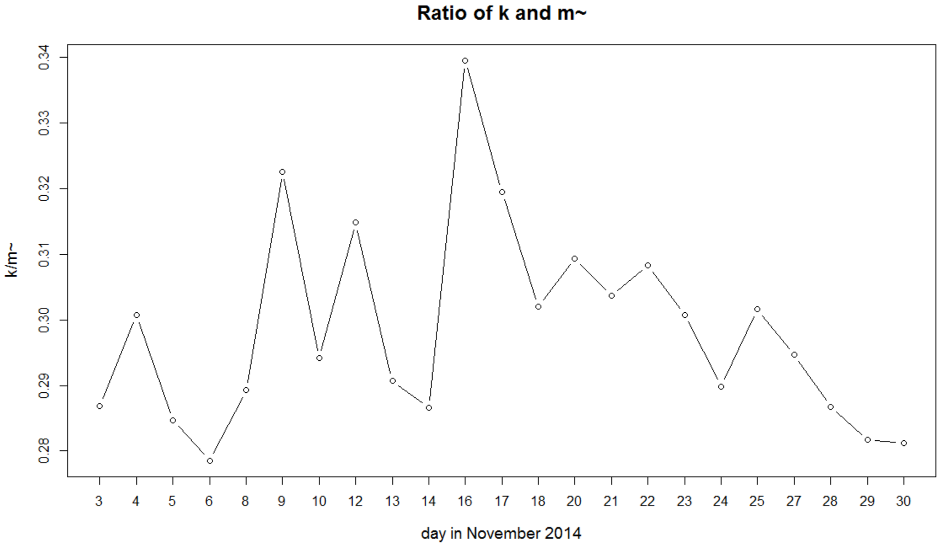


Figure 5.15: Ratio of k and $\tilde{m}$ of Islamic State tweets in November 2014

The ratio $\frac{k}{\tilde{m}}$ stays the same for most of the days, indicating that the number of necessary components for explaining the data sufficiently, is roughly 30 % of the number of all terms on this day. We then obtain the reduced matrix of the principal components F_k and the reduced loading matrix L_k, as stated in Theorem 4.24.

Example 5.6. *Example 5.5 continued - Factor 1 and Factor 2 on November 16, 2014*

For the purpose of getting an overview and for the graphical clustering, we now focus on the first two factors. A summary of the first factor, i.e. the first column of the reduced matrix of principal components F_k on November 16, is given below:

```
> summary(F_1)

      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-0.0027340 -0.0020980 -0.0016260  0.0000000 -0.0006399  0.0531800
```

We see that most of the first principal coordinates of the tweets, i.e. entries of the first column of F_k, are around zero but there are also some higher values in the positive direction. The high loadings of the first and the second factor, so all entries of the first and the second column of the reduced loading matrix L_k on November 16, which are larger than 0.5 are:

First column (First factor):

```
[1] "#demandforaction" "@kinonuri"        "criteria"         "denying"
[5] "dishonor"         "doesn_t"          "genocide"         "recognize"
[9] "sweden"           "un"               "victims"
```

Second column (Second factor):

```
[1] "strong" "video"
```

We see, that the most frequent terms on this day like for example *peter*, *kassig*, *video* or *beheading* are not the most dominant terms with the highest loadings on the first factor. In fact the terms which have high loadings are treating a totally different subject. By verifying them against online headlines we see that there was another headline one day before containing all the terms which load high on the first factor, see [17]. The article criticizes Sweden and the United Nations for not recognizing the actions of the Islamic State as genocide. Therefore the first and most important factor clusters the tweets regarding if they do not treat the 'Kassig incident' but therefore belong to the other topic.

```
#demandforaction            #iraq            #isis    #islamicstate
         0.97546         -0.00460         -0.01438         -0.01100
           #news           #syria        @kinonuri              aid
        -0.03525         -0.00926          0.56390         -0.14173
        american        americans          another               ap
        -0.07330         -0.02479         -0.02289         -0.03615
         appears             army        beheaded            behead
        -0.02366         -0.02530         -0.14448         -0.01707
       beheading          beheads           beirut            bring
        -0.09006         -0.06795         -0.03815         -0.00603
         cameron          claimed           claims         condemns
        -0.01229         -0.01685         -0.14861         -0.03452
          confir         confirms         criteria            death
        -0.03926         -0.05825          0.98052         -0.04280
         denying         depraved         dishonor          does_t
         0.98194         -0.00966          0.93921          0.89245
         evil          family           fight        fighting
     -0.01886        -0.05736        -0.01043        -0.01655
        force          former        genocide         graphic
     -0.01452        -0.02033         0.94011        -0.04131
        group         hostage           house        internet
     -0.07380        -0.11452        -0.07173        -0.03822
         iraq            isil            isis           islam
     -0.02284        -0.00365        -0.02927        -0.00666
       jihadi        jihadist          kassig         kassigs
     -0.00759        -0.01779        -0.21114        -0.02095
       killed         killing         kurdish          latest
     -0.04705        -0.04701        -0.01186        -0.00811
       leader       militants          murder             new
     -0.01215        -0.04476        -0.02177        -0.03754
         news           obama          people           peter
     -0.02400        -0.05049        -0.00896        -0.14861
         post          posted       president            pure
     -0.01979        -0.02056        -0.02795        -0.01781
     purports          ranger       recognize        released
     -0.02696        -0.01812         0.94228        -0.07779
     releases        responds         reuters          review
     -0.02046        -0.05632        -0.04407        -0.03916
           rt            said            says            show
     -0.01294        -0.02941        -0.04216        -0.05113
      showing           shows        soldiers         state_s
     -0.02063        -0.01952        -0.00959        -0.03344
    statement          states          strong          sunday
     -0.04892        -0.01927        -0.01213        -0.05368
       sweden           syria          terror          threat
      0.91528        -0.02093        -0.00701        -0.00611
        times           today          troops              uk
     -0.01988        -0.02786        -0.00983        -0.01025
          uks              un              us             usa
     -0.01043         0.90173        -0.19863        -0.02551
      veteran             via         victims           video
     -0.01843        -0.04835         0.92631        -0.17582
      warning      washington           white            will
     -0.00789        -0.01775        -0.07168        -0.00657
       worker
     -0.14139
```

If we have a closer look at the first column of the reduced loading matrix L_k, given above, we see that all ‘Kassig incident‘ related words have a negative sign. We therefore expect the majority of the tweets of this day to be around zero and become more and more positive the less they have to do with the beheading of Peter Kassig. The second factor has its highest loading in the term *video*. This is a sign that the second factor clusters the tweets regarding if they treat the video of Peter Kassig's beheading or not. This might indicate that there are general tweets about Peter Kassig, for example concerning his life, his work or his family, and tweets with focus on the tragedy itself.

All high loadings of the first factor of each day of November 2014 are given in the Appendix A.4.

5.5 Clustering of tweets on a single day

Example 5.7. *Example 5.6 continued - Visual clustering with the two main factors of November 16, 2014*

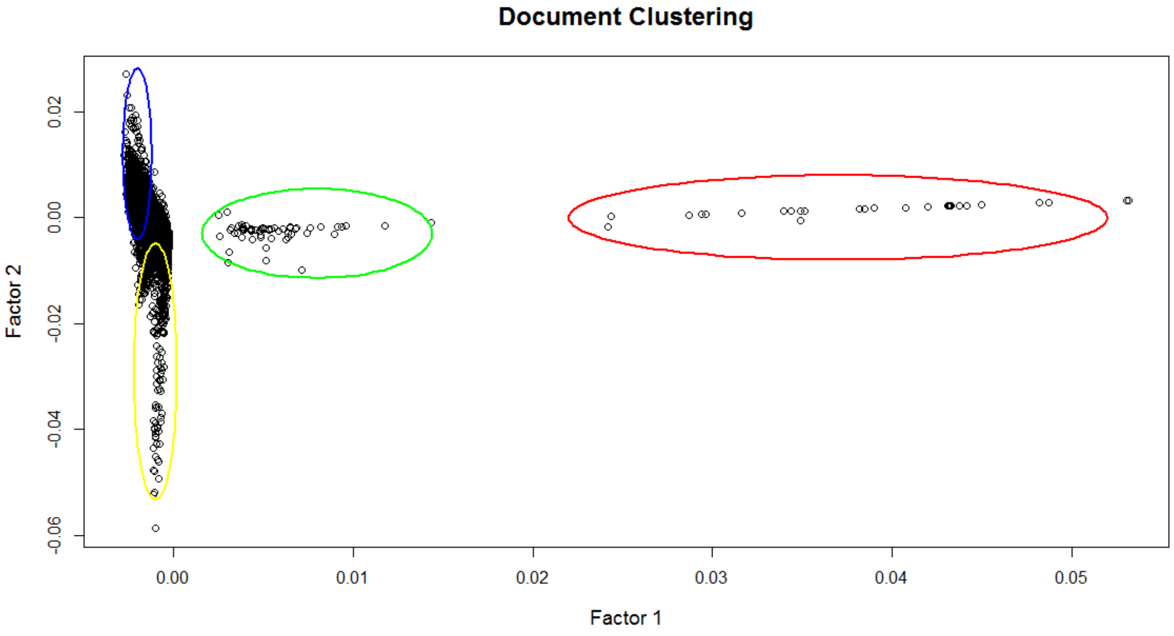


Figure 5.16: Document clustering with the two main factors of November 16, 2014

If we have a look at all tweets of November 16, 2014 plotted in the Factor 1 - Factor 2 plane we can draw several conclusions. First of all, most of the tweets are situated very close around zero of the Factor 1 axis. These are all the ‘Kassig incident‘ tweets. Furthermore, among those we see that there is a spread in the Factor 2 dimension. As mentioned above, the tweets with positive Factor 2 values are the Kassig tweets related to the video of his death, as the loading of video in the second factor has a positive sign, indicated by the blue ellipses and general tweets concerning Kassig with a negative Factor 2 value, indicated by the yellow ellipses. Secondly, we see two further accumulations in the Factor 1 dimension. One is situated between 0.001 and 0.01 and the other between 0.025 and 0.055. The second accumulation, marked by the red ellipses, corresponds to all tweets related to the second topic, i.e. the ‘U.N. genocide‘ topic mentioned above, as the Factor 1 values are the highest. The first accumulation, surrounded by the green ellipses,

are most probably tweets which have nothing to do with either of the two topics.
So we are able to identify four clusters at a first glance:

Identified Clusters

1. ‘Kassig Incident‘
   (a) related to the video of his beheading $\widehat{=}$ blue ellipses
   (b) general tweets related to Peter Kassig $\widehat{=}$ yellow ellipses
2. Tweets treating the ‘Sweden denies genocide‘ topic $\widehat{=}$ red ellipses
3. remaining tweets $\widehat{=}$ green ellipses

With these two factors 12.74 % of the overall variance is captured.

Table 5.5 contains the captured variance by the first two factors for each day of November 2014.

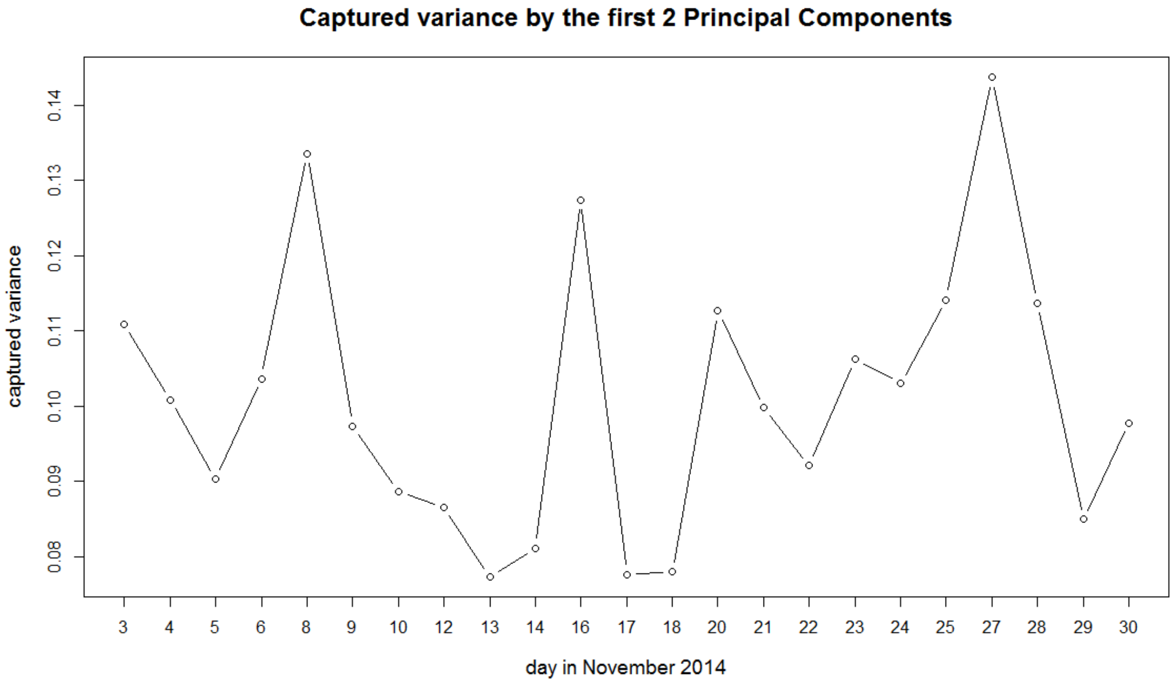


Figure 5.17: Captured variance by the first two Principal Components

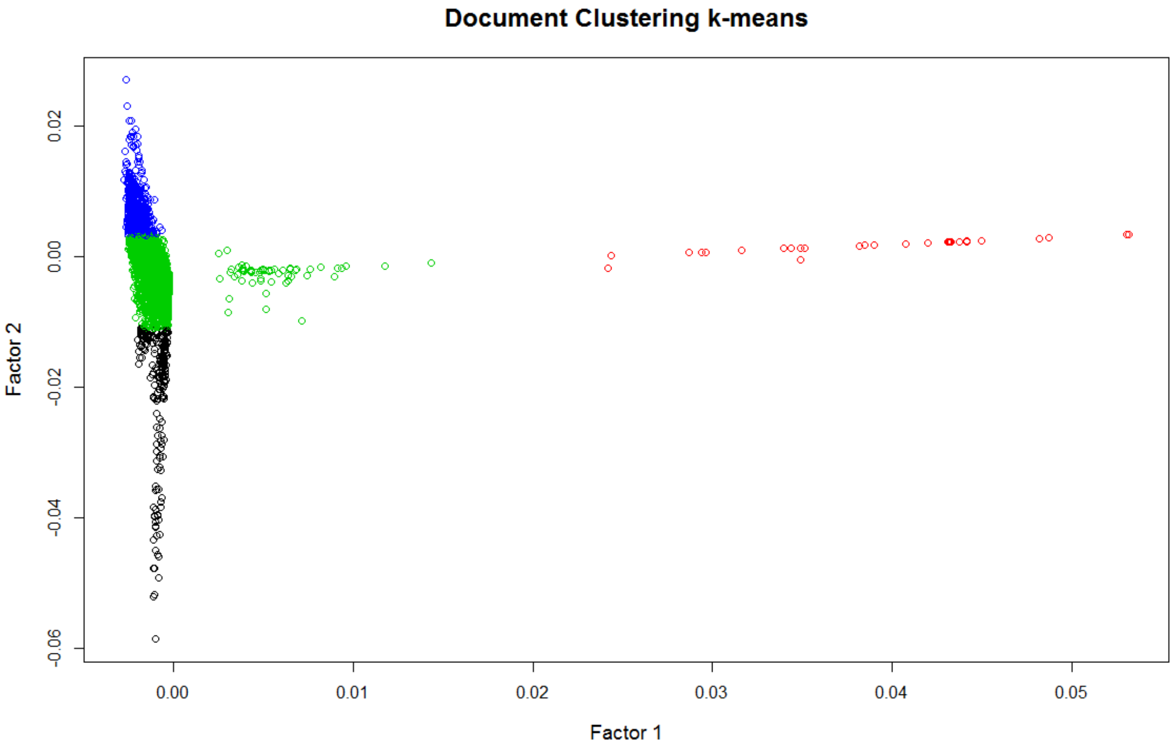
```
> summary(capt.variance)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.07739 0.08758 0.09980 0.10090 0.11170 0.14370
```

As we see from the summary above the first two factors cover 10.1 % of the overall variance in average.

Example 5.8. *Example 5.7 continued - Clustering with the two main factors of November 16, 2014*

The next step is to verify if the intuitive clusters, which we determined by looking at the loading matrix and the 2-dimensional plot above, are valid. Furthermore we want to get more informative clusters automatically. We therefore apply the k-mean clustering algorithm introduced in Algorithm 1 to the 2-dimensional new coordinates of the tweets. So we want to find the k-means cluster of the first two columns of the matrix F. As we have to decide on the number of clusters upfront, we run the algorithm with three, four and five clusters and decide afterwards which number was most suitable. As it turns out, four clusters are most reasonable, which coincides with our first intuitive assessment. Figure 5.18 top shows the tweets of November 16, again plotted in the factor 1 - factor 2 plane, and coloured with respect to the clusters the k-means algorithm detected. We now have a closer look at the four clusters. Therefore, we examine the most frequent words in each cluster to identify them with a topic.

Table 5.6 shows the ten most frequent words and the number of tweets in each cluster on November 16, 2014. Looking at the most frequent terms contained in the black Cluster 1, we can conclude that the majority of them treats the condemnation of the killing of Peter Kassig by the President of the United States of America Barack Obama, as well as by the Prime Minister of the United Kingdom David Cameron. This cluster corresponds to the Cluster 1b above. With the help of the k-mean algorithm we are even able to specify our first classification of these tweets as general tweets about the Kassig incident to a specific topic. The second cluster highlighted by the colour red is the smallest with 473 tweets contained in it. It treats the discussion about the recognition of the actions of the Islamic State as genocide and is the same as Cluster 2 described before. Again this cluster is clearly separated from the rest. The two largest cluster with over 13600 tweets contained in them, both deal with the actual beheading of Peter Kassig. We might suggest that the green Cluster 3, which is by far the largest with 9568 tweets, focuses on the facts of the beheading of the American hostage Peter Kassig, whereas the blue Cluster 4 was the spread of the information that a group of the Islamic State released a video of the beheading and claim that the person is the U.S. aid worker Peter Kassig. These two clusters match with the Cluster 1a established before. Only Cluster 3 from the first intuitive classification above could not be verified with the help of k-means clustering. So in conclusion, we not only verified the first intuitive categorization of the tweets but also made the clustering more specific via the k-means algorithm.



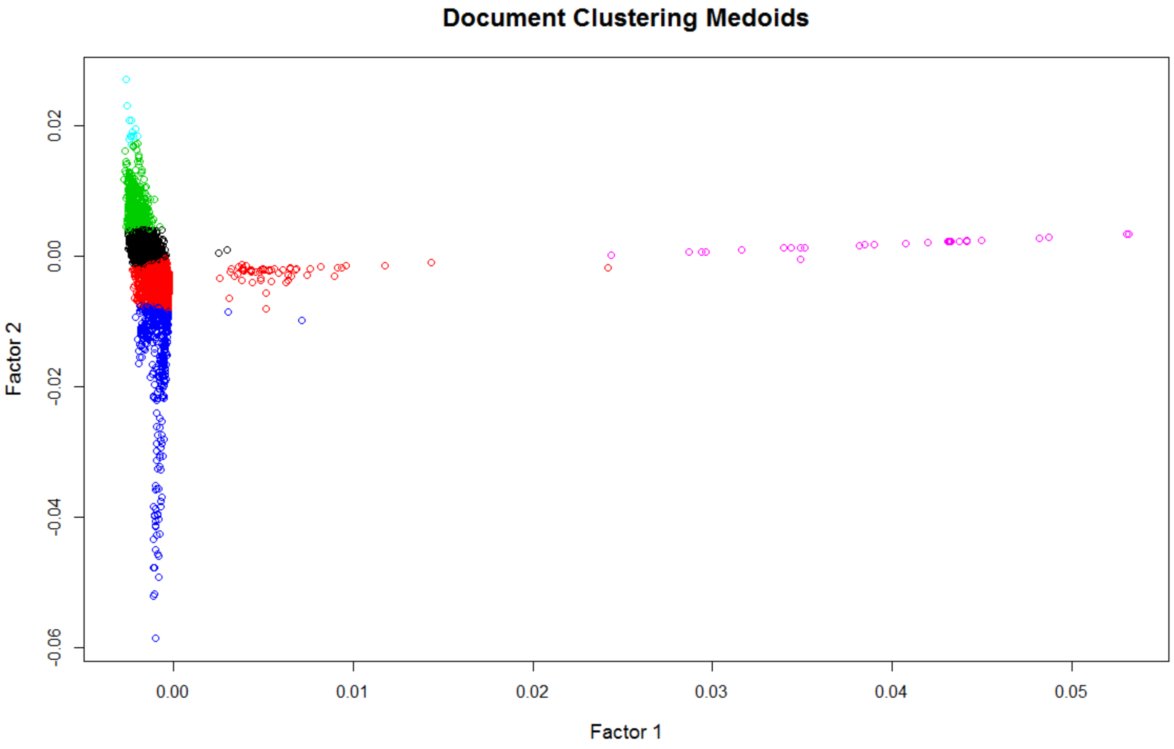


Figure 5.18: Top: Clustering of Islamic State tweets of November 16, 2014 via k-means and two factor
Bottom: Clustering of Islamic State tweets of November 16, 2014 via k-medoids and two factors

date	11/03/14	11/04/14	11/05/14	11/06/14	11/08/14	11/09/14	11/10/14	11/12/14	11/13/14	11/14/14	11/16/14	11/17/14
capt. variance	0.111	0.101	0.0903	0.104	0.134	0.0974	0.0886	0.112	0.0774	0.0811	0.127	0.0776

date	11/18/14	11/20/14	11/21/14	11/22/14	11/23/14	11/24/14	11/25/14	11/27/14	11/28/14	11/29/14	11/30/14
capt. variance	0.078	0.113	0.0998	0.0921	0.106	0.103	0.114	0.144	0.114	0.085	0.0978

Table 5.5: Captured variance by the first two components

	colour	size	TOP 1	TOP 2	TOP 3	TOP 4	TOP 5	TOP 6	TOP 7	TOP 8	TOP 9	TOP 10
Cluster 1	black	882	condemns	obama	killing	kassig	peter	states	cameron	uks	president	depraved
Cluster 2	red	473	denying	criteria	#demandforaction	victims	un	recognize	genocide	doesnot	dishonor	sweden
Cluster 3	green	9568	kassig	us	beheaded	peter	hostage	claims	video	american	aid	worker
Cluster 4	blue	4176	video	us	aid	worker	kassig	claims	beheading	peter	group	released

Table 5.6: Clusters detected by k-means and two factors

	colour	size	TOP 1	TOP 2	TOP 3	TOP 4	TOP 5	TOP 6	TOP 7	TOP 8	TOP 9	TOP 10
Cluster 1	black	4679	kassig	us	claims	beheaded	video	peter	hostage	aid	worker	american
Cluster 2	red	4886	us	kassig	peter	hostage	obama	beheaded	via	isis	claims	american
Cluster 3	green	3456	video	us	aid	worker	kassig	claims	peter	beheading	group	beheaded
Cluster 4	blue	1350	condemns	obama	kassig	killing	peter	states	cameron	president	uks	leader
Cluster 5	cyan	256	house	white	beheading	family	released	responds	video	kassig	statement	sunday
Cluster 6	pink	472	denying	criteria	#demandforaction	victims	un	recognize	doesnot	genocide	dishonor	sweden

Table 5.7: Clusters detected by k-medoids and two factors

Another possibility to classify the extracted Islamic State tweets is the k-medoids algorithm, introduced in Section 4.7 which is more robust but much slower than the k-means. Additionally we include the silhouette technique also introduced in Section 4.7 to detect the optimal number of clusters K. We only briefly look at the similarities and differences in the clustering compared to the one detected by the k-means algorithm. The resulting coloured plot is given in Figure 5.18 bottom. The silhouette technique detects six clusters. We now have a closer look at them and therefore examine the most frequent words in each cluster detected by the medoids algorithm to identify them with a topic. Table 5.7 shows the ten most frequent words and the number of tweets in each cluster of November 16, 2014. As we see the Cluster 1 to 3 are very similar to Cluster 3 and 4 in the k-means clustering. Cluster 6 corresponds to Cluster 2 of the k-means clustering and Cluster 4 to the former Cluster 1, although more tweets have been assigned to this cluster via the k-medoids algorithm. The most striking difference of the k-medoids clustering, in comparison to the k-means clustering, is that it detects an additional cluster containing the tweets which state that the Kassig family responds to the Islamic State beheading video, see [18]. This is remarkable as this cluster is detected despite its small size of just 256 tweets. Cluster 2 was most probably separated from the very similar Cluster 3 because of the term *ISIS*. As this is just another term for Islamic State, this cluster does not contain any additional information. This comparison is supported by Table 5.8 in which we see how many tweets have been assigned to each cluster by the k-means and the k-medoids clustering respectively. So although the main results are the same in both clustering procedures the k-medoids algorithm detects more specific and smaller clusters.

Generally we see that the structure of the data in the new two dimensional coordinates is very similar for most of the examined days by having a look at Figure 5.19. We see that in all of the exemplary days of November, 2014 which are shown, the structure with the distinctive spires occurs. This is an indication that on each of those days there are at least as many different topics as spires and therefore a clustering analogously to the previous section is not only possible but will lead to coherent clusters.

kmedoids\kmeans	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Sum
Cluster 1	0	0	4215	464	4679
Cluster 2	0	1	4885	0	4886
Cluster 3	0	0	0	3456	3456
Cluster 4	882	0	468	0	1350
Cluster 5	0	0	0	256	256
Cluster 6	0	472	0	0	472
Sum	882	473	9568	4176	15099

Table 5.8: Comparison of cluster membership between k-means clustering and k-medoids clustering

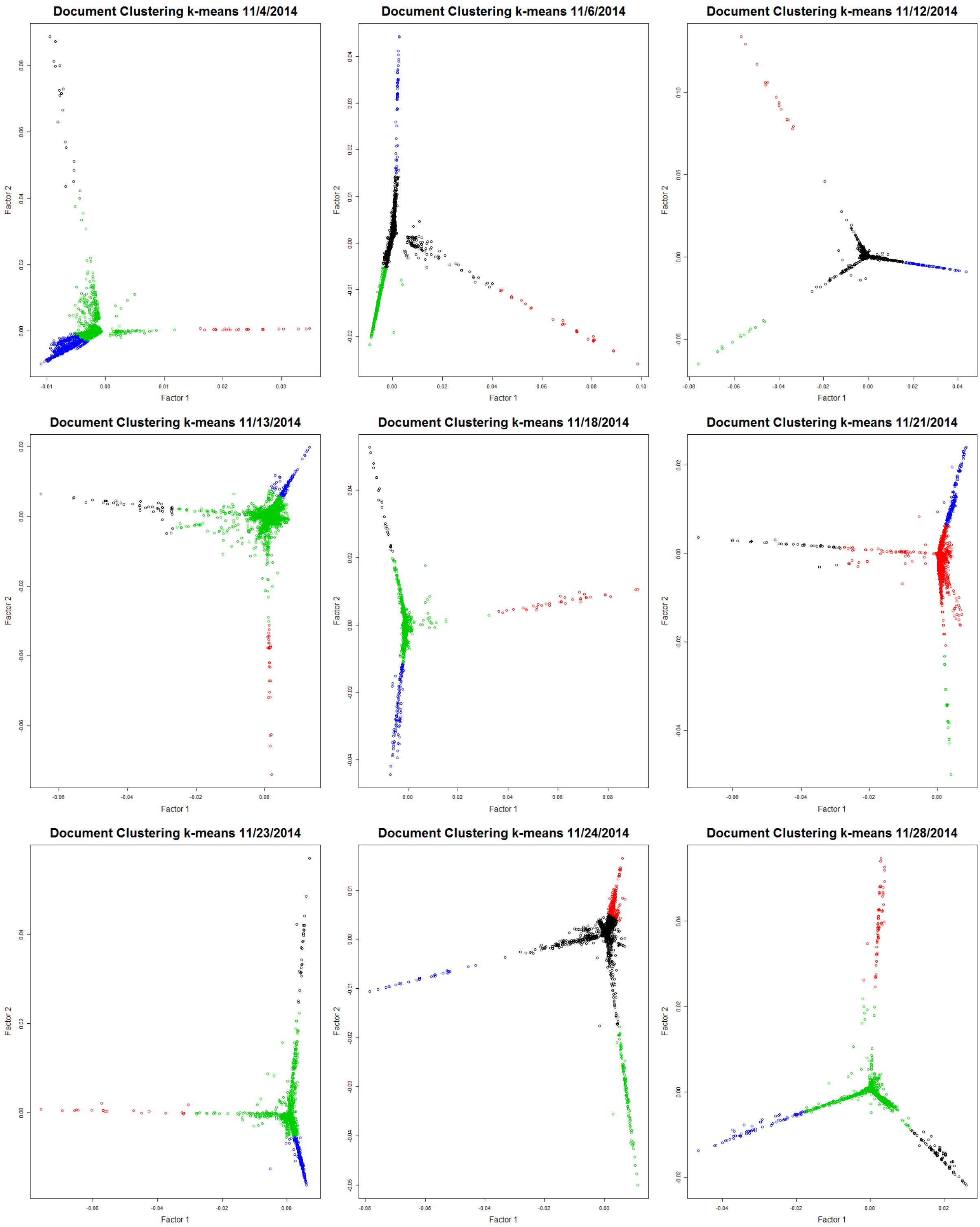


Figure 5.19: Structure of the data in the new two dimensional coordinates for nine exemplary days in November 2014

Example 5.9. *Example 5.8 continued - Clustering with the three main factors of November 16, 2014*

Another possibility is to look at 3D scatterplots, as we capture more of the overall variance by including three instead of two factors in the analysis and also the visualization could lead to a better insight. By including three factors, the captured variance increases from 10 % to 15 % in average.

```
> summary(capt.variance)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.1123  0.1285  0.1450  0.1449  0.1577  0.2011
```

The respective number for each day of November 2014 is given in Table 5.10.

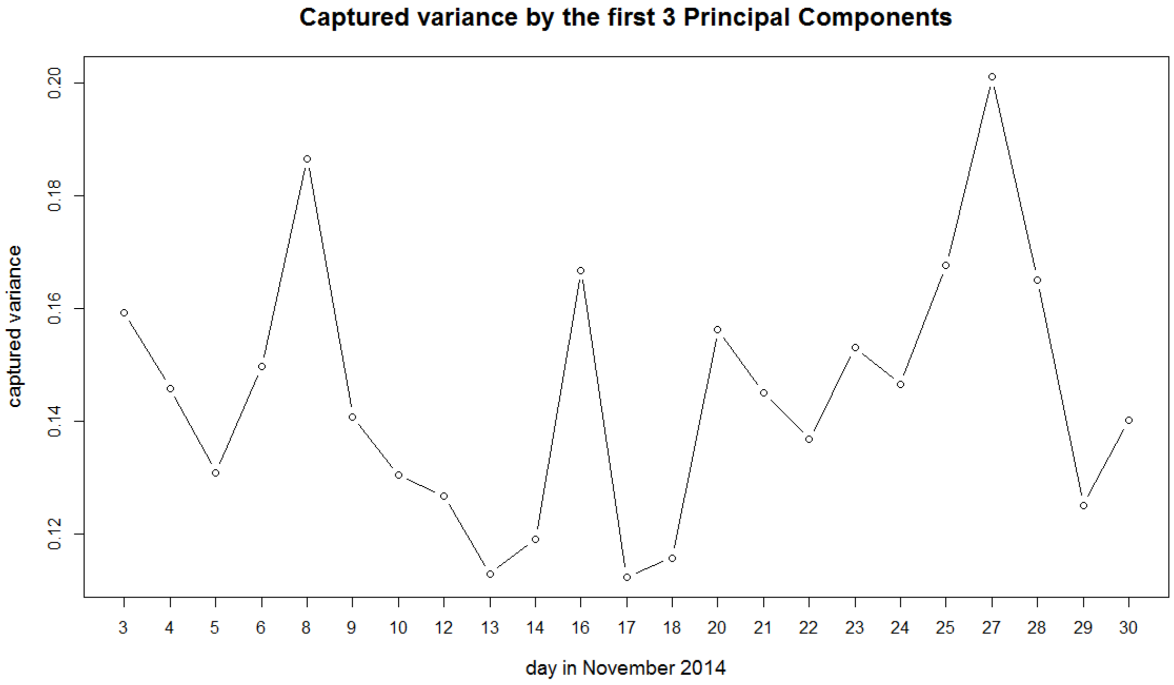


Figure 5.20: Captured variance by the first three Principal Components

Figure 5.21 shows the crude 3D scatterplot without any clustering.

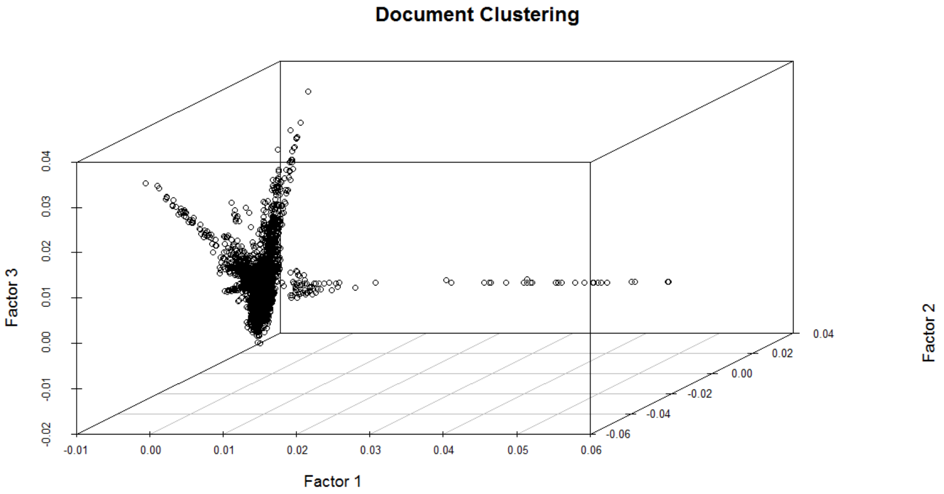


Figure 5.21: 3D plot of Islamic State tweets of November 16, 2014

The k-means algorithm carried out on the first three factors of November 16, 2014 results in a different picture than if it is performed on two factors. Analogously to the two factor k-means clustering we have to decide on the number of clusters upfront. As it turns out five clusters are most suitable, as we thus get the 'UN genocide' cluster as a separate cluster. Giving the five clusters a closer look we can detect the differences and the similarities compared to the two factor case. Table 5.11 shows the five clusters, the most frequent terms contained in them and their respective size. First of all we see that the blue Cluster 1, the green Cluster 2, the red Cluster 3 and the black Cluster 4 are almost the same as in the two factor case, although only 707 instead of 882 tweets were assigned to Cluster 1, 5843 instead of 9568 tweets were assigned to Cluster 3 and 6887 instead of 4176 to Cluster 4. But in contrary to the two factor case we now also detect the cyan coloured Cluster 5 which treats the death confirmation of the White House and the resulting response video of Peter Kassig's family. The clustering with three factors is superior compared to the two factor clustering, as it explains more of the variance. Therefore, by looking at Table 5.9 we can state a certain classification rate by comparing the number of tweets assigned to the clusters.

Cluster 1: $\frac{703}{882} \approx 0.797$ Cluster 2: $\frac{473}{473} = 1$

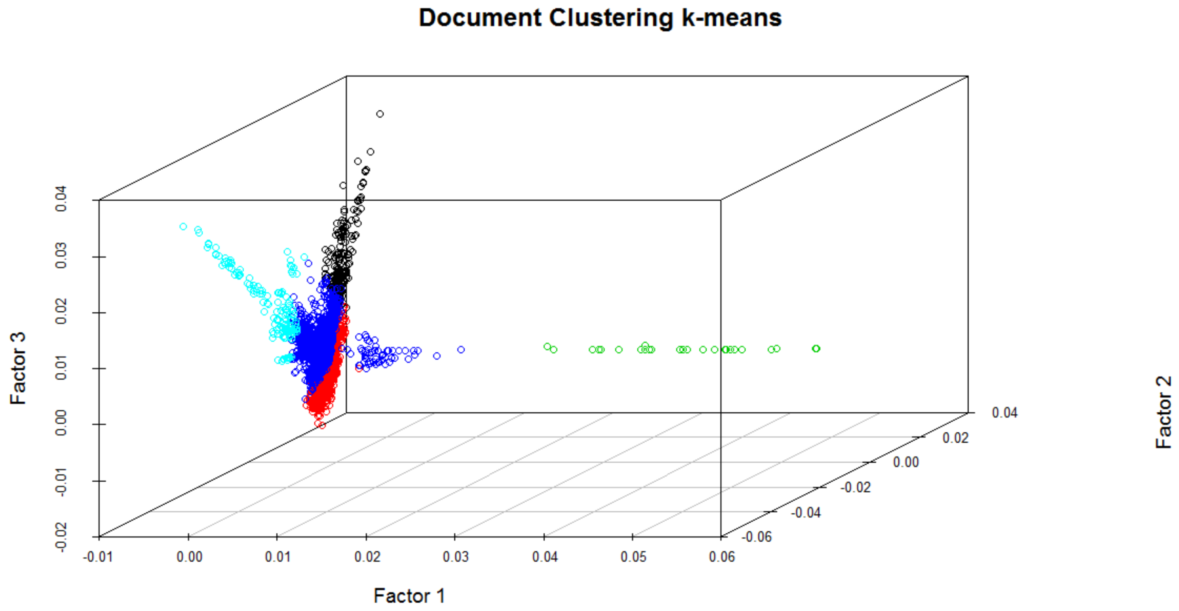
Cluster 3: $\frac{5424}{9568} \approx 0.659$ Cluster 4: $\frac{2752}{4176} \approx 0.567$

The interpretation of these numbers is that the two factor clustering assigned 80% of Cluster 1 correctly to Cluster 1 and almost 57% of Cluster 3 correctly to Cluster 3. Furthermore Cluster 4 assigned 66% of the Cluster 4 tweets correctly to this Cluster. Cluster 2 is even 100% assigned correctly, if we assume that the three factor clustering detects the true clusters.

kmeans 3\kmeans 2	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Sum
Cluster 1	703	0	4	0	707
Cluster 2	0	473	0	0	473
Cluster 3	179	0	5424	246	5843
Cluster 4	0	0	4129	2752	6887
Cluster 5	0	0	11	1178	1189
Sum	882	473	9568	4176	15099

Table 5.9: Comparison of cluster membership between k-means clustering with 2 factors and k-means clustering with 3 factors

The k-medoids algorithm is not very satisfactory as the silhouette technique only detects two clusters which is not sufficient. As Table 5.12 shows, the tweets are only separated into the pure information of the beheading and the reaction of the White House and Peter Kassig's family.



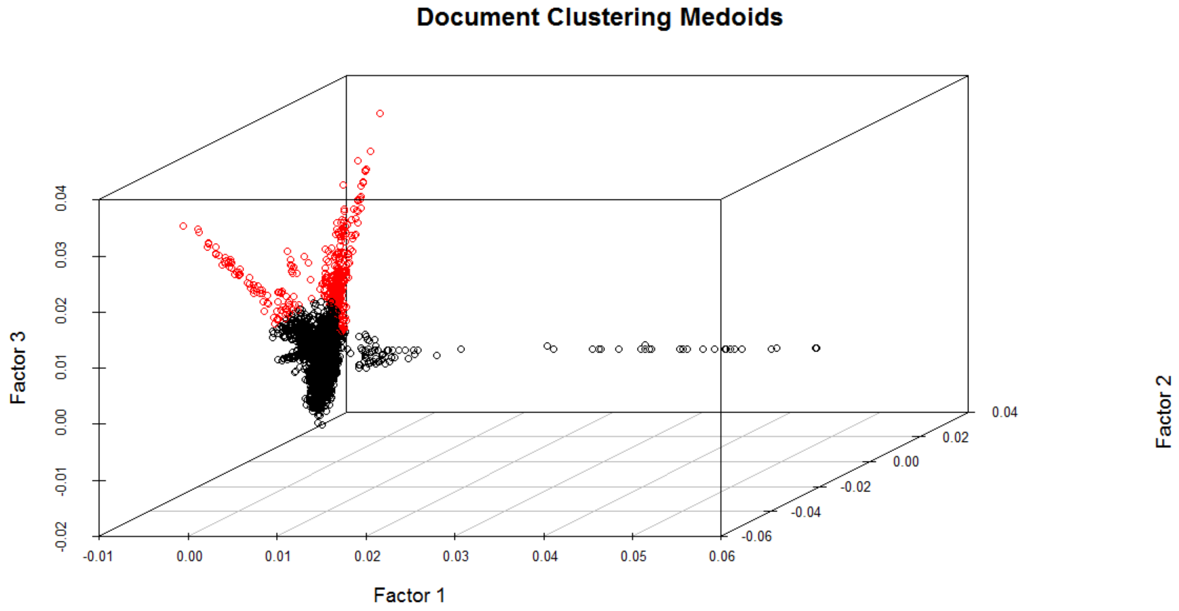


Figure 5.22: Top: Clustering of Islamic State tweets of November 16, 2014 via k-means and three factor
Bottom: Clustering of Islamic State tweets of November 16, 2014 via k-medoids and three factors

date	11/03/14	11/04/14	11/05/14	11/06/14	11/08/14	11/09/14	11/10/14	11/12/14	11/13/14	11/14/14	11/16/14	11/17/14
capt. variance	0.159	0.146	0.131	0.150	0.187	0.141	0.130	0.127	0.113	0.119	0.167	0.112

date	11/18/14	11/20/14	11/21/14	11/22/14	11/23/14	11/24/14	11/25/14	11/27/14	11/28/14	11/29/14	11/30/14
capt. variance	0.116	0.156	0.145	0.137	0.153	0.146	0.168	0.201	0.165	0.125	0.140

Table 5.10: Captured variance by the first three components

	colour	size	TOP 1	TOP 2	TOP 3	TOP 4	TOP 5	TOP 6	TOP 7	TOP 8	TOP 9	TOP 10
Cluster 1	blue	707	obama	condemns	states	killing	kassig	peter	president	cameron	said	kurdish
Cluster 2	green	473	denying	criteria	#demandforaction	victims	un	recognize	genocide	doesnot	dishonor	sweden
Cluster 3	red	5849	kassig	us	peter	obama	video	via	hostage	isis	killing	beheading
Cluster 4	black	6881	us	kassig	video	claims	beheaded	aid	worker	peter	hostage	group
Cluster 5	cyan	1189	video	kassig	beheading	family	responds	white	house	confirms	us	death

Table 5.11: Clusters detected by k-means and three factors

	colour	size	TOP 1	TOP 2	TOP 3	TOP 4	TOP 5	TOP 6	TOP 7	TOP 8	TOP 9	TOP 10
Cluster 1	black	12810	us	kassig	video	claims	peter	beheaded	aid	worker	hostage	group
Cluster 2	red	2289	video	kassig	beheading	white	house	us	peter	family	responds	aid

Table 5.12: Clusters detected by k-medoids and three factors

Example 5.10. *Example 5.9 continued - Clustering with k factors of November 16, 2014*

As a final step we carry out the k-means clustering with k factors, whereas k is the number of components to keep suggested by the Scree test. The captured variance by the first k Principal Components is given in Table 5.13 for each day of November 2014.

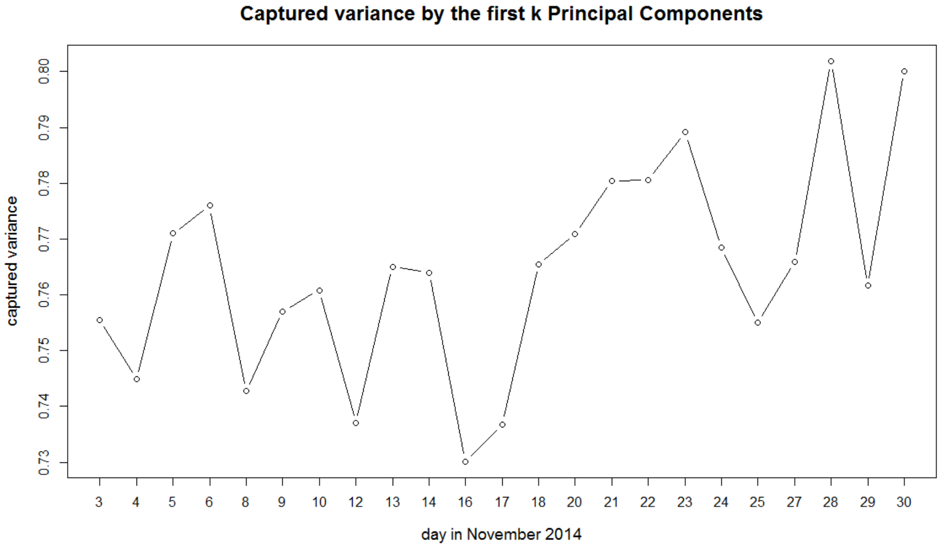


Figure 5.23: Captured variance by the first k Principal Components

```
> summary(capt.variance)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.7301  0.7552  0.7650  0.7644  0.7735  0.8019 
```

By looking at the summary of the captured variance we see that we obtained a very good approximation, with a mean of 76.44 %, although we reduced the amount of variables drastically. On November 16, 2014 k was determined to be 15. Again choosing five clusters makes most sense and we detect the clusters given in Table 5.14. This time the clustering is not as clear as in the precedent clusterings. Although Cluster 2 is identical, the rest is less distinguishable. Cluster 1 is comparable to the three factor Cluster 1 with 704 instead of 707 tweets, whereas Cluster 3, 4 and 5 all treat the video of the beheading. Therefore the three factor clustering gives better and clearer outcomes. This might result from too much noise which was included by taking 15 factors into account. So we have to examine how many factors give the most reasonable clustering as too many factors can add irrelevant information to the meaningful factors and therefore distort the grouping of tweets. This time the k-medoids clustering with 15 factors leads to a better grouping. The resulting eight clusters are very sharp and reasonable, which can be seen in Table 5.15.

date	11/03/14	11/04/14	11/05/14	11/06/14	11/08/14	11/09/14	11/10/14	11/12/14	11/13/14	11/14/14	11/16/14	11/17/14	11/18/14	11/20/14	11/21/14	11/22/14	11/23/14	11/24/14	11/25/14	11/27/14	11/28/14	11/29/14	11/30/14
capt. variance	0.756	0.745	0.771	0.776	0.743	0.757	0.761	0.737	0.765	0.764	0.730	0.737	0.765	0.771	0.780	0.781	0.789	0.768	0.755	0.766	0.802	0.762	0.800

Table 5.13: Captured variance by the first k components

	size	TOP 1	TOP 2	TOP 3	TOP 4	TOP 5	TOP 6	TOP 7	TOP 8	TOP 9	TOP 10
Cluster 1	704	us	confirms	death	review	video	kassig	house	white	aid	worker
Cluster 2	473	denying	criteria	#demandforaction	victims	un	recognize	genocide	doesnot	dishonor	sweden
Cluster 3	5114	kassig	claims	beheaded	hostage	video	us	peter	american	beheading	family
Cluster 4	3336	aid	us	worker	video	kassig	claims	peter	group	beheaded	beheads
Cluster 5	5472	us	video	peter	kassig	via	new	beheading	hostage	isis	aid

Table 5.14: Clusters detected by k-means and k factors

	size	TOP 1	TOP 2	TOP 3	TOP 4	TOP 5	TOP 6	TOP 7	TOP 8	TOP 9	TOP 10
Cluster 1	3903	us	aid	worker	video	claims	kassig	peter	group	beheaded	beheads
Cluster 2	5831	kassig	video	us	peter	via	beheading	hostage	new	american	obama
Cluster 3	532	confirms	us	death	review	video	house	white	aid	worker	peter
Cluster 4	476	family	beheading	video	responds	kassig	peter	restraint	kassigs	call	via
Cluster 5	2915	kassig	beheaded	claims	hostage	us	peter	video	american	reuters	militants
Cluster 6	718	obama	kassig	condemns	killing	peter	state's	president	said	statement	states
Cluster 7	473	denying	criteria	#demandforaction	victims	un	recognize	genocide	doesn't	dishonor	sweden
Cluster 8	251	house	released	statement	white	kassig	beheading	family	responds	video	sunday

Table 5.15: Clusters detected by k-medoids and k factors

To sum up we reduced the number of variables to one third of the original ones and obtained new factors via the Principal Component Analysis. These new factors explain up to 75 % of the original data and we can identify the main clusters among the tweets of every day. Furthermore we can examine the shapening within those clusters in order to partition them even further. Even by considering only the first two or three principal components we obtain very good clustering results, as we can draw conclusions on what the most important topics have been on this day.
The code, which carries out the Principal Component Analysis, computes all the above mentioned figures, produces the plots and determines the clusters, is given in the Appendix B.6.

5.6 Assessment of new data

It is now possible to express a new incoming tweet via the factors and assign it to a cluster. Therefore we could assess the tweets very fast and exact and see developments and changes in the hot topics. As mentioned above, one third of the data already captures 95% of all terms appearing on this day. For this reason we only use the first 35 % of our data to perform the Principal Component Analysis and detect the clusters. We then treat the remaining tweets as new incoming ones and classify them via the calculated new factors and the identified clusters.

Example 5.11. *Clustering of new tweets*

As we see in Figure 5.24, which depicts the first third of the tweets on November 16, 2014 and their cluster according to the new factors, the picture is of course different compared to Figure 5.18 containing all tweets.

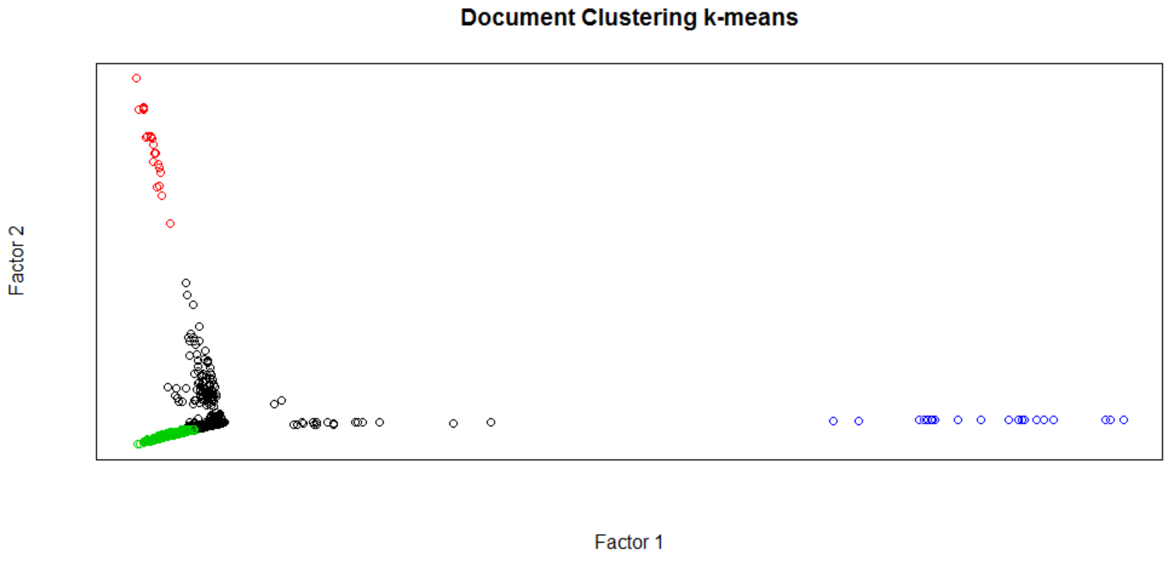


Figure 5.24: Document Clustering of training data with the two main factors of November 16, 2014

But as we can see in Table 5.16 the clusters are similar up to a certain degree and again reasonable.

	colour	TOP 1	TOP 2	TOP 3	TOP 4	TOP 5
Cluster 1	black	us	isil	peter	via	isis
Cluster 2	red	threat	economy	front	jihadi	pages
Cluster 3	green	kassig	us	video	peter	obama
Cluster 4	blue	criteria	denying	#demandforaction	victims	un

	TOP 6	TOP 7	TOP 8	TOP 9	TOP 10
Cluster 1	kassig	fight	video	slaughter	syria
Cluster 2	warning	threats	bring	terror	uk
Cluster 3	beheading	aid	worker	confirms	family
Cluster 4	genocide	dishonor	doesnt	recognize	sweden

Table 5.16: Clusters detected by kmeans and two factors of the training data

As in Section 5.5 the recognition of the actions of the Islamic State as genocide, i.e. the blue Cluster 4, general information of the beheading video, i.e. the black Cluster 1 and the statement of Barack Obama regarding the death of Peter Kassig, i.e. the green Cluster 3 are the main clusters. Note that the second most frequent term in Cluster 1 *isil* is another synonym for Islamic State and the abbreviation for Islamic State of Iraq and the Levant. Now new tweets can be classified. We therefore show the exemplary classification of the three new tweets below:

Tweet 1	RT@DemandForAction: By denying genocide, you dishonor victims. Sweden doesnt recognize UN criteria: #DemandForAction
Tweet 2	”Islamic State video claims American beheaded http://t.co/Yhfe4VbT8f via @USATODAY”
Tweet 3	”The White House has confirmed the death of American aid worker Peter Kassig.

Table 5.17: New incoming tweets

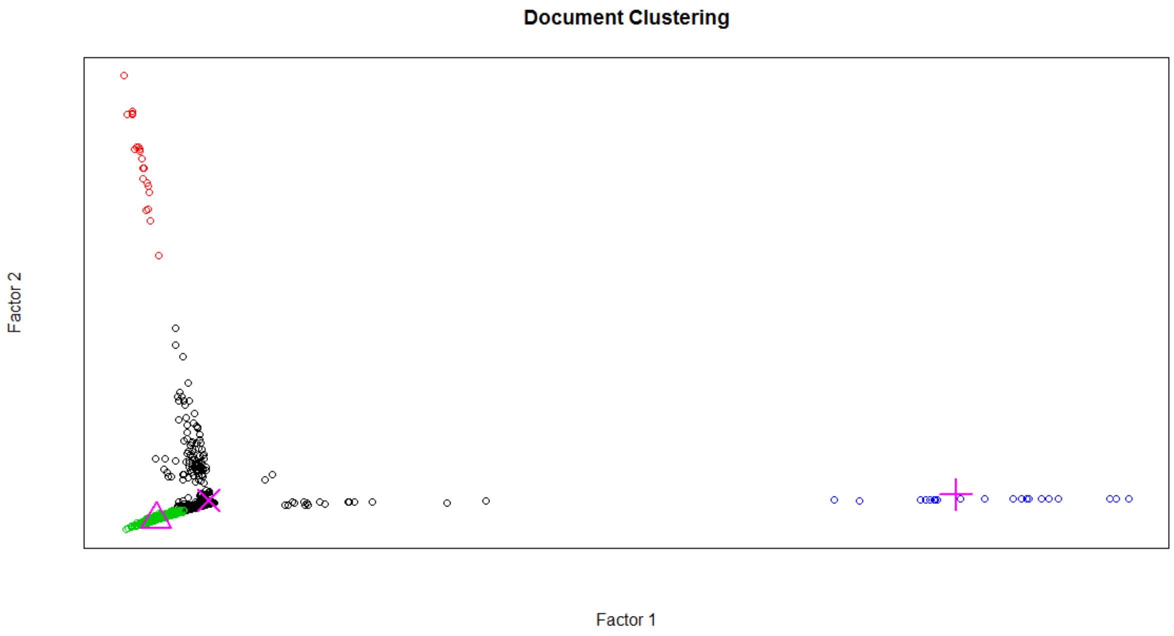


Figure 5.25: Document Clustering of new tweets with the two main factors of November 16, 2014

The cross stands for the first tweet, the X for the second and the third tweet is marked by the triangle. The first tweet is clearly assigned to the blue cluster as it solely describes the debate on the genocide recognition. The second tweet is located in the middle of the black cluster as it gives general information about the beheading. And the third tweet is classified as belonging to the green cluster. This is very interesting as Obama is not mentioned by name but instead the White House confirmed the death. Of course that makes sense as Barack Obama stands for the White House and vice versa.

So the tweets are assigned to their respective cluster automatically and in the same way it would be done by a human mind. The assessment process even captures related tweets, although they do not contain the same terms but have the same meaning.
The code, which performs the clustering with a training and a test data set is given in the Appendix B.7.

Chapter 6

Application 2 - Topic: Ebola

The second example is the outbreak of Ebola in West Africa at the end of 2014. As in these months, especially in certain countries of West Africa, e.g. Guinea, Liberia and Sierra Leone, Ebola got spread out and the deaths rose dramatically, we will briefly examine the relationship between the tweets treating the subject Ebola and the official deaths from the website of the WHO, see [19].

6.1 Examination of certain countries

First of all we examine all tweets containing the keyword *Ebola* in November and December 2014 and January 2015. We look at the relative appearance of the words *Guinea*, *Liberia* and *Sierra Leone*. In other words we get the percentage of in how many tweets of all extracted Ebola tweets these three country names are contained.
The respective code is given in the Appendix B.8.
Looking at Figure 6.1 we see that there is a change over time how often the certain countries are mentioned. We can also see significant differences between two consecutive days. So there are days where certain countries are more in the focus of the public opinion than on other days. So we might assume that on these days the death rates due to Ebola increased in this country. The assumption now is that the more deaths occurred due to Ebola in a certain country on a certain day, the more tweets contain the country name on this day.

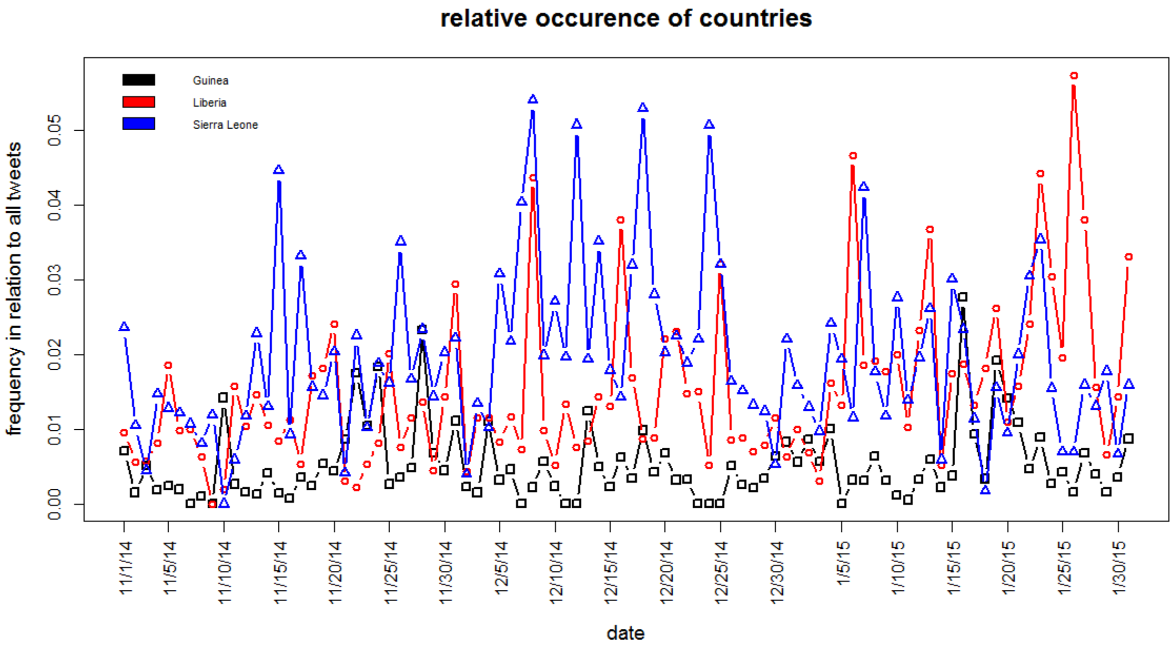


Figure 6.1: Country Appearance on Twitter

6.2 Relationship between tweets and official WHO data

Unfortunately the data on the WHO website is only on a weekly basis. Therefore we take the weekly average of the relative occurrence of the tweets for each country to make the data comparable. Unfortunately this eliminates the observed daily effect.
Table 6.1 shows the weekly average occurrence of tweets of the respective country.

week	1	2	3	4	5	6	7	8	9	10
Guinea [%]	0.274	0.098	0.390	0.271	0.274	0.655	1.053	0.780	0.417	0.420
Liberia [%]	0.953	0.988	0.691	1.261	1.208	1.360	0.867	1.279	1.394	1.598
Sierra Leone [%]	1.065	1.145	0.753	1.791	2.346	1.226	2.058	1.636	2.916	2.701

week	11	12	13	14	15	16
Guinea [%]	0.391	0.39	0.519	0.325	1.267	0.472
Liberia [%]	1.398	1.177	1.640	1.893	1.725	3.275
Sierra Leone [%]	3.076	1.665	1.941	1.752	1.595	1.777

Table 6.1: Weekly average occurrence of tweets of the respective country

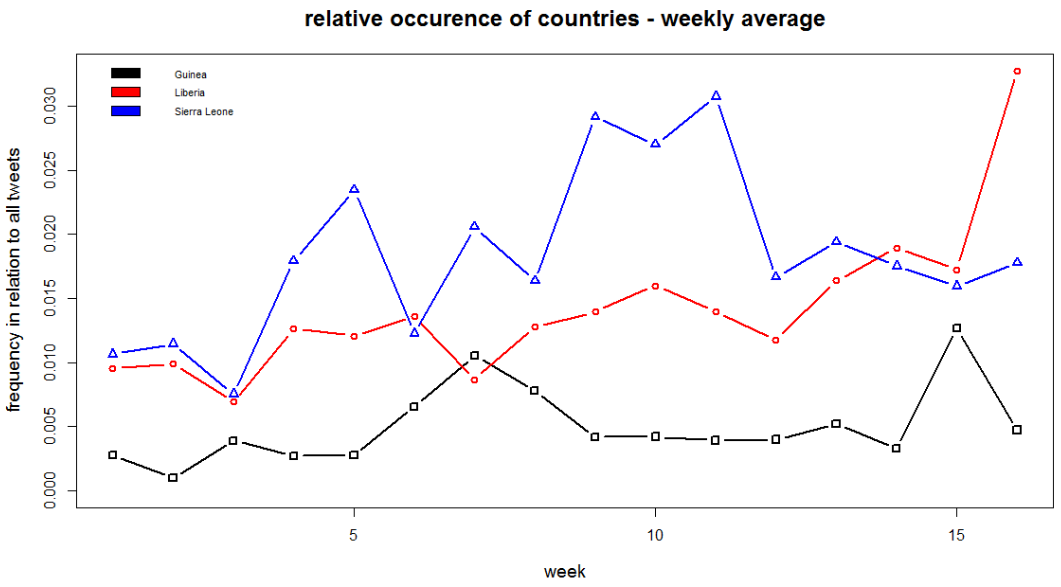


Figure 6.2: Country Appearance on Twitter - weekly average

Now we want to examine the correlation between the Twitter and the WHO data. For this reason we first plot the average weekly data from Twitter against the weekly new deaths according to the WHO data.

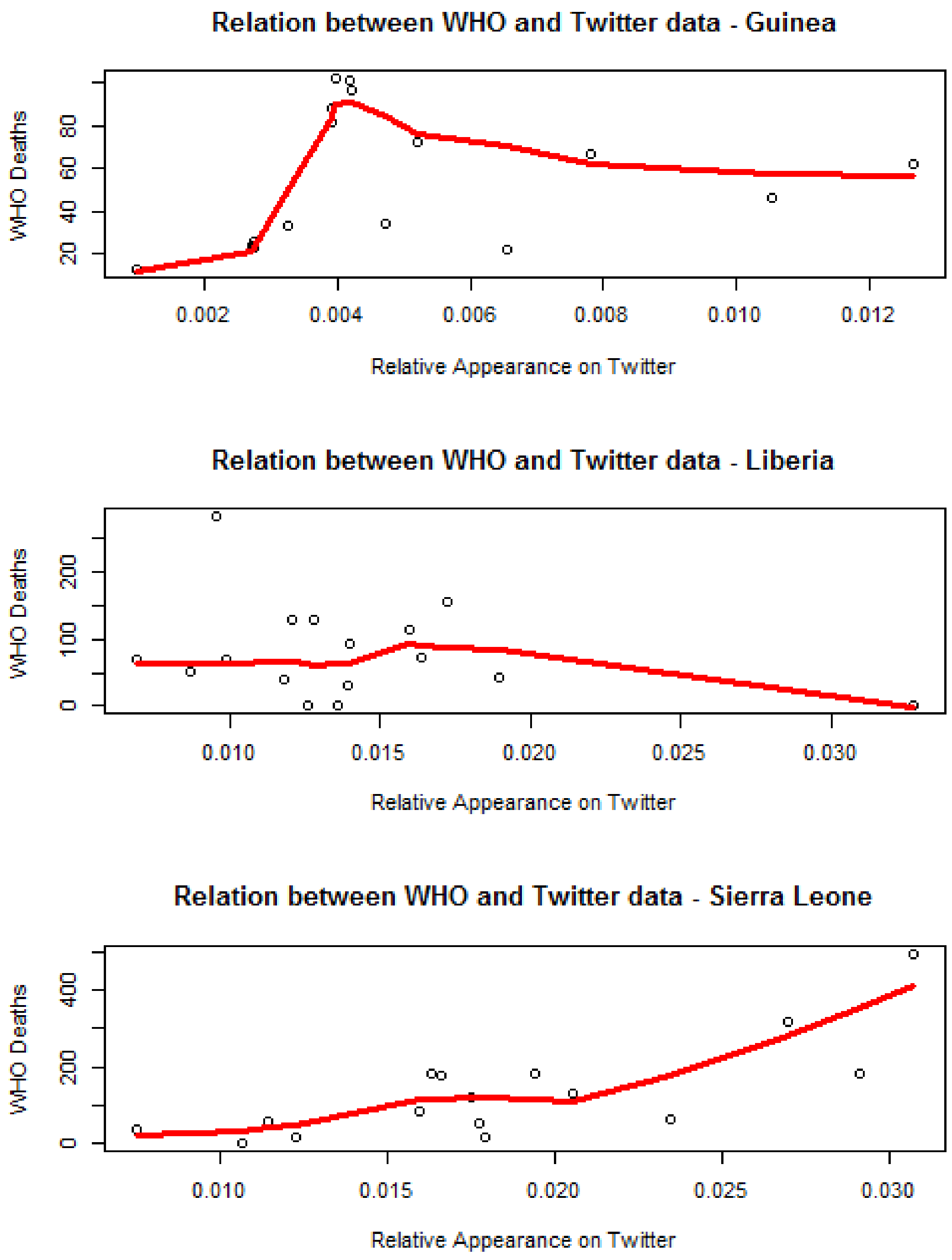
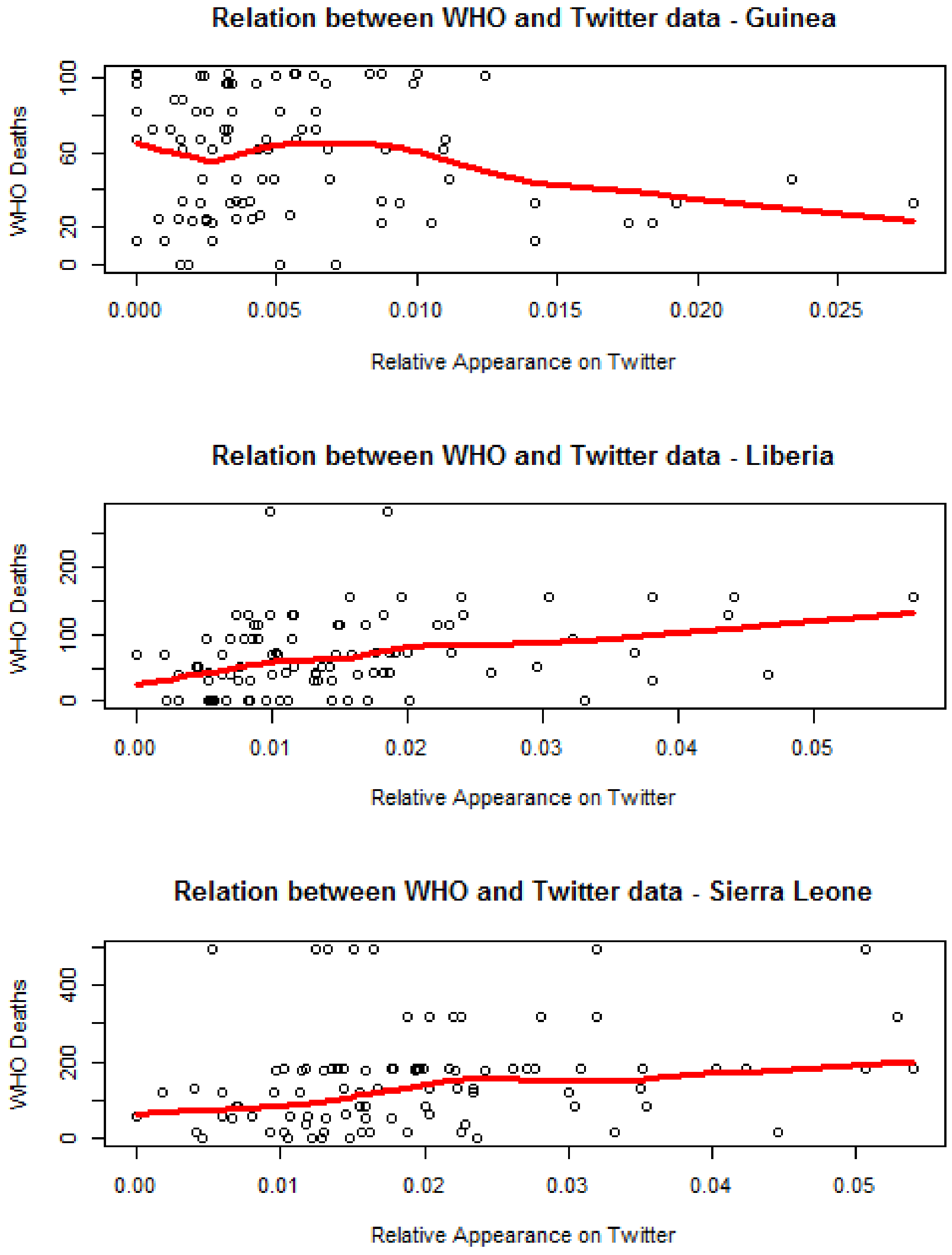


Figure 6.3: Top: Scatterplot of weekly WHO deaths and Twitter data - Guinea
Middle: Scatterplot of weekly WHO deaths and Twitter data - Liberia
Bottom: Scatterplot of weekly WHO deaths and Twitter data - Sierra Leone

As it turns out, only for Sierra Leone the data has some kind of a linear relationship. In order to detect stronger dependencies we plot the data from Twitter against the WHO death data, which is enriched by taking the weekly deaths for every day in this week.



Relation between WHO and Twitter data - Guinea

Relation between WHO and Twitter data - Liberia

Relation between WHO and Twitter data - Sierra Leone

Figure 6.4: Top: Scatterplot of daily WHO deaths and Twitter data - Guinea
Middle: Scatterplot of daily WHO deaths and Twitter data - Liberia
Bottom: Scatterplot of daily WHO deaths and Twitter data - Sierra Leone

This time also the data for Liberia shows a clear positive correlation.
As the figures for new Ebola cases in these three countries are also available on the WHO website, we additionally examine if there is a relation between the new cases and

the Twitter data analogously to the death data. So first we compare the weekly average Twitter data with the new Ebola cases provided by WHO.

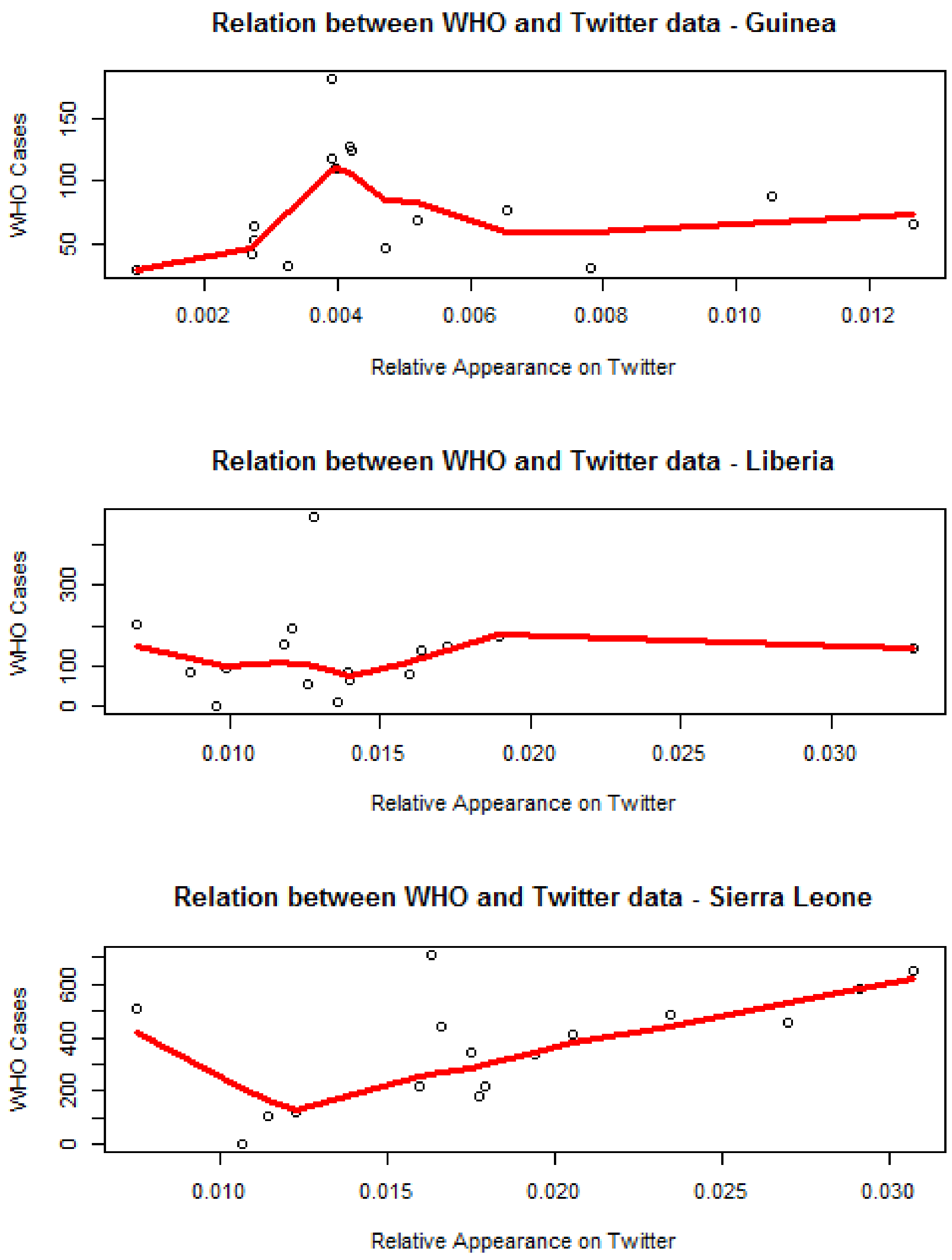


Figure 6.5: Top: Scatterplot of weekly WHO cases and Twitter data - Guinea
Middle: Scatterplot of weekly WHO cases and Twitter data - Liberia
Bottom: Scatterplot of weekly WHO cases and Twitter data - Sierra Leone

Again only Sierra Leone shows some correlation pattern. For this reason we enrich the data by taking the weekly cases for every day in this week.

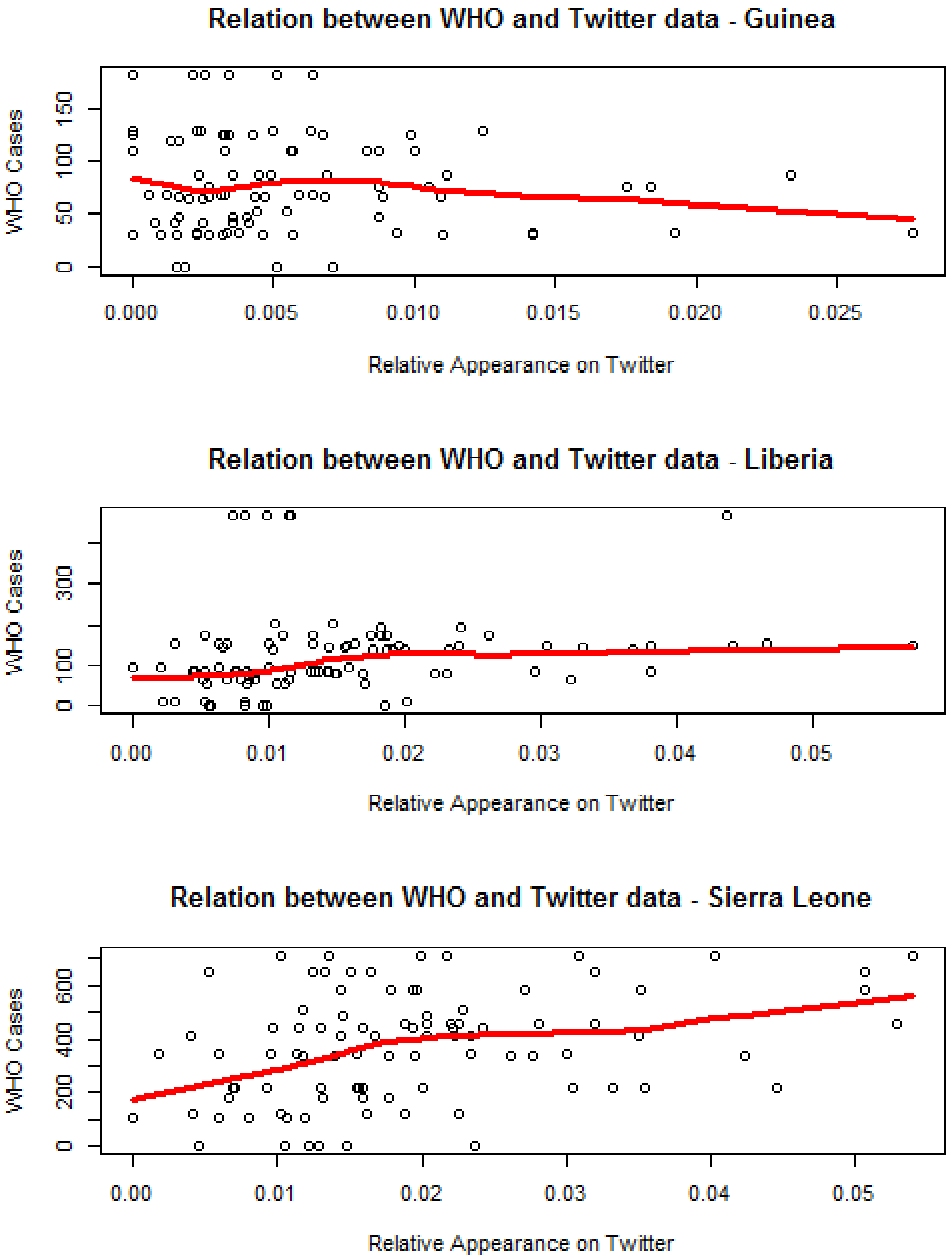


Figure 6.6: Top: Scatterplot of daily WHO cases and Twitter data - Guinea
Middle: Scatterplot of daily WHO cases and Twitter data - Liberia
Bottom: Scatterplot of daily WHO cases and Twitter data - Sierra Leone

We see a clear positive relationship between the relative occurrence of the countries Liberia and Sierra Leone in the Ebola tweets and the official amount of new Ebola cases in these countries. It seems that the more the respective country is mentioned in tweets treating Ebola, the more people have died or were infected by the disease on this day. Therefore we could use the relative occurrence of these countries as a first indicator of how many people died or were infected on this day.

As a last step a linear regression model, according to Definition 4.11 is applied to the response variable deaths in a certain country on a certain day, respectively the new cases of Ebola in a certain country on a certain day and the relative occurrence of this country in all Ebola tweets on this day as the covariate, is set up. By looking at the summaries of the linear regression models which take the deaths as a response variable given in the Appendix B.8, we see that the Twitter covariate is statistically significant for Liberia and Sierra Leone at a 0.01 respectively 0.05 significance level, although the R-squared, as defined in [Fahrmeir (1996)] p. 108, is very low for both models, indicating that further investigations should be carried out. If the new cases of Ebola are taken as the response variable, the covariate is only statistically significant for Sierra Leone but at a 0.001 significance level and with a larger adjusted R-squared than in the precedent models. This brief approach using linear regression was only thought of as a starting point for future work, as more would have gone beyond the scope of this thesis.

Chapter 7

Conclusion and Outlook

This thesis aimed for the investigation of different ways of analyzing large amounts of data extracted from the social media platform Twitter. We therefore examined all tweets in November 2014 containing the key term Islamic State and carried out several basic approaches in order to get a first insight in the data and its structure. We then set the focus on the Principal Component Analysis, as it is thus possible to construct new factors which are combinations of the original variables, reduce their number, reveal former hidden patterns and solely contain useful information. Subsequently we expressed the data with the help of the new variables and clustered the resulting data points via two different clustering methods. As it turned out it is crucial to decide on the number of components to keep, as we could have lost important information by excluding too many factors or on the other hand include too much noise by considering too many factors. As it was shown taking only the first two, respectively three components is sufficient, as they already explain the data and are also better to visualize. Afterwards the different clusters were examined, verified and also compared to each other in order to evaluate the results. In conclusion the developed algorithm detected meaningful and logical clusters which reflected the public opinion and the concerns and attitude of the Twitter users. As a final step an automatic procedure was set up, which used the first couple of hours of a day to carry out the Principal Component Analysis and detect the clusters and then assigned new incoming tweets automatically to the already existing clusters. By doing this a functioning end-to-end process for clustering Twitter data on a particular day concerning an arbitrary issue was established. Lastly a second example was introduced briefly by examining the dependency between the appearance of the keyword Ebola in combination with certain country names on Twitter and the new cases respectively deaths due to this disease according to official WHO data. A first result was a statistically significant relationship between those two variables.

Several starting points for further research appeared along this thesis. The problem of the abortion error while fetching the tweets could be addressed as well as different and more advanced text mining methods could be examined regarding the treatment of the extracted tweets. Furthermore the focus could be set on other methods than the Principal Component Analysis and the choice of the number of clusters could be further investigated. Especially a deeper statistical investigation of the second application is necessary to get a clearer picture of the dependencies.

Appendix A

Tables

A.1 List elements of the output of the ‘searchTwitter‘ function

name of element	content	type	example tweet
text	actual tweet	character	RT @SAP_UA: #SAP Big Data...
id	ID of tweet	character	535358012748099585
screenName	name of user	character	NoSQLDigest
created	date and time	date format	2014-11-20 09:04:31
longitude	longitude	character	NA
latitude	latitude	character	NA
isRetweet	is this retweet	TRUE/FALSE	TRUE
retweeted	tweet been retweeted	TRUE/FALSE	FALSE
retweetCount	number of retweets	numeric	1
favorited	marked as favorite	TRUE/FALSE	FALSE
favoriteCount	number of favorited	numeric	0
replyToSID	ID of reply tweet	character	NA
replyToSN	to whom reply tweet	character	NA
replytoUID	user ID of reply tweet	character	NA
statusSource	user agent	character	...www.simbasystems.com...
truncated	is the tweet truncated	TRUE/FALSE	FALSE

Table A.1: List elements of the ‘searchTwitter‘ function

A.2 English stopwords

```
> # loading necessary library
> library(tm)
> # list of english stopwords
> stopwords("english")
```

```
  [1] "i"          "me"         "my"         "myself"     "we"
  [6] "our"        "ours"       "ourselves"  "you"        "your"
 [11] "yours"      "yourself"   "yourselves" "he"         "him"
 [16] "his"        "himself"    "she"        "her"        "hers"
 [21] "herself"    "it"         "its"        "itself"     "they"
 [26] "them"       "their"      "theirs"     "themselves" "what"
 [31] "which"      "who"        "whom"       "this"       "that"
 [36] "these"      "those"      "am"         "is"         "are"
 [41] "was"        "were"       "be"         "been"       "being"
 [46] "have"       "has"        "had"        "having"     "do"
 [51] "does"       "did"        "doing"      "would"      "should"
 [56] "could"      "ought"      "i`m"        "you`re"     "he`s"
 [61] "she`s"      "it`s"       "we`re"      "they`re"    "i`ve"
 [66] "you`ve"     "we`ve"      "they`ve"    "i`d"        "you`d"
 [71] "he`d"       "she`d"      "we`d"       "they`d"     "i`ll"
 [76] "you`ll"     "he`ll"      "she`ll"     "we`ll"      "they`ll"
 [81] "isn`t"      "aren`t"     "wasn`t"     "weren`t"    "hasn`t"
 [86] "haven`t"    "hadn`t"     "doesn`t"    "don`t"      "didn`t"
 [91] "won`t"      "wouldn`t"   "shan`t"     "shouldn`t"  "can`t"
 [96] "cannot"     "couldn`t"   "mustn`t"    "let`s"      "that`s"
[101] "who`s"      "what`s"     "here`s"     "there`s"    "when`s"
[106] "where`s"    "why`s"      "how`s"      "a"          "an"
[111] "the"        "and"        "but"        "if"         "or"
[116] "because"    "as"         "until"      "while"      "of"
[121] "at"         "by"         "for"        "with"       "about"
[126] "against"    "between"    "into"       "through"    "during"
[131] "before"     "after"      "above"      "below"      "to"
[136] "from"       "up"         "down"       "in"         "out"
[141] "on"         "off"        "over"       "under"      "again"
[146] "further"    "then"       "once"       "here"       "there"
[151] "when"       "where"      "why"        "how"        "all"
[156] "any"        "both"       "each"       "few"        "more"
[161] "most"       "other"      "some"       "such"       "no"
[166] "nor"        "not"        "only"       "own"        "same"
[171] "so"         "than"       "too"        "very"
```

A.3 Outlier terms of Islamic State tweets of November 2014

date	outlier terms	online reference	publishing time	matching terms

11/03/2014	iraqi(1319)
	group(935)
	kills(754)
	officials(612)
	iraq(596)
	syria(496)
	tribe(463)
	women(462)
	say(460)
	publicly(441)
		[nov3]	11:37	10

11/04/2014	group(800)
	kurdish(638)
	syrian(553)
	militants(478)
	hostages(463)
	rights(457)
	tortured(440)
	children(411)
	says(383)
	kurds(365)
		[nov4]	10:59	9 - kurds

11/05/2014	new(571)
	us(558)
	iraq(543)
	obama(482)
	fight(431)
	will(382)
	zealand(336)
	reuters(334)
	iraqi(305)
	britain(304)
		[nov5]	23:19	10

11/06/2014	obama(2790)
	wrote(2027)
	letter(1928)
	fighting(1780)
	khamenei(1597)
	secret(1459)
	irans(1194)
	secretly(707)
	leader(688)
	supreme(665)
		[nov6]	19:22	10

11/08/2014	kassig(1963)
	video(1903)
	us(1588)
	peter(1118)
	claims(983)
	beheaded(897)
	aid(876)
	worker(857)
	beheading(749)
	released(612)
		[nov8]	9:38	10

date	outlier terms	online reference	publishing time	matching terms

11/09/2014	leader(3320)
	iraqi(2569)
	officials(2559)
	say(2218)
	wounded(1828)
	albaghdadi(1714)
	airstrike(1421)
	group(1410)
	us(1289)
	airstrikes(1143)
		[nov9]	18:03	10

11/10/2014	leader(768)
	us(546)
	group(515)
	iraqi(459)
	aide(419)
	albaghdadi(397)
	abu(389)
	bakr(384)
	iraq(383)
	reuters(377)
		[nov10]	4:07 (11.11.2014)	10

11/12/2014	kurds(257)
	kobani(226)
	supply(163)
	syrias(154)
	block(149)
	route(149)
	two(147)
	suspected(119)
	via(117)
	#isis(114)
		[nov12]	8:14	7 - two, suspected, via

11/13/2014	leader(2516)
	says(1941)
	fight(1264)
	audio(1190)
	isis(1002)
	will(959)
	releases(860)
	abu(843)
	message(765)
	group(700)
		[nov13]	21:32	10

11/14/2014	group(899)
	isis(689)
	iraq(670)
	syria(632)
	un(447)
	war(403)
	crimes(391)
	says(358)
	oil(350)
	commanders(328)
		[nov14]	18:03	9 - says

date	outlier terms	online reference	publishing time	matching terms

11/16/2014	iraqi(1319)
	group(935)
	kills(754)
	officials(612)
	iraq(596)
	syria(496)
	tribe(463)
	women(462)
	say(460)
	publicly(441)
		[nov16]	16:54	10

11/17/2014	video(777)
	frenchman(691)
	peter(395)
	states(382)
	kassig(379)
	father(362)
	reuters(303)
	believed(255)
	son(249)
	jihadi(237)
		[nov17]	22:52	8 - states, father

11/18/2014	frenchman(595)
	video(526)
	kassig(476)
	peter(439)
	beheadings(369)
	seen(349)
	states(333)
	via(312)
	obama(243)
	believed(230)
		[nov18]	10:31(17.11.2014)	8 - states, via

11/20/2014	suicide(303)
	video(291)
	killed(195)
	jihadist(183)
	group(177)
	bombing(174)
	iraqs(173)
	arbil(172)
	claims(172)
	senior(172)
		[nov20]	6:45(21.11.2014)	8 - video, senior

11/21/2014	us(840)
	reuters(697)
	attacks(665)
	allies(639)
	air(599)
	capital(590)
	provincial(589)
	iraq(587)
	strikes(491)
	militants(478)
		[nov21]	20:41	10

date	outlier terms	online reference	publishing time	matching terms

11/22/2014	reuters(761)
	ramadi(736)
	iraqi(683)
	officials(644)
	us(630)
	allies(563)
	militants(536)
	air(509)
	attacks(431)
	tribesmen(424)
		[nov22]	18:08	9 - allies

11/23/2014	iraqi(1061)
	group(809)
	children(691)
	recruits(598)
	exploits(594)
	fight(518)
	forces(497)
	ramadi(434)
	premier(433)
	baghdad(415)
		[nov23]	5:15	4 - forces, ramadi, premier, baghdad

11/24/2014	saudi(528)
	syria(396)
	arabia(390)
	us(358)
	attack(317)
	troops(315)
	shiites(279)
	iraqi(277)
	children(249)
	group(245)
		[nov24]	22:54	10

11/25/2014	syrian(585)
	strikes(510)
	kill(464)
	air(428)
	iraqi(288)
	forces(278)
	fighters(266)
	activists(250)
	two(235)
	battle(233)
		[nov25]	22:14	8 - two, battle

11/27/2014	pope(328)
	violence(324)
	condemns(320)
	iraq(320)
	residents(288)
	cuts(286)
	isis(280)
	mosul(269)
	tribes(258)
	syria(235)
		[nov27]	19:37	9 - tribes

date	outlier terms	online reference	publishing time	matching terms

11/28/2014	us(1305)
	pope(588)
	hit(357)
	allies(351)
	targets(334)
	fifteen(333)
	since(333)
	syria(332)
	wednesday(332)
	mideast(318)
		[nov28]	15:45	8 - pope, mideast

11/29/2014	kobani(695)
	turkey(682)
	group(495)
	isis(430)
	syrian(429)
	town(422)
	iraq(391)
	border(332)
	suicide(330)
	attacking(301)
		[nov29]	15:52	10

11/30/2014	group(786)
	usled(478)
	strikes(411)
	hit(365)
	raqqa(355)
	us(352)
	reuters(341)
	syrias(324)
	monitoring(298)
	will(295)
		[nov30]	15:44	9 - will

Table A.2: Outlier terms of November 2014

A.4 High loadings of the first factor of each day of November 2014

```
11/03/2014
[1] "airstrikes" "degrade"    "destroy"    "fail"       "ibd"
[6] "obamas"     "puny"

11/04/2014
[1] "airstrikes" "degrade"    "destroy"    "fail"       "ibd"
[6] "obamas"     "puny"

11/05/2014
[1] "beef"     "citizens" "joining"  "monitor"  "new"      "prevent"
[7] "steps"    "stop"     "vows"     "zealand"

11/06/2014
[1] "aged"      "christian" "girls"     "list"      "price"     "shows"
[7] "slave"     "sold"      "yazidi"

11/08/2014
[1] "_isis"       "ally"        "army"        "cooperation" "former"
[6] "member"      "reveals"     "sees"        "turkey"      "turkish"

11/09/2014
[1] "iraqi"     "leader"    "officials" "recover"   "say"       "wounded"

11/10/2014
[1] "atlas"        "executes"     "geller"       "heres"        "journalists"
[6] "mosul"        "pamela"       "shrugs"

11/12/2014
 [1] "activists" "atlas"     "beheads"   "block"     "human"     "kobani"
 [7] "kurds"     "libya"     "photo"     "rights"    "route"     "shrugs"
[13] "supply"    "syrias"

11/13/2014
[1] "_volcanoes" "chief"      "contain"    "emerge"     "jihad_"
[6] "online"     "posted"     "said"       "urging"     "voic"

11/14/2014
 [1] "al"          "central"     "command"     "forces"      "hit"
 [6] "iraqi"       "oil"         "qaedalinked" "strikes"     "town"
[11] "usled"
```

```
11/16/2014
[1] "#demandforaction" "@kinonuri"        "criteria"         "denying"
[5] "dishonor"         "doesn_t"          "genocide"         "recognize"
[9] "sweden"           "un"               "victims"

11/17/2014
[1] "father"  "medical" "monday"  "says"    "son"     "student" "thinks"
[8] "uk"      "video"

11/18/2014
[1] "barbaric" "families" "lures"    "madness"  "shock"    "sons"

11/20/2014
[1] "arbil"    "bombing"  "car"      "claimed"  "claims"   "group"
[7] "iraqs"    "jihadist" "rare"     "suicide"  "today"

11/21/2014
[1] "allegedly" "court"     "dutch"     "mum"       "rescues"   "sy"
[7] "teen"      "teenager"  "travelled"

11/22/2014
[1] "across"    "affinity"  "among"     "black"     "militants" "officia"
[7] "operating" "pakistan"  "signs"     "standard"

11/23/2014
[1] "based"  "face"   "french" "jets"   "join"   "jordan" "set"    "soon"
[9] "stat"

11/24/2014
[1] "found"       "guilty"      "hagel"       "house"       "jv"
[6] "nyt"         "portraying"  "resignation" "team"        "white"

11/25/2014
[1] "air"      "baghdad"  "baiji"    "battle"   "fighters" "forces"
[7] "iraqi"    "kill"     "near"     "refinery" "strikes"  "syrian"

11/27/2014
[1] "control"   "experts"   "funds"     "iraqi"     "keep"      "lacks"
[7] "ma"        "militants" "syrian"    "territory"

11/28/2014
 [1] "alleged"    "allies"     "arif"       "fifteen"    "fighter"
 [6] "hit"        "left"       "majeed"     "months"     "mumbai"
[11] "questioned" "since"      "six"        "targets"    "us"
```

```
[16] "wednesday"

11/29/2014
[1] "airstrikes" "areas"      "baghdad"    "held"       "iraq"
[6] "kill"       "seventeen"

11/30/2014
[1] "agenda"   "breaks"   "congress" "crams"    "days"     "ebola"    "final"
[8] "measures" "tax"
```

Appendix B

Codes

B.1 Authentication process of Twitter

```
> # loading necessary library
> library(twitteR)
> # authentication process
> twitCred <- OAuthFactory$new(consumerKey = "P1***RvR" ,
+ consumerSecret = "EG***5DO" ,
+ requestURL = "https://api.twitter.com/oauth/request_token" ,
+ accessURL = "https://api.twitter.com/oauth/access_token" ,
+ authURL = "http://api.twitter.com/oauth/authorize")
> # download certificate
> download.file(url= "http://curl.haxx.se/ca/cacert.pem" ,
+               destfile = "cacert.pem")
> twitCred$handshake(cainfo = "cacert.pem")
> registerTwitterOAuth(twitCred)
[1] TRUE
> # save it for a future sessions
> save(list="twitCred", file="twitteR_credentials")
```

B.2 General Time Patterns of Islamic State Tweets in November 2014

```
> # loading necessary libraries
> library(twitteR)
> library(graphics)
> # load verification standard of twitter
> load("twitteR_credentials")
> registerTwitterOAuth(twitCred)
> # reading in all twitter extractions from november containing the keyword
> # "islamic_state"
> days <- 30
> tweets.november <- list()
> tweets.minute <- list()
> n <- vector()
> m <- vector()
> ind.incomplete <- vector()
> label.day <- vector()
> for (i in 1:days){
+   # importing routine
+   if(i < 10){
+     number <- paste("0" , i , sep = "")
+   }else{
+     number <- i
+   }
+   name <- paste("~/Uni/Master/Master Thesis/Coding/twitter_data/islamic_state/
+                 01.november_2014/isdf_2014-11-" , number , sep = "")
+   name <- paste(name , ".RData" , sep = "")
+   load(name)
+   data.frame <- isdf
+   m[i] <- length(isdf$text)
+   remove(isdf)
+   # list containing the daily tweets
+   tweets.november[[i]] <- data.frame
+   tweets.minute[[i]] <- data.frame
+   # list containing the corresponding times (aggregated to minutes)
+   timestamp <- strptime(data.frame$created , "%Y-%m-%d %H:%M:%S")
+   tweets.minute[[i]]$created <- as.factor(format(timestamp , '%H:%M'))
+   n[i] <- length(levels(tweets.minute[[i]]$created))
+ }
```

```
> # clean up
> remove(data.frame)
> remove(name)
> # plot the number of extracted tweets per day in a barplot
> seq <- c(1 , 5 , 10 , 15 , 20 , 25 , 30)
> mp <- barplot(m , xaxt = "n" , main = "extracted tweets per day" ,
+               ylab = "number of tweets per day" , xlab = "day in November 2014" ,
+               ylim = c(0 , max(m) + 5000) , cex.lab = 1.2 , cex.main = 1.5)
> text(x = mp , y = m + 1000 , labels = m , cex = 0.6)
> axis(side = 1 , at = mp[seq] , labels = seq)
> # determine the most complete day
> max <- which.max(n)
> # create basic plot
> seq <- seq(n[max] , 1 , -60)
> smoothx <- list()
> smoothy <- list()
> max.smooth <- vector(mode = "numeric" , length = days)
> ind <- vector(mode = "numeric" , length = days)
> for (i in 1:days){
+   smoothx[[i]] <- smooth.spline((n[max]-n[i]+1):n[max] ,
+                   summary(tweets.minute[[i]]$created , maxsum = n[i]) ,
+                                                        df = 10)$x
+   smoothy[[i]] <- smooth.spline((n[max]-n[i]+1):n[max] ,
+                   summary(tweets.minute[[i]]$created , maxsum = n[i]) ,
+                                                        df = 10)$y
+   ind[i] <- which.max(smoothy[[i]])
+   max.smooth[i] <- max(smoothy[[i]])
+ }
> maxi.smooth <- max(max.smooth)
> plot(1:n[max] , summary(tweets.minute[[max]]$created , maxsum = n[max]) ,
+      type = "n" , ylab = "number of tweets per minute" ,
+      xlab = "time of day in UTC" , cex.lab = 1.2 , cex.main = 1.5 ,
+      xaxt = "n" , main = "tweet frequencies" , ylim = c(0 , maxi.smooth + 5))
> axis(side = 1 , at = seq , labels = levels(tweets.minute[[max]]$created)[seq])
> # add smoothed curve of tweet course over the day for each day in a different
> # colour
> for (i in 1:days){
+   lines(smooth.spline((n[max]-n[i]+1):n[max] ,
+                        summary(tweets.minute[[i]]$created , maxsum = n[i]) ,
+                                df = 10) , col = i , lwd = 1)
+   if(i < 10){
+     label.day <- paste("0" , i , sep = "")
+     label.day <- paste(label.day , ".11.2014" , sep = "")
+   }
+   label.day <- paste(i , ".11.2014" , sep = "")
+   text(x = smoothx[[i]][ind[i]] , y = smoothy[[i]][ind[i]] + 1 ,
+        labels = label.day , col = i , cex = 0.6)
+ }
> # remove days where the cancelation error was after 12 am
> # (so less than 12 hours coverage)
> n.complete <- n
> for (i in 1:days){
+   if (n[days - i + 1] < (12*60)){
+     print(days - i + 1)
+     print(n[days - i + 1])
+     ind.incomplete[i] <- (days - i + 1)
+     n.complete <- n.complete[-(days - i + 1)]
+     tweets.minute[[days - i + 1]] <- NULL
+   }
+ }
> ind.incomplete <- ind.incomplete[!is.na(ind.incomplete)]
> days.complete <- length(tweets.minute)
> max.complete <- which.max(n.complete)
> max.smooth <- vector(mode = "numeric" , length = days.complete)
> ind <- vector(mode = "numeric" , length = days.complete)
> for (i in 1:days.complete){
+   smoothx[[i]] <- smooth.spline((n.complete[max.complete]-n.complete[i]+1):
+   n.complete[max.complete] , summary(tweets.minute[[i]]$created ,
+                                     maxsum = n.complete[i]) , df = 10)$x
+   smoothy[[i]] <- smooth.spline((n.complete[max.complete]-n.complete[i]+1):
+   n.complete[max.complete] , summary(tweets.minute[[i]]$created ,
+                                     maxsum = n.complete[i]) , df = 10)$y
+   ind[i] <- which.max(smoothy[[i]])
+   max.smooth[i] <- max(smoothy[[i]])
+ }
> maxi.smooth <- max(max.smooth)
> plot(1:n.complete[max.complete] , summary(tweets.minute[[max.complete]]$created ,
+      maxsum = n.complete[max.complete]) , type = "n" , cex.lab = 1.2 , cex.main
+      = 1.5 , xlab = "time of day in UTC" , ylab = "number of tweets per minute" ,
+      xaxt = "n" , main = "tweet frequencies" , ylim = c(0 , maxi.smooth + 5))
```

```
> axis(side = 1 , at = seq , labels = levels(tweets.minute[[max.complete]]
+     $created)[seq])
> # add smoothed curve of tweet course over the day for each day in a different
> # colour
> for (i in 1:days){
+   if(i < 10){
+     label.day[i] <- paste("0" , i , sep = "")
+     label.day[i] <- paste(label.day[i] , ".11.2014" , sep = "")
+   }
+   label.day[i] <- paste(i , ".11.2014" , sep = "")
+ }
> label.day <- label.day[-ind.incomplete]
> for (i in 1:days.complete){
+   lines(smooth.spline((n.complete[max.complete]-n.complete[i]+1):
+         n.complete[max.complete] , summary(tweets.minute[[i]]$created ,
+         maxsum = n.complete[i]) , df = 10) , col = i , lwd = 1)
+   text(x = smoothx[[i]][ind[i]] , y = smoothy[[i]][ind[i]] + 1 ,
+        labels = label.day[i] , col = i , cex = 0.6)
+ }
```

B.3 Tweet Activity Outliers of November 2014

```
> options(width=65)
> # loading necessary libraries
> library(twitteR)
> library(graphics)
> library(tm)
> # load verification standard of twitter
> load("twitteR_credentials")
> registerTwitterOAuth(twitCred)
> # reading in all twitter extractions from november containing the keyword
> # "islamic_state" and identify the tweet activity outliers for each day
> # and the main terms contained in them
> days <- 1
> sum <- list()
> out <- list()
> peaktext <- list()
> termfreq <- list()
> toptermfreq <- list()
> ind <- list()
> tweets.november <- list()
> tweets.minute <- list()
> n <- vector()
> for(i in 1:days){
+   # importing routine
+   if(i < 10){
+     number <- paste("0" , i , sep = "")
+   }else{
+     number <- i
+   }
+   name <- paste("~/Uni/Master/Master Thesis/Coding/twitter_data/islamic_state/
+                 01.november_2014/isdf_2014-11-" , number , sep = "")
+   name <- paste(name , ".RData" , sep = "")
+   load(name)
+   data.frame <- isdf
+   remove(isdf)
+   # list containing the daily tweets
+   tweets.november[[i]] <- data.frame
+   tweets.minute[[i]] <- data.frame
+   # list containing the corresponding times (aggregated to minutes)
+   timestamp <- strptime(data.frame$created , "%Y-%m-%d %H:%M:%S")
+   tweets.minute[[i]]$created <- as.factor(format(timestamp , '%H:%M'))
+   n[i] <- length(levels(tweets.minute[[i]]$created))
+   sum[[i]] <- summary(tweets.minute[[i]]$created , maxsum = n[i])
+   # plot barplot to get an insight on very active time intervals
+   seq <- seq(n[i] , 1 , -60)
+   seq <- c(seq , 1)
+   mp <- barplot(sum[[i]] , xaxt = "n" , ylab = "number of tweets per minute" ,
+                 xlab = "time of day in UTC" , main = "tweet frequencies")
+   axis(side = 1 , at = mp[seq] , labels = levels(tweets.minute[[i]]$created)[seq]
+        , cex.axis = 0.7)
+   # observe outliers
+   boxplot(sum[[i]] , ylab ="number of tweets per minute" , main = "boxplot")
+   ind[[i]] <- which(sum[[i]] %in% boxplot.stats(sum[[i]])$out)
+   out[[i]] <- sum[[i]][ind[[i]]]
+   if(length(out[[i]]) != 0){
+     # examine top terms of peak times for each day
+     inda <- vector()
+     for (j in 1:length(out[[i]])){
+       inda <- append(inda , which(tweets.minute[[i]]$created ==
+                      names(out[[i]][j])))
+     }
+     peaktext[[i]] <- tweets.minute[[i]]$text[inda]
+     # building a corpus containing the outlier tweet texts
+     Corpus <- Corpus(VectorSource(peaktext[[i]]))
+     # convert to lower case
+     Corpus <- tm_map(Corpus , tolower)
+     # remove punctuation but not # and @
+     removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+     Corpus <- tm_map(Corpus , removesomepunct)
+     # remove numbers
+     Corpus <- tm_map(Corpus , removeNumbers)
+     # remove URLs
+     removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+     Corpus <- tm_map(Corpus , removeURL)
+     # remove stopwords adjusted where necessary
+     # including rt which stands for retweet and
+     # the keyword islamic and state which appear
+     # in every tweet
+     myStopwords <- c(stopwords("english") , "rt" , "islamic" , "state")
+     Corpus <- tm_map(Corpus , removeWords , myStopwords)
+     # building a term document matrix without weighting
+     DTM <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf)))
+     # calculating term frequency
+     termfreq[[i]] <- colSums(as.matrix(DTM))
+     # sort the terms decreasingly
+     termfreq[[i]] <- sort(termfreq[[i]] , decreasing = TRUE)
+     # only consider the ten most frequent terms
+     toptermfreq[[i]] <- termfreq[[i]][1:10]
+   }
+ }
> # get time of outliers
> for(i in 1:days){
+  print(i)
+  print(sum[[i]][ind[[i]]])
+  print(toptermfreq[[i]])
+ }
```

B.4 Development of most frequent terms in the tweets of November 2014

```
> # loading necessary libraries
> library(tm)
> # reading in all twitter extractions from november containing the keyword
> # "islamic_state" and have a look at the development of the most frequent
> # terms over time
> days <- 2
> termfreq <- list()
> toptermfreq <- list()
> n <- vector()
> for(i in 1:days){
+   # importing routine
+   if(i < 10){
+     number <- paste("0" , i , sep = "")
+   }else{
+     number <- i
+   }
+   name <- paste("~/Uni/Master/Master Thesis/Coding/twitter_data/islamic_state/
+                  01.november_2014/isdf_2014-11-" , number , sep = "")
+   name <- paste(name , ".RData" , sep = "")
+   load(name)
+   data.frame <- isdf
+   remove(isdf)
+   # building a corpus containing the tweet texts
+   Corpus <- Corpus(VectorSource(data.frame$text))
+   # convert to lower case
+   Corpus <- tm_map(Corpus , tolower)
+   # remove punctuation but not # and @
+   removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+   Corpus <- tm_map(Corpus , removesomepunct)
+   # remove numbers
+   Corpus <- tm_map(Corpus , removeNumbers)
+   # remove URLs
+   removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+   Corpus <- tm_map(Corpus , removeURL)
+   # remove stopwords adjusted where necessary
+   # including rt which stands for retweet and
+   # the keywords islamic and state which appear
+   # in every tweet
+   myStopwords <- c(stopwords("english") , "rt" , "islamic" , "state")
+   Corpus <- tm_map(Corpus , removeWords , myStopwords)
+   # building a term document matrix without weighting
+   DTM <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf)))
+   n[i] <- dim(DTM)[2]
+   # remove sparse terms to make the document term matrix more tractable
+   DTM <- removeSparseTerms(DTM , 0.99)
+   # counting the occurrence of the terms
+   termfreq[[i]] <- colSums(as.matrix(DTM))/n[i]
+   # save the terms which appear in more than ten percent of all tweets
+   # of this day
+   toptermfreq[[i]] <- subset(termfreq[[i]] , termfreq[[i]] >= 0.1)
+ }
> # clean up
> remove(data.frame)
> # create data frame containing the relative occurrence of every top term
> # of every day in the month
> names <- names(toptermfreq[[1]])
> for (i in 1:(days-1)){
+   names <- union(names , names(toptermfreq[[i+1]]))
+ }
> s <- length(names)
> df <- data.frame(row.names = names)
> for(i in 1:days){
+   t <- length(toptermfreq[[i]])
+   if(t > 0){
+     for (j in 1:t){
+       df[names(toptermfreq[[i]][j]) , i] <-  toptermfreq[[i]][j]
+     }
+     }else{
+       df[ , i] <- 0
+     }
+   }
> df[is.na(df)] <- 0
> # plotting the development of the most frequent terms over time
> sq <- seq(1 , s , 8)
> for (k in sq){
+   if((s-k)<8){
+     plot(y = df[k , ] , x = 1:days , type = "n" , xlab = "day in November 2014" ,
+          ylab = "relative frequency of terms" ,
+          ylim = c(0 , max(df[k:s , ])+0.05)) ,
+          main = "development of most frequent terms in november" ,
+     for (i in 1:8){
+       lines(y = df[k+i-1 , ] , x = 1:days , type = "b" , col = i)
+     }
+     legend("topright" , names[k:s] , bty = "n" , cex = 0.8 , col = 1:8 ,
+            lty = 1 , ncol = 2 , y.intersp = 0.3)
+   }else{
+     plot(y = df[k , ] , x = 1:days , type = "n" , xlab = "day in November 2014" ,
+          ylab = "relative frequency of terms" , cex.lab = 1.2 , cex.main = 1.5 ,
+          ylim = c(0 , max(df[k:(k+7),])+0.05) ,
+          main = "development of most frequent terms in november")
+     for (i in 1:8){
+       lines(y = df[k+i-1 , ] , x = 1:days , type = "b" , col = i)
+     }
+     legend("topright" , names[k:(k+7)] , bty = "n" , cex = 0.8 , col = 1:8 ,
+           lty = 1 , ncol = 2 ,  y.intersp = 0.3)
+   }
+ }
```

B.5 Determining sufficient amount of tweets to cover 95 % of all terms

```
> # loading necessary libraries
> library(tm)
> # define variables
> days <- 2
> suff <- vector(mode = "numeric" , length = days)
> for (l in 11:days){
+   # importing routine
+   if(l < 10){
+     number <- paste("0" , l , sep = "")
+   }else{
+     number <- l
+   }
+   name <- paste("~/Uni/Master/Master Thesis/Coding/twitter_data/islamic_state/
+                 01.november_2014/isdf_2014-11-" , number , sep = "")
+   name <- paste(name , ".RData" , sep = "")
+   load(name)
+   data.frame <- isdf
+   # delete retweets
+   ind <- which(data.frame$isRetweet == FALSE)
+   data.frame <- data.frame[ind , ]
+   # building a corpus containing the tweet texts
+   Corpus <- Corpus(VectorSource(data.frame$text))
+   # convert to lower case
+   Corpus <- tm_map(Corpus , tolower)
+   # remove punctuation but not # and @
+   removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+   Corpus <- tm_map(Corpus , removesomepunct)
+   # remove numbers
+   Corpus <- tm_map(Corpus , removeNumbers)
+   # remove URLs
+   removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+   Corpus <- tm_map(Corpus , removeURL)
+   # remove stopwords adjusted where necessary
+   # including the keywords islamic and state
+   # which  appear in every tweet
+   myStopwords <- c(stopwords("english") , "islamic" , "state")
+   Corpus <- tm_map(Corpus , removeWords , myStopwords)
+   # building a term document matrix with TfIdf weighting
+   DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf) ,
+                                                       weighting = weightTfIdf))
+   DTMw <- removeSparseTerms(DTMw , 0.99)
+   DTMw <- as.matrix(DTMw)
+   # identify number of tweets sufficient for 95% of all terms
+   n <- dim(DTMw)[1]
+   termdim <- dim(DTMw)[2]
+   if(n < 100){
+     seq <- seq(10 , n , 10)
+     seq <- c(2 , seq)
+     seq <- c(seq , n)
+   }else{
+     seq <- seq(100 , n , 100)
+     seq <- c(2 , seq)
+     seq <- c(seq , n)
+   }
+   for(i in 1:length(seq)){
+     if(length(which(!apply(DTMw[1:seq[i] , ] == 0 , 2 , all)))/termdim > 0.95){
+       suff[l] <- seq[i]/n
+       break
+     }
+   }
+   summary(suff)
+ }
```

B.6 Principal component Analysis and clustering of Islamic State tweets of November 2014

```
> # loading necessary libraries
> # package for text mining
> library(tm)
> # package for multivariate normality tests
> library(MVN)
> # package for rank computation of matrices
> library(Matrix)
> # package for trace computation of matrices
> library(psych)
> # package for medoids clustering
> library(fpc)
> # 3D plots package
> library(scatterplot3d)
> # define variables
> # number of days of examined month
> days <- 3
> # vector containing the percentage of normally distributed
> # columns of original normalized data matrix Z for each day
> meas1 <- vector(mode = "numeric" , length = days)
> # vector containing if reduced normalized data matrix Ztilde
> # is multivariate normally distributed for each day
> meas2 <- vector(mode = "numeric" , length = days)
> # vector containing if KMO index is larger 0.5 for each day
> meas3 <- vector(mode = "numeric" , length = days)
> # vector containing KMO index for each day
> KMO <- vector(mode = "numeric" , length = days)
> # vector containing dimensions of empirical correlation matrix R for each day
> dimensionR <- vector(mode = "numeric" , length = days)
> # vector containing number of eigenvalues of R larger 1 for each day
> k <- vector(mode = "numeric" , length = days)
> # vector containing captured variance by first 2 principal components
> # for each day
> captvar2 <- vector(mode = "numeric" , length = days)
> # vector containing captured variance by first 3 principal components
> # for each day
> captvar3 <- vector(mode = "numeric" , length = days)
> # vector containing captured variance by first k principal components
> # for each day
> captvark <- vector(mode = "numeric" , length = days)
> # vector containing extracted tweets on each day
> twpd <- vector(mode = "numeric" , length = days)
> # vector containing ratio of eigenvalues larger 1 =  k
> # and total tweets for each day
> ratio <- vector(mode = "numeric" , length = days)
> # vector containing ratio of eigenvalues larger 1 =  k
> # and dimension of R for each day
> reduct <- vector(mode = "numeric" , length = days)
> # vector containing number of detected clusters by silhouette
> # technqiue for 2 factors for each day
> kclus2 <- vector(mode = "numeric" , length = days)
> # vector containing number of detected clusters by silhouette
> # technqiue for 3 factors for each day
> kclus3 <- vector(mode = "numeric" , length = days)
> # vector containing number of detected clusters by silhouette
> # technqiue for k factors for each day
> kclusk <- vector(mode = "numeric" , length = days)
> # initiation for vector containing all the incomplete days
> seque <- 0
> # list containing all the eigenvalues of R for each day
> lambda <- list()
> # list containing column names of Ztilde, i.e. terms on this day for each day
> terms <- list()
> # list containing reduced normalized data matrix Ztilde for each day
> Ztilde <- list()
> # list containing normalized data matrix Ztilde with possible linear
> # dependent columns for each day
> Zdep <- list()
> # list containing empirical correlation matrix R for each day
> R <- list()
> # list containing matrix of principal component matrix F for each day
> F <- list()
> # list containing loading matrix L for each day
> L <- list()
> # list containing reduced loading matrix LHK (first k factors) for each day
> LHK <- list()
> # list containing high loadings of reduced loading matrix LHK for each day
> high<- list()
> # list containing the clustering result from the kmeans clustering
> # with two factors
> clusterkmeans2 <- list()
> # list containing the clustering result from the kmedoid clustering
> # with two factors
> clusterkmedoid2 <- list()
> # list containing the clustering result from the kmeans clustering
> # with three factors
> clusterkmeans3 <- list()
> # list containing the clustering result from the kmedoid clustering
> # with three factors
> clusterkmedoid3 <- list()
> # list containing the clustering result from the kmeans clustering
> # with k factors
> clusterkmeansk <- list()
> # list containing the clustering result from the kmedoid clustering
> # with k factors
> clusterkmedoidk <- list()
> # list containing the top 10 most frequent terms in the clusters resulting
> # from the kmeans clustering with 2 factors
> clustertermskmeans2 <- list()
> # list containing the top 10 most frequent terms in the clusters resulting
> # from the kmedoids clustering with 2 factors
> clustertermskmedoid2 <- list()
> # list containing the top 10 most frequent terms in the clusters resulting
> # from the kmeans clustering with 3 factors
> clustertermskmeans3 <- list()
> # list containing the top 10 most frequent terms in the clusters resulting
> # from the kmedoids clustering with 3 factors
> clustertermskmedoid3 <- list()
> # list containing the top 10 most frequent terms in the clusters resulting
> # from the kmeans clustering with k factors
> clustertermskmeansk <- list()
> # list containing the top 10 most frequent terms in the clusters resulting
> # from the kmedoids clustering with k factors
> clustertermskmedoidk <- list()
> for (l in 1:days){
+   # importing routine
+   if(l < 10){
+     number <- paste("0" , l , sep = "")
+   }else{
+     number <- 1
+   }
+   # adjust location of twitter data if necessary
+   name <- paste("~/Uni/Master/Master Thesis/Coding/twitter_data/islamic_state/
+                 01.november_2014/isdf_2014-11-"
+                 , number , sep = "")
+   name <- paste(name , ".RData" , sep = "")
+   load(name)
+   # exclude day if less than half of it could be extracted
+   last <- length(isdf$text)
+   timestamp <- strptime(isdf$created[last] , "%Y-%m-%d %H:%M:%S")
+   latest <- as.numeric(format(timestamp , '%H'))
+   if(is.na(latest) == TRUE){
+     latest <- 0
+   }
+   if(latest < 12){
+   # extracted number of tweets per day
+   twpd[l] <- length(isdf$text)
+   data.frame <- isdf
+   # delete retweets
+   ind <- which(data.frame$isRetweet == FALSE)
+   data.frame <- data.frame[ind , ]
+   # building a corpus containing the tweet texts
+   Corpus <- Corpus(VectorSource(data.frame$text))
+   # convert to lower case
+   Corpus <- tm_map(Corpus , tolower)
+   # remove punctuation but not # and @
+   removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+   Corpus <- tm_map(Corpus , removesomepunct)
+   # remove numbers
+   Corpus <- tm_map(Corpus , removeNumbers)
+   # remove URLs
+   removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+   Corpus <- tm_map(Corpus , removeURL)
+   # remove stopwords adjusted where necessary
+   # including the keywords islamic and state
+   # which  appear in every tweet
+   myStopwords <- c(stopwords("english") , "islamic" , "state")
+   Corpus <- tm_map(Corpus , removeWords , myStopwords)
+   # building a term document matrix with TfIdf weighting
+   DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf) ,
+                                                     weighting = weightTfIdf))
+   # remove sparse terms to make the document term matrix more tractable
+   DTMw <- removeSparseTerms(DTMw , 0.99)
+   DTMw <- as.matrix(DTMw)
+   # calculate the standardized data matrix Z
+   n <- dim(DTMw)[1]
+   m <- dim(DTMw)[2]
+   # calculate the empirical mean of the variables
+   mean <- colMeans(DTMw)
+   # calculate the empirical variance of the variables
+   s <- vector(mode = "numeric" , length = m)
+   for (i in 1:m){
+     s[i] <- sum((DTMw[ , i] - mean[i])^2)/(n-1)
+   }
+   Z <- matrix(nrow = n , ncol = m)
+   for (i in 1:m){
+     for (j in 1:n){
+       Z[j , i] <- (DTMw[j , i] - mean[i])/(sqrt(n-1)*sqrt(s[i]))
+     }
+   }
+   colnames(Z) <- colnames(DTMw)
+   # remove columns with NaN entries
+   Z <- Z[ , complete.cases(t(Z))]
+   Zdep[[l]] <- Z
+   # find linear dependent variables and delete them
+   rankMatrix(Z)
+   if(rankMatrix(Z) < m){
+     for (i in m:1){
+       if(rankMatrix(Z[ , -i]) == rankMatrix(Z)){
+         Z <- Z[ , -i]
+       }
+     }
+   }
+   # save Z with only linear independent columns as Ztilde
+   Ztilde[[l]] <- Z
+   rankMatrix(Ztilde[[l]])
+   # save remaining terms
+   terms[[l]] <- colnames(Ztilde[[l]])
+ m <- dim(Ztilde[[l]])[2]
+ # test if columns of Ztilde are normally distributed
+ for(i in 1:m){
+   if(ks.test(Ztilde[[l]][ , i] , y = 'pnorm' , alternative = 'two.sided')
+          $p.value < 0.1
+      ){
+   }else{
+     meas1[l] <- meas1[l] + 1
+   }
+ }
+ # how many of the columns are normally distributed (= meas1)
+ meas1[l] <- meas1[l]/m
+ # test if Ztilde is multivariate normally distributed (= meas2)
+ print(mardiaTest(Ztilde[[l]] , cov = TRUE , qqplot = TRUE))
+ if(mardiaTest(Ztilde[[l]] , cov = TRUE , qqplot = TRUE)@p.value.skew > 0.05 &
+ mardiaTest(Ztilde[[l]] , cov = TRUE , qqplot = TRUE)@p.value.kurt > 0.05){
+ meas2[l] <- 1
+  }
+ # calculate empirical correlation matrix
+ R[[l]] <- t(Ztilde[[l]])%*%Ztilde[[l]]
+ rankMatrix(R[[l]])
+ dimensionR[l] <- dim(R[[l]])[1]
+ # Bartlett's test if PCA makes sense
+ if(meas2[l] == 1){
+   chi <- -(n-1-(2*m+5)/6)*log(det(R[[l]]))
+   dof <- m*(m-1)/2
+   pchisq(chi , dof , lower.tail = FALSE)
+   qchisq(0.1 , dof , lower.tail = FALSE)
+   chi > qchisq(0.1 , dof , lower.tail = FALSE)
+ }
+ # KMO index
+ # calculate inverse of R
+ V <- solve(R[[l]])
+ # calculate partial correlation matrix due to Corollary 4.20
+ A <- matrix(nrow = m , ncol = m)
+ for(i in 1:m){
+   for(j in 1:m){
+     if(i == j){
+       A[i , j] <- 1-1/V[i , j]
+     }else{
```

```
+         A[i , j] <- -V[i , j]/(sqrt(V[i , i]*V[j , j]))
+       }
+     }
+   }
+   # calculate KMO index itself (meas3 = 1 if KMO higher 0.5)
+   qusur <- vector(mode = "numeric" , length = m)
+   qusua <- vector(mode = "numeric" , length = m)
+   for(i in 1:m){
+     for(j in 1:m){
+       if(i != j){
+         qusur[i] <- qusur[i] + R[[l]][i , j]^2
+         qusua[i] <- qusua[i] + A[i , j]^2
+       }
+     }
+   }
+   qusur <- sum(qusur)
+   qusua <- sum(qusua)
+   KMO[l] <- qusur/(qusur+qusua)
+   if(KMO[l] > 0.5){
+     meas3[l] <- 1
+   }
+   # PCA
+   # calculation of the eigenvalues
+   lambda[[l]] <- eigen(R[[l]])$value
+   Lambda <- diag(lambda[[l]])
+   # calculation of the eigenvectors
+   T <- eigen(R[[l]])$vectors
+   # compute the loading matrix
+   L[[l]] <- T %*% sqrt(Lambda)
+   # compute principal components
+   F[[l]] <- Ztilde[[l]]%*%(solve(R[[l]]))%*%L[[l]]
+   # plot eigenvalues to choose k
+   seq <- seq(5 , m , 5)
+   seq <- c(1 , seq)
+   plot(1:m , lambda[[l]] , xlab = "index" , ylab = "eigenvalues" , xaxt = "n"
+        , main = "Eigenvalues of R" , cex.lab = 1.2 , cex.main = 1.5)
+   axis(side = 1 , at = seq)
+   abline(1 , 0  ,lty = "dotted")
+   lines(lambda[[l]] , col = "red")
+   # how many eigenvalues are larger than 1
+   k[l] <- length(which(lambda[[l]] >= 1))
+   # ratio of k and total tweets
+   ratio[l] <- k[l]/twpd[l]
+   # how many variables are left in percent
+   reduct[l] <- k[l]/dimensionR[l]
+   # consider only first k components
+   # compute reduced loading matrix
+   LHK[[l]] <- L[[l]][ , 1:k[l]]
+   rownames(LHK[[l]]) <- c(colnames(Ztilde[[l]]))
+   colnames(LHK[[l]]) <- c(1:k[l])
+   # identify high loadings
+   high[[l]] <- which(abs(LHK[[l]][ , 1]) > 0.5)
+   high[[l]] <- names(high[[l]])
+   # compute reduced factor matrix
+   colnames(F[[l]][,1:2]) <- c(1:2)
+   # how much of the variance is captured with the first k principal components
+   captvark[l] <- sum(lambda[[l]][1:k[l]])/tr(R[[l]])
+   # how much of the variance is captured with the first 2 principal components
+   captvar2[l] <- sum(lambda[[l]][1:2])/tr(R[[l]])
+   # plot documents in new coordinates
+   plot(F[[l]][ , 1] , F[[l]][ , 2] , main = "Document Clustering" ,
+        cex.lab = 1.2 , cex.main = 1.5 , xlab = "Factor 1"  , ylab = "Factor 2" ,
+        xlim = c(min(F[[l]][ , 1]) , max(F[[l]][ , 1])) ,
+        ylim = c(min(F[[l]][ , 2]) , max(F[[l]][ , 2])))
+   # clustering via k-means and 2 factors
+   clus <- 4
+   set.seed(4221)
+   kmeansResult2 <- kmeans(F[[l]][ , 1:2] , clus)
+   clusterkmeans2[[l]] <- kmeansResult2$cluster
+   # plot documents with the colours of the respective clusters
+   plot(F[[l]][ , 1] , F[[l]][ , 2] , main = "Document Clustering k-means"  ,
+        xlab = "Factor 1"  , ylab = "Factor 2" , cex.lab = 1.2 , cex.main = 1.5 ,
+        xlim = c(min(F[[l]][ , 1]) , max(F[[l]][ , 1])) ,
+        ylim = c(min(F[[l]][ , 2]) , max(F[[l]][ , 2])) ,
+        col = clusterkmeans2[[l]])
+   # finding cluster names/topics
+   indc <- list()
+   clusdfkmeans2 <- data.frame(matrix(vector() , 4 , 10 , dimnames =
+                               list(c("Cluster 1" , "Cluster 2" , "Cluster 3" ,
+                               "Cluster 4") , c("TOP1" , "TOP2" , "TOP3" , "TOP4" ,
+                               "TOP5" , "TOP6" , "TOP7" , "TOP8" , "TOP9" ,
+                               "TOP10"))) , stringsAsFactors=F)
+   for(i in 1:clus){
+     indc[[i]] <- which(clusterkmeans2[[l]] == i)
+     # getting only the tweets of the respective cluster
+     dfc <- data.frame[indc[[i]] , ]
+     # building a corpus containing the tweet texts
+     Corpus <- Corpus(VectorSource(dfc$text))
+     # convert to lower case
+     Corpus <- tm_map(Corpus , tolower)
+     # remove punctuation but not # and @
+     removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+     Corpus <- tm_map(Corpus , removesomepunct)
+     # remove numbers
+     Corpus <- tm_map(Corpus , removeNumbers)
+     # remove URLs
+     removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+     Corpus <- tm_map(Corpus , removeURL)
+     # remove stopwords adjusted where necessary
+     # including the keywords islamic and state
+     # which  appear in every tweet
+     myStopwords <- c(stopwords("english") , "islamic" , "state")
+     Corpus <- tm_map(Corpus , removeWords , myStopwords)
+     # building a term document matrix with TfIdf weighting
+     DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf)))
+     # detecting most frequent terms in the respective cluster
+     termfrequency <- colSums(as.matrix(DTMw))
+     termfrequency <- sort(termfrequency , decreasing = TRUE)
+     # save top ten words in each cluster
+     clusdfkmeans2[i , 1:10] <- names(termfrequency[1:10])
+   }
+   clustertermskmeans2[[l]] <- clusdfkmeans2
+   # summary of how many tweets are in each cluster
+   summary(as.factor(clusterkmeans2[[l]]))
+   # clustering via k-medoids and 2 factors
+   set.seed(4222)
+   pamResultr2 <- pamk(F[[l]][ , 1:2] , metric = "manhattan")
+   # number of clusters identified
+   kclus2[[l]] <- pamResultr2$nc
+   pamResult2 <- pamResultr2$pamobject
+   clusterkmedoid2[[l]] <- pamResult2$clustering
+   # plot documents with the colours of the respective clusters
+   plot(F[[l]][ , 1] , F[[l]][ , 2] , main = "Document Clustering Medoids" ,
+        xlab = "Factor 1"  , ylab = "Factor 2" , cex.lab = 1.2 , cex.main = 1.5 ,
+        xlim = c(min(F[[l]][ , 1]) , max(F[[l]][ , 1])) ,
+        ylim = c(min(F[[l]][ , 2]) , max(F[[l]][ , 2])) ,
+        col = clusterkmedoid2[[l]])
+   # finding cluster names/topics
+   indc <- list()
+   clusdfkmedoids2 <- data.frame(matrix(vector() , kclus2[[l]] , 10 , dimnames =
+                                 list(c() , c("TOP1" , "TOP2" , "TOP3" , "TOP4" ,
+                                              "TOP5" , "TOP6" , "TOP7" , "TOP8" ,
+                                              "TOP9" , "TOP10"))) ,
+                                 stringsAsFactors=F)
+   for(i in 1:kclus2[[l]]){
+     indc[[i]] <- which(clusterkmedoid2[[l]] == i)
+     # getting only the tweets of the respective cluster
+     dfc <- data.frame[indc[[i]] , ]
+     # building a corpus containing the tweet texts
+     Corpus <- Corpus(VectorSource(dfc$text))
+     # convert to lower case
+     Corpus <- tm_map(Corpus , tolower)
+     # remove punctuation but not # and @
+     removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+     Corpus <- tm_map(Corpus , removesomepunct)
+     # remove numbers
+     Corpus <- tm_map(Corpus , removeNumbers)
+     # remove URLs
+     removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+     Corpus <- tm_map(Corpus , removeURL)
+     # remove stopwords adjusted where necessary
+     # including the keywords islamic and state
+     # which  appear in every tweet
+     myStopwords <- c(stopwords("english") , "islamic" , "state")
+     Corpus <- tm_map(Corpus , removeWords , myStopwords)
+     # building a term document matrix with TfIdf weighting
+     DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf)))
+     # detecting most frequent terms in the respective cluster
+     termfrequency <- colSums(as.matrix(DTMw))
+     termfrequency <- sort(termfrequency , decreasing = TRUE)
+     # save top ten words in each cluster
+     clusdfkmedoids2[i , 1:10] <- names(termfrequency[1:10])
+   }
+   clustertermskmedoid2[[l]] <- clusdfkmedoids2
+   # summary of how many tweets are in each cluster
+   summary(as.factor(clusterkmedoid2[[l]]))
+   # consider the first 3 components
+   # how much of the variance is captured with the first 3 principal components
+   captvar3[l] <- sum(lambda[[l]][1:3])/tr(R[[l]])
+   # 3D plot of documents in new coordinates
+   scatterplot3d(F[[l]][ , 1] , F[[l]][ , 2] , F[[l]][ , 3] ,
+                 xlab = "Factor 1" , ylab = "Factor 2" , zlab = "Factor 3" ,
+                 main = "Document Clustering" , cex.lab = 1.2 , cex.main = 1.5)
+   # clustering with regard to the first 3 components
+   # clustering via k-means and 3 factors
+   clus <- 5
+   set.seed(4331)
+   kmeansResult3 <- kmeans(F[[l]][ , 1:3] , clus)
+   clusterkmeans3[[l]] <- kmeansResult3$cluster
+   # plot documents with the colours of the respective clusters
+   scatterplot3d(F[[l]][ , 1] , F[[l]][ , 2] , F[[l]][ , 3] ,
+                 xlab = "Factor 1" , ylab = "Factor 2" , zlab = "Factor 3" ,
+                 main = "Document Clustering k-means" , cex.lab = 1.2 ,
+                 color = clusterkmeans3[[l]] ,  cex.main = 1.5)
+   # finding cluster names/topics
+   indc <- list()
+   clusdfkmeans3 <- data.frame(matrix(vector() , 5 , 10 , dimnames =
+                               list(c("Cluster 1" , "Cluster 2" , "Cluster 3" ,
+                               "Cluster 4" , "Cluster 5") , c("TOP1" , "TOP2" ,
+                               "TOP3" , "TOP4" , "TOP5" , "TOP6" , "TOP7" ,
+                               "TOP8" , "TOP9" , "TOP10"))) , stringsAsFactors=F)
+   for(i in 1:clus){
+     indc[[i]] <- which(clusterkmeans3[[l]] == i)
+     # getting only the tweets of the respective cluster
+     dfc <- data.frame[indc[[i]] , ]
+     # building a corpus containing the tweet texts
+     Corpus <- Corpus(VectorSource(dfc$text))
+     # convert to lower case
+     Corpus <- tm_map(Corpus , tolower)
+     # remove punctuation but not # and @
+     removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+     Corpus <- tm_map(Corpus , removesomepunct)
+     # remove numbers
+     Corpus <- tm_map(Corpus , removeNumbers)
+     # remove URLs
+     removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+     Corpus <- tm_map(Corpus , removeURL)
+     # remove stopwords adjusted where necessary
+     # including the keywords islamic and state
+     # which  appear in every tweet
+     myStopwords <- c(stopwords("english") , "islamic" , "state")
+     Corpus <- tm_map(Corpus , removeWords , myStopwords)
+     # building a term document matrix with TfIdf weighting
+     DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf)))
+     # detecting most frequent terms in the respective cluster
+     termfrequency <- colSums(as.matrix(DTMw))
+     termfrequency <- sort(termfrequency , decreasing = TRUE)
+     # save top ten words in each cluster
+     clusdfkmeans3[i , 1:10] <- names(termfrequency[1:10])
+   }
+   clustertermskmeans3[[l]] <- clusdfkmeans3
+   # summary of how many tweets are in each cluster
+   summary(as.factor(clusterkmeans3[[l]]))
+   # clustering via k-medoids and 3 factors
+   set.seed(4332)
+   pamResultr3 <- pamk(F[[l]][ , 1:3] , metric = "manhattan")
+   # number of clusters identified
+   kclus3[[l]] <- pamResultr3$nc
+   pamResult3 <- pamResultr3$pamobject
+   clusterkmedoid3[[l]] <- pamResult3$clustering
+   # plot documents with the colours of the respective clusters
+   scatterplot3d(F[[l]][ , 1] , F[[l]][ , 2] , F[[l]][ , 3] ,
+                 xlab = "Factor 1" , ylab = "Factor 2" , zlab = "Factor 3" ,
+                 main = "Document Clustering Medoids" , cex.lab = 1.2 , cex.main =
+                 color = clusterkmedoid3[[l]])
+   # finding cluster names/topics
+   indc <- list()
+   clusdfkmedoids3 <- data.frame(matrix(vector() , kclus3[[l]] , 10 , dimnames =
+                                 list(c() , c("TOP1" , "TOP2" , "TOP3" , "TOP4" ,
+                                      "TOP5" , "TOP6" , "TOP7" , "TOP8" , "TOP9" ,
+                                     "TOP10"))) , stringsAsFactors=F)
+   for(i in 1:kclus3[[l]]){
+     indc[[i]] <- which(clusterkmedoid3[[l]] == i)
+     # getting only the tweets of the respective cluster
+     dfc <- data.frame[indc[[i]] , ]
+     # building a corpus containing the tweet texts
+     Corpus <- Corpus(VectorSource(dfc$text))
+     # convert to lower case
+     Corpus <- tm_map(Corpus , tolower)
+     # remove punctuation but not # and @
+     removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+     Corpus <- tm_map(Corpus , removesomepunct)
+     # remove numbers
+     Corpus <- tm_map(Corpus , removeNumbers)
+     # remove URLs
+     removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+     Corpus <- tm_map(Corpus , removeURL)
+     # remove stopwords adjusted where necessary
+     # including the keywords islamic and state
+     # which  appear in every tweet
+     myStopwords <- c(stopwords("english") , "islamic" , "state")
+     Corpus <- tm_map(Corpus , removeWords , myStopwords)
+     # building a term document matrix with TfIdf weighting
+     DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf)))
+     # detecting most frequent terms in the respective cluster
+     termfrequency <- colSums(as.matrix(DTMw))
+     termfrequency <- sort(termfrequency , decreasing = TRUE)
+     # save top ten words in each cluster
+     clusdfkmedoids3[i , 1:10] <- names(termfrequency[1:10])
+   }
+   clustertermskmedoid3[[l]] <- clusdfkmedoids3
+   # summary of how many tweets are in each cluster
+   summary(as.factor(clusterkmedoid3[[l]]))
+   # clustering with regard to the first k components
+   # clustering via k-means and k factors
+   clus <- 5
+   set.seed(4991)
+   kmeansResultk <- kmeans(F[[l]][ , 1:15] , clus)
+   clusterkmeansk[[l]] <- kmeansResultk$cluster
+   # finding cluster names/topics
+   indc <- list()
+   clusdfkmeansk <- data.frame(matrix(vector() , 5 , 10 , dimnames =
+                               list(c("Cluster 1" , "Cluster 2" , "Cluster 3" ,
+                               "Cluster 4" , "Cluster 5") , c("TOP1" , "TOP2" ,
+                               "TOP3" , "TOP4" , "TOP5" , "TOP6" , "TOP7" ,
+                               "TOP8" , "TOP9" , "TOP10"))) , stringsAsFactors=F)
+   for(i in 1:clus){
+     indc[[i]] <- which(clusterkmeansk[[1]] == i)
+     # getting only the tweets of the respective cluster
+     dfc <- data.frame[indc[[i]] , ]
+     # building a corpus containing the tweet texts
+     Corpus <- Corpus(VectorSource(dfc$text))
+     # convert to lower case
+     Corpus <- tm_map(Corpus , tolower)
+     # remove punctuation but not # and @
+     removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+     Corpus <- tm_map(Corpus , removesomepunct)
+     # remove numbers
+     Corpus <- tm_map(Corpus , removeNumbers)
+     # remove URLs
+     removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+     Corpus <- tm_map(Corpus , removeURL)
+     # remove stopwords adjusted where necessary
+     # including the keywords islamic and state
+     # which  appear in every tweet
+     myStopwords <- c(stopwords("english") , "islamic" , "state")
+     Corpus <- tm_map(Corpus , removeWords , myStopwords)
+     # building a term document matrix with TfIdf weighting
+     DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf)))
+     # detecting most frequent terms in the respective cluster
+     termfrequency <- colSums(as.matrix(DTMw))
+     termfrequency <- sort(termfrequency , decreasing = TRUE)
+     # save top ten words in each cluster
+     clusdfkmeansk[i , 1:10] <- names(termfrequency[1:10])
+   }
+   clustertermskmeansk[[1]] <- clusdfkmeansk
+   # summary of how many tweets are in each cluster
+   summary(as.factor(clusterkmeansk[[1]]))
+   # clustering via k-medoids and k factors
+   set.seed(4992)
+   pamResultrk <- pamk(F[[l]][ , 1:15] , metric = "manhattan")
+   # number of clusters identified
+   kclusk[[l]] <- pamResultrk$nc
+   pamResultk <- pamResultrk$pamobject
+   clusterkmedoidk[[l]] <- pamResultk$clustering
+   # finding cluster names/topics
+   indc <- list()
+   clusdfkmedoidsk <- data.frame(matrix(vector() , kclusk[[l]] , 10 , dimnames =
+                                 list(c() , c("TOP1" , "TOP2" , "TOP3" , "TOP4" ,
+                                              "TOP5" , "TOP6" , "TOP7" , "TOP8" ,
+                                              "TOP9" , "TOP10"))) ,
+                                 stringsAsFactors=F)
+   for(i in 1:kclusk[[l]]){
+     indc[[i]] <- which(clusterkmedoidk[[l]] == i)
+     # getting only the tweets of the respective cluster
+     dfc <- data.frame[indc[[i]] , ]
+     # building a corpus containing the tweet texts
+     Corpus <- Corpus(VectorSource(dfc$text))
+     # convert to lower case
+     Corpus <- tm_map(Corpus , tolower)
+     # remove punctuation but not # and @
+     removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+     Corpus <- tm_map(Corpus , removesomepunct)
+     # remove numbers
+     Corpus <- tm_map(Corpus , removeNumbers)
+     # remove URLs
+     removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+     Corpus <- tm_map(Corpus , removeURL)
+     # remove stopwords adjusted where necessary
+     # including the keywords islamic and state
+     # which  appear in every tweet
+     myStopwords <- c(stopwords("english") , "islamic" , "state")
+     Corpus <- tm_map(Corpus , removeWords , myStopwords)
+     # building a term document matrix with TfIdf weighting
+     DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf)))
+     # detecting most frequent terms in the respective cluster
+     termfrequency <- colSums(as.matrix(DTMw))
+     termfrequency <- sort(termfrequency , decreasing = TRUE)
+     # save top ten words in each cluster
+     clusdfkmedoidsk[i , 1:10] <- names(termfrequency[1:10])
+   }
+   clustertermskmedoidk[[l]] <- clusdfkmedoidsk
+   # summary of how many tweets are in each cluster
+   summary(as.factor(clusterkmedoidk[[l]]))
+   }else{
+     # print insufficient days
+     print(l)
+     seque <- c(seque , l)
+   }
+ }
> # days with insufficient data
> seque <- seque[-1]
> # comparison between the dimension of the columns of Z and Ztilde
> seque2 <- c(1:days)
> seque2 <- seque2[-seque]
> dimz.clean <- vector(mode = "numeric" , length = length(seque2))
> dimztil.clean <- vector(mode = "numeric" , length = length(seque2))
> for(i in seque2){
+   dimz.clean[i] <- dim(Zdep[[i]])[2]
+   dimztil.clean[i] <- dim(Ztilde[[i]])[2]
+ }
> dimz.clean <- dimz.clean[seque2]
> dimztil.clean <- dimztil.clean[seque2]
> summary(dimztil.clean/dimz.clean)
> plot(y = dimz.clean , x = 1:length(seque2) , type = "b" , main = "dimension
+      comparison of Z and Z~" , xlab = "day in November 2014" ,
+      ylab = "number of columns" , ylim = c(0,max(dimz.clean)) , cex.lab = 1.2 ,
+      cex.main = 1.5 , xaxt = "n")
> lines(y = dimztil.clean , x = 1:length(seque2) , type = "b" , col = "red")
> axis(side = 1 , at = 1:length(seque2) , labels = seque2)
> legend("bottomright" , legend = c("Z" , "Z~") , bty = "n" , cex = 0.8 ,
+         col = 1:8 , y.intersp = 0.3 , lty = 1)
> # summary and plots of certain figures
> # KMO index
> KMO.clean <- KMO[seque2]
> summary(KMO.clean)
> plot(x = 1:length(seque2) , y = KMO.clean , main = "KMO index for November 2014
+      tweets of Islamic State" , xlab = "day in November 2014" , ylab =
+      "KMO index" , cex.main = 1.5 , type = "b" , xaxt = "n"  , cex.lab = 1.2 )
> axis(side = 1 , at = 1:length(seque2) , labels = seque2)
> # Eigenvalues larger 1
> k.clean <- k[seque2]
> summary(k.clean/dimztil.clean)
> plot(y = k.clean , x = 1:length(seque2) , xlab = "day in November 2014" ,
+      ylab = "k", main = "Eigenvalues of R larger one", cex.lab = 1.2 ,
+      cex.main = 1.5 , type = "b" , xaxt = "n")
> axis(side = 1 , at = 1:length(seque2) , labels = seque2)
> # plot of ratio of k and m tilde
> plot(y = k.clean/dimztil.clean , x = 1:length(seque2) , xlab = "day in November
+      2014" , ylab = "k/m~" , main = "Ratio of k and m~", cex.lab = 1.2 ,
+      cex.main = 1.5 , type = "b" , xaxt = "n")
> axis(side = 1 , at = 1:length(seque2) , labels = seque2)
> # captured variance with 2 factors
> captvar2.clean <- captvar2[seque2]
> summary(captvar2.clean)
> plot(y = captvar2.clean , x = 1:length(seque2) , xlab = "day in November 2014" ,
+      ylab = "captured variance", cex.lab = 1.2 ,
+      main = "Captured variance by the first 2 Principal Components",
+      cex.main = 1.5 , type = "b" , xaxt = "n")
> axis(side = 1 , at = 1:length(seque2) , labels = seque2)
> # captured variance with 3 factors
> captvar3.clean <- captvar3[seque2]
> summary(captvar3.clean)
> plot(y = captvar3.clean , x = 1:length(seque2) , xlab = "day in November 2014" ,
+      ylab = "captured variance", main = "Captured variance by the first 3
+      Principal Components", cex.lab = 1.2 , cex.main = 1.5 , type = "b",
+      xaxt = "n")
> axis(side = 1 , at = 1:length(seque2) , labels = seque2)
> # captured variance with k components
> captvark.clean <- captvark[seque2]
> summary(captvark.clean)
> plot(y = captvark.clean , x = 1:length(seque2) , xlab = "day in November 2014" ,
+      ylab = "captured variance", main = "Captured variance by the first k
+      Principal Components", cex.lab = 1.2 , cex.main = 1.5 , type = "b",
+      xaxt = "n")
> axis(side = 1 , at = 1:length(seque2) , labels = seque2)
> # clean up
> remove(i)
> remove(j)
> remove(k)
> remove(l)
> remove(m)
> remove(n)
> remove(days)
> remove(clus)
> remove(qusur)
> remove(qusua)
> remove(last)
> remove(latest)
> remove(name)
> remove(number)
> remove(mean)
> remove(s)
> remove(seq)
> remove(seque)
> remove(seque2)
> remove(timestamp)
> remove(ind)
> remove(myStopwords)
> remove(A)
> remove(DTMw)
> remove(Lambda)
> remove(T)
> remove(V)
> remove(Z)
> remove(termfrequency)
> remove(data.frame)
> remove(isdf)
> remove(dfc)
> remove(clusdfkmeans2)
> remove(clusdfkmeans3)
> remove(clusdfkmeansk)
> remove(clusdfkmedoids2)
> remove(clusdfkmedoids3)
> remove(clusdfkmedoidsk)
> remove(kmeansResult2)
> remove(kmeansResult3)
> remove(kmeansResultk)
> remove(pamResult2)
> remove(pamResultr2)
> remove(pamResult3)
> remove(pamResultr3)
> remove(pamResultk)
> remove(pamResultrk)
> remove(indc)
> remove(Corpus)
> remove(removeURL)
> remove(removesomepunct)
```

B.7 Produce clusters with training data and assess remaning data automatically

```
> # loading necessary libraries
> library(tm)
> library(MVN)
> library(Matrix)
> library(psych)
> # which day of november shall be considered
> l <- 16
> # importing routine
> if(l < 10){
+   number <- paste("0" , l , sep = "")
+ }else{
+   number <- l
+ }
> name <- paste("~/Uni/Master/Master Thesis/Coding/twitter_data/islamic_state/
+               01.november_2014/isdf_2014-11-" , number , sep = "")
> name <- paste(name , ".RData" , sep = "")
> load(name)
> # extracted number of tweets per day
> twpd <- length(isdf$text)
> data.frame <- isdf
> train <- 0.35*twpd
> # take first 35% of data for training
> data.frame.train <- data.frame[1:train , ]
> # delete retweets
> ind <- which(data.frame.train$isRetweet == FALSE)
> data.frame.train <- data.frame.train[ind , ]
> # building a corpus containing the tweet texts
> Corpus <- Corpus(VectorSource(data.frame.train$text))
> # convert to lower case
> Corpus <- tm_map(Corpus , tolower)
> # remove punctuation but not # and @
> removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
> Corpus <- tm_map(Corpus , removesomepunct)
> # remove numbers
> Corpus <- tm_map(Corpus , removeNumbers)
> # remove URLs
> removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
> Corpus <- tm_map(Corpus , removeURL)
> # remove stopwords adjusted where necessary
> # including the keywords islamic and state
> # which  appear in every tweet
> myStopwords <- c(stopwords("english") , "islamic" , "state")
> Corpus <- tm_map(Corpus , removeWords , myStopwords)
> # building a term document matrix with TfIdf weighting
> DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf) ,
+                                                    weighting = weightTfIdf))
> # remove sparse terms to make the document term matrix more tractable
> DTMw <- removeSparseTerms(DTMw , 0.99)
> DTMw <- as.matrix(DTMw)
> # calculate the standardized data matrix Z
> n <- dim(DTMw)[1]
> m <- dim(DTMw)[2]
> # calculate the empirical mean of the variables
> mean <- colMeans(DTMw)
> # calculate the empirical variance of the variables
> s <- vector(mode = "numeric" , length = m)
> for (i in 1:m){
+   s[i] <- sum((DTMw[ , i] - mean[i])^2)/(n-1)
+ }
> Z <- matrix(nrow = n , ncol = m)
> for (i in 1:m){
+   for (j in 1:n){
+     Z[j , i] <- (DTMw[j , i] - mean[i])/(sqrt(n-1)*sqrt(s[i]))
+   }
+ }
> colnames(Z) <- colnames(DTMw)
> # remove columns with NaN entries
> Z <- Z[ , complete.cases(t(Z))]
> # find linear dependent variables and delete them
> rankMatrix(Z)
> if(rankMatrix(Z) < m){
+   for (i in m:1){
+     if(rankMatrix(Z[ , -i]) == rankMatrix(Z)){
+       Z <- Z[ , -i]
+     }
+   }
+ }
```

```
> rankMatrix(Z)
> # save remaining terms
> terms <- colnames(Z)
> m <- dim(Z)[2]
> # test if columns of Z are normally distributed
> for(i in 1:m){
+   if(ks.test(Z[ , i] , y = 'pnorm' , alternative = 'two.sided')$p.value < 0.1){
+   }else{
+     meas1 <- meas1 + 1
+   }
+ }
> Zred <- Z
> # calculate empirical correlation matrix
> R <- t(Zred)%*%Zred
> rankMatrix(R)
> dimensionR <- dim(R)[1]
> # KMO index
> # calculate inverse of R
> V <- solve(R)
> # calculate partial correlation matrix due to Corollary 4.18
> A <- matrix(nrow = m , ncol = m)
> for(i in 1:m){
+   for(j in 1:m){
+     if(i == j){
+       A[i , j] <- 1-1/V[i , j]
+     }else{
+       A[i , j] <- -V[i , j]/(sqrt(V[i , i]*V[j , j]))
+     }
+   }
+ }
> # calculate KMO index itself (meas3 = 1 if KMO higher 0.5)
> qusur <- vector(mode = "numeric" , length = m)
> qusua <- vector(mode = "numeric" , length = m)
> for(i in 1:m){
+   for(j in 1:m){
+     if(i != j){
+       qusur[i] <- qusur[i] + R[i , j]^2
+       qusua[i] <- qusua[i] + A[i , j]^2
+     }
+   }
+ }
> qusur <- sum(qusur)
> qusua <- sum(qusua)
> KMO <- qusur/(qusur+qusua)
> if(KMO > 0.5){
+   meas3 <- 1
+ }
> # PCA
> # calculation of the eigenvalues
> lambda <- eigen(R)$value
> Lambda <- diag(lambda)
> # calculation of the eigenvectors
> T <- eigen(R)$vectors
> # compute the loading matrix
> L <- T %*% sqrt(Lambda)
> # compute principal components
> F <- Zred%*%(solve(R))%*%L
> # plot eigenvalues to choose k
> seq <- seq(5 , m , 5)
> seq <- c(1 , seq)
> plot(1:m , lambda , xlab = "index" , ylab = "eigenvalues" , xaxt = "n"
+      , main = "Eigenvalues of R")
> axis(side = 1 , at = seq)
> abline(1 , 0  ,lty = "dotted")
> lines(lambda , col = "red")
> # which eigenvalues are bigger than 1
> k <- length(which(lambda >= 1))
> # ratio of k and total tweets
> ratio <- k/twpd
> # how many variables are left in percent
> reduct <- k/dimensionR
> # consider only first k components
> # compute reduced loading matrix
> LHK <- L[ , 1:k]
> rownames(LHK) <- c(colnames(Zred))
> colnames(LHK) <- c(1:k)
> # identify high loadings
> high <- which(abs(LHK[ , 1]) > 0.5)
> high <- names(high)
> # compute reduced factor matrix
> colnames(F[,1:2]) <- c(1:2)
> # how much of the variance is captured with the first k principal components
> captvark <- sum(lambda[1:k])/tr(R)
> # how much of the variance is captured with the first 2 principal components
> captvar2 <- sum(lambda[1:2])/tr(R)
> # plot documents in new coordinates
> set.seed(9876)
> #names <- c(1:n)
> plot(F[ , 1] , F[ , 2] , main = "Document Clustering" ,
+      xaxt = "n" , yaxt = "n" , xlab = "Factor 1"  , ylab = "Factor 2" ,
+      xlim = c(min(F[ , 1]) , max(F[ , 1])) ,
+      ylim = c(min(F[ , 2]) , max(F[ , 2])))
> # clustering via k-means
> clus <- 4
> set.seed(8798)
> kmeansResult <- kmeans(F[ , 1:2] , clus)
> clusterkmeans <- kmeansResult$cluster
> plot(F[ , 1] , F[ , 2] , main = "Document Clustering k-means" ,
+      xaxt = "n" , yaxt = "n" , xlab = "Factor 1"  , ylab = "Factor 2" ,
+      xlim = c(min(F[ , 1]) , max(F[ , 1])) ,
+      ylim = c(min(F[ , 2]) , max(F[ , 2])) ,
+      col = clusterkmeans)
> # finding cluster names/topics
> indc <- list()
> clusdfkmeans <- data.frame(matrix(vector() , 4 , 10 , dimnames =
+                           list(c("Cluster 1" , "Cluster 2" , "Cluster 3" ,
+                                  "Cluster 4") , c("TOP1" , "TOP2" , "TOP3" ,
+                                  "TOP4" , "TOP5" , "TOP6" , "TOP7" , "TOP8" ,
+                                  "TOP9" , "TOP10"))) , stringsAsFactors = F)
> for(i in 1:clus){
+   indc[[i]] <- which(clusterkmeans == i)
+   # getting only the tweets of the respective cluster
+   dfc <- data.frame.train[indc[[i]] , ]
+   # building a corpus containing the tweet texts
+   Corpus <- Corpus(VectorSource(dfc$text))
+   # convert to lower case
+   Corpus <- tm_map(Corpus , tolower)
+   # remove punctuation but not # and @
+   removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+   Corpus <- tm_map(Corpus , removesomepunct)
+   # remove numbers
+   Corpus <- tm_map(Corpus , removeNumbers)
+   # remove URLs
+   removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+   Corpus <- tm_map(Corpus , removeURL)
+   # remove stopwords adjusted where necessary
+   # including the keywords islamic and state
+   # which  appear in every tweet
+   myStopwords <- c(stopwords("english") , "islamic" , "state")
+   Corpus <- tm_map(Corpus , removeWords , myStopwords)
+   # building a term document matrix with TfIdf weighting
+   DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf)))
+   # detecting most frequent terms in the respective cluster
+   termfrequency <- colSums(as.matrix(DTMw))
+   termfrequency <- sort(termfrequency , decreasing = TRUE)
+   clusdfkmeans[i , 1:10] <- names(termfrequency[1:10])
+ }
> clustertermskmeans <- clusdfkmeans
> # classify the remaining 75% of the tweets but first bring test data
> # into suitable form
> data.frame.test <- data.frame[(train + 1):twpd , ]
> # building a corpus containing the tweet texts
> Corpus <- Corpus(VectorSource(data.frame.test$text))
> # convert to lower case
> Corpus <- tm_map(Corpus , tolower)
> # remove punctuation but not # and @
> removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
> Corpus <- tm_map(Corpus , removesomepunct)
> # remove numbers
> Corpus <- tm_map(Corpus , removeNumbers)
> # remove URLs
> removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
> Corpus <- tm_map(Corpus , removeURL)
> # remove stopwords adjusted where necessary
> # including the keywords islamic and state
> # which  appear in every tweet
> myStopwords <- c(stopwords("english") , "islamic" , "state")
> Corpus <- tm_map(Corpus , removeWords , myStopwords)
> # building a term document matrix with TfIdf weighting
> DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf) ,
+                                                   weighting = weightTfIdf))
> # remove sparse terms to make the document term matrix more tractable
> DTMw <- removeSparseTerms(DTMw , 0.99)
> DTMw <- as.matrix(DTMw)
> # adjust DTM so its comparable to the DTM of the training set
> DTMw.adj <- DTMw[ , (colnames(DTMw) %in% rownames(LHK))]
> missing.terms <- rownames(LHK)[!(rownames(LHK) %in% colnames(DTMw))]
> missing.terms <- data.frame(matrix(vector() , dim(DTMw)[1] ,
+                                    length(missing.terms) , dimnames =
+                             list(c() , missing.terms)) , stringsAsFactors = F)
> missing.terms[is.na(missing.terms)] <- 0
> colnames(missing.terms) <- gsub("X." , "#" , colnames(missing.terms))
> DTMw.adj <- cbind(DTMw.adj , missing.terms)
> DTMw.adj <- DTMw.adj[ , sort(colnames(DTMw.adj))]
> # calculate the standardized data matrix Z
> n <- dim(DTMw.adj)[1]
> m <- dim(DTMw.adj)[2]
> # calculate the empirical mean of the variables
> mean <- colMeans(DTMw.adj)
> # calculate the empirical variance of the variables
> s <- vector(mode = "numeric" , length = m)
> for (i in 1:m){
+   s[i] <- sum((DTMw.adj[ , i] - mean[i])^2)/(n-1)
+ }
> Z <- matrix(nrow = n , ncol = m)
> for (i in 1:m){
+   for (j in 1:n){
+     Z[j , i] <- (DTMw.adj[j , i] - mean[i])/(sqrt(n-1)*sqrt(s[i]))
+   }
+ }
> colnames(Z) <- colnames(DTMw.adj)
> Z[is.na(Z)] <- 0
> # compute coordinates of test tweets in the coordinates of the PCA of
> # the training data
> F.new <- Z%*%(solve(R))%*%LHK
> # plotting new tweets
> set.seed(9876)
> plot(F[ , 1] , F[ , 2] , main = "Document Clustering" ,
+      xaxt = "n" , yaxt = "n" , xlab = "Factor 1"  , ylab = "Factor 2" ,
+      xlim = c(min(F[ , 1]) , max(F[ , 1])) ,
+      ylim = c(min(F[ , 2]) , max(F[ , 2])) , col = clusterkmeans)
> points(F.new[6 , 1] , F.new[6 , 2] , col = "magenta" , pch = 2 ,
+        cex = 3 , lwd = 2)
> points(F.new[394 , 1] , F.new[394 , 2] , col = "magenta" , pch = 3 ,
+        cex = 3 , lwd = 2)
> points(F.new[21 , 1] , F.new[552 , 2] , col = "magenta" , pch = 4 ,
+        cex = 3 , lwd = 2)
> # clean up
> remove(i)
> remove(j)
> remove(k)
> remove(l)
> remove(m)
> remove(n)
> remove(train)
> remove(clus)
> remove(qusur)
> remove(qusua)
> remove(name)
> remove(number)
> remove(mean)
> remove(s)
> remove(seq)
> remove(ind)
> remove(myStopwords)
> remove(A)
> remove(DTMw)
> remove(DTMw.adj)
> remove(Lambda)
> remove(T)
> remove(V)
> remove(Z)
> remove(termfrequency)
> remove(data.frame)
> remove(isdf)
> remove(dfc)
> remove(clusdfkmeans)
> remove(kmeansResult)
> remove(indc)
> remove(Corpus)
> remove(removeURL)
> remove(removesomepunct)
```

B.8 Examination of deaths due to Ebola in certain countries

```
> # loading necessary libraries
> library(tm)
> # define variables
> days <- 30
> reltermfreq <- list()
> termfreq <- list()
> country <- data.frame(matrix(ncol = 3, nrow = 92))
> colnames(country) <- c("Guinea" , "Liberia" , "Sierra Leone")
> twpd <- vector(mode = "numeric" , length = days)
> for (l in 1:days){
+   # importing routine
+   if(l < 10){
+     number <- paste("0" , l , sep = "")
+   }else{
+     number <- l
+   }
+   name <- paste("~/Uni/Master/Master Thesis/Coding/twitter_data/ebola/
+                 01.november_2014/ebodf_2014-11-" , number , sep = "")
+   name <- paste(name , ".RData" , sep = "")
+   load(name)
+   # extracted number of tweets per day
+   twpd[l] <- length(ebodf$text)
+   data.frame <- ebodf
+   # building a corpus containing the tweet texts
+   Corpus <- Corpus(VectorSource(data.frame$text))
+   # convert to lower case
+   Corpus <- tm_map(Corpus , tolower)
+   # remove punctuation but not # and @
+   removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+   Corpus <- tm_map(Corpus , removesomepunct)
+   # remove numbers
+   Corpus <- tm_map(Corpus , removeNumbers)
+   # remove URLs
+   removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+   Corpus <- tm_map(Corpus , removeURL)
+   # remove stopwords adjusted where necessary
+   # including the keywords ebola
+   # which  appear in every tweet
+   myStopwords <- c(stopwords("english") , "ebola")
+   Corpus <- tm_map(Corpus , removeWords , myStopwords)
+   # building a term document matrix with TfIdf weighting
+   DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf) ,
+                                                      weighting = weightTfIdf))
+   # remove sparse terms to make the document term matrix more tractable
+   DTMw <- removeSparseTerms(DTMw , 0.995)
+   DTMw <- as.matrix(DTMw)
+   # calculating term frequency
+   termfreq[[l]] <- colSums(as.matrix(DTMw))
+   # sort the terms decreasingly
+   termfreq[[l]] <- sort(termfreq[[l]] , decreasing = TRUE)
+   # calculate relative term frequency
+   reltermfreq[[l]] <- termfreq[[l]]/twpd[l]
+   # collect relative frequency of certain countries
+   countriesday <- cbind(unname(reltermfreq[[l]]["guinea"]) ,
+                           unname(reltermfreq[[l]]["liberia"]))
+   countriesday <- cbind(countriesday , unname(reltermfreq[[l]]["sierra"]))
+   country[l , ] <- countriesday
+ }
> days <- 31
> for (l in 1:days){
+   # importing routine
+   if(l < 10){
+     number <- paste("0" , l , sep = "")
+   }else{
+     number <- l
+   }
+   name <- paste("~/Uni/Master/Master Thesis/Coding/twitter_data/ebola/
+                 02.december_2014/ebodf_2014-12-" , number , sep = "")
+   name <- paste(name , ".RData" , sep = "")
+   load(name)
+   # extracted number of tweets per day
+   twpd[l] <- length(ebodf$text)
+   data.frame <- ebodf
+   # building a corpus containing the tweet texts
+   Corpus <- Corpus(VectorSource(data.frame$text))
+   # convert to lower case
+   Corpus <- tm_map(Corpus , tolower)
+   # remove punctuation but not # and @
+   removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+   Corpus <- tm_map(Corpus , removesomepunct)
+   # remove numbers
+   Corpus <- tm_map(Corpus , removeNumbers)
+   # remove URLs
+   removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+   Corpus <- tm_map(Corpus , removeURL)
+   # remove stopwords adjusted where necessary
+   # including the keywords ebola
+   # which  appear in every tweet
+   myStopwords <- c(stopwords("english") , "ebola")
+   Corpus <- tm_map(Corpus , removeWords , myStopwords)
+   # building a term document matrix with TfIdf weighting
+   DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf) ,
+                                                      weighting = weightTfIdf))
+   # remove sparse terms to make the document term matrix more tractable
+   DTMw <- removeSparseTerms(DTMw , 0.995)
+   DTMw <- as.matrix(DTMw)
+   # calculating term frequency
+   termfreq[[l]] <- colSums(as.matrix(DTMw))
+   # sort the terms decreasingly
+   termfreq[[l]] <- sort(termfreq[[l]] , decreasing = TRUE)
+   # calculate relative term frequency
+   reltermfreq[[l]] <- termfreq[[l]]/twpd[l]
+   # collect relative frequency of certain countries
+   countriesday <- cbind(unname(reltermfreq[[l]]["guinea"]) ,
+                         unname(reltermfreq[[l]]["liberia"]))
+   countriesday <- cbind(countriesday , unname(reltermfreq[[l]]["sierra"]))
+   country[l + 30 , ] <- countriesday
+ }
> for (l in 1:days){
+   # importing routine
+   if(l < 10){
+     number <- paste("0" , l , sep = "")
+   }else{
+     number <- l
+   }
+   name <- paste("~/Uni/Master/Master Thesis/Coding/twitter_data/ebola/
+                 03.january_2015/ebodf_2015-01-" , number , sep = "")
+   name <- paste(name , ".RData" , sep = "")
+   load(name)
+   # extracted number of tweets per day
+   twpd[l] <- length(ebodf$text)
+   data.frame <- ebodf
+   # building a corpus containing the tweet texts
+   Corpus <- Corpus(VectorSource(data.frame$text))
+   # convert to lower case
+   Corpus <- tm_map(Corpus , tolower)
+   # remove punctuation but not # and @
+   removesomepunct <- function(x) gsub("[^[:alnum:][:space:]#@]" , "" , x)
+   Corpus <- tm_map(Corpus , removesomepunct)
+   # remove numbers
+   Corpus <- tm_map(Corpus , removeNumbers)
+   # remove URLs
+   removeURL <- function(x) gsub("http[[:alnum:]]*" , "" , x)
+   Corpus <- tm_map(Corpus , removeURL)
+   # remove stopwords adjusted where necessary
+   # including the keywords ebola
+   # which  appear in every tweet
+   myStopwords <- c(stopwords("english") , "ebola")
+   Corpus <- tm_map(Corpus , removeWords , myStopwords)
+   # building a term document matrix with TfIdf weighting
+   DTMw <- DocumentTermMatrix(Corpus , control = list(wordLengths = c(2 , Inf) ,
+                                                      weighting = weightTfIdf))
+   # remove sparse terms to make the document term matrix more tractable
+   DTMw <- removeSparseTerms(DTMw , 0.995)
+   DTMw <- as.matrix(DTMw)
+   # calculating term frequency
+   termfreq[[l]] <- colSums(as.matrix(DTMw))
+   # sort the terms decreasingly
+   termfreq[[l]] <- sort(termfreq[[l]] , decreasing = TRUE)
+   # calculate relative term frequency
+   reltermfreq[[l]] <- termfreq[[l]]/twpd[l]
+   # collect relative frequency of certain countries
+   countriesday <- cbind(unname(reltermfreq[[l]]["guinea"]) ,
+                         unname(reltermfreq[[l]]["liberia"]))
+   countriesday <- cbind(countriesday , unname(reltermfreq[[l]]["sierra"]))
+   country[l + 61 , ] <- countriesday
+ }
```

```
> # clean data by replacing NA
> country[is.na(country)] <- 0
> # plot 3 countries
> par(mar=c(7,6,4,2) , mgp=c(5,1,0))
> plot(country[ , 1] , xlab = "date" , main = "relative occurrence of countries" ,
+      ylab = "frequency in relation to all tweets" , type = "b" , xaxt = "n" ,
+      ylim = c(0 , max(country)) , pch = 0  ,lwd = 2, cex.lab = 1.2 ,
+      cex.main = 1.5)
> lines(country[ , 2] , col = "red" , type = "b" , pch = 1  ,lwd = 2)
> lines(country[ , 3] , col = "blue" , type = "b" , pch = 2 , lwd = 2)
> axis(side = 1 , at = c(1,5,10,15,20,25,30,35,40,45,50,55,60,66,71,76,81,86,91) ,
+      labels = c("11/1/14" , "11/5/14" , "11/10/14" , "11/15/14" , "11/20/14" ,
+                 "11/25/14" , "11/30/14" , "12/5/14" , "12/10/14" , "12/15/14" ,
+                 "12/20/14" , "12/25/14" , "12/30/14" ,"1/5/15" , "1/10/15" ,
+                 "1/15/15" , "1/20/15" , "1/25/15" , "1/30/15") , las = 2)
> legend("topleft" , cex = 0.3 , legend = colnames(country) , fill =
+        c("black" , "red" , "blue") , bty = "n")
> # read in data from WHO
> ebola.who.death <- read.csv("ebola_who_deaths.csv" , header = TRUE)
> ebola.who.cases <- read.csv("ebola_who_cases.csv" , header = TRUE)
> # take weekly average from twitter ebola data to fit to WHO data
> country.week <- data.frame(matrix(ncol = 3, nrow = 17))
> colnames(country.week) <- c("Guinea" , "Liberia" , "Sierra Leone")
> country.week[1 , ] <- country[1 , ]
> for (i in 1:3){
+   country.week[2 , i] <- sum(country[2:5 , i])/4
+ }
> for (i in 1:3){
+   country.week[3 , i] <- sum(country[6:7 , i])/2
+ }
> for (i in 1:3){
+   country.week[4 , i] <- sum(country[8:12 , i])/5
+ }
> for (i in 1:3){
+   country.week[5 , i] <- sum(country[13:14 , i])/2
+ }
> for (i in 1:3){
+   country.week[6 , i] <- sum(country[15:19 , i])/5
+ }
> for (i in 1:3){
```

```
+   country.week[7 , i] <- sum(country[20:21 , i])/2
+ }
> for (i in 1:3){
+   country.week[8 , i] <- sum(country[22:26 , i])/5
+ }
> for (i in 1:3){
+   country.week[9 , i] <- sum(country[27:33 , i])/7
+ }
> for (i in 1:3){
+   country.week[10 , i] <- sum(country[34:40 , i])/7
+ }
> for (i in 1:3){
+   country.week[11 , i] <- sum(country[41:47 , i])/7
+ }
> for (i in 1:3){
+   country.week[12 , i] <- sum(country[48:54 , i])/7
+ }
> for (i in 1:3){
+   country.week[13 , i] <- sum(country[55:61 , i])/7
+ }
> for (i in 1:3){
+   country.week[14 , i] <- sum(country[62:68 , i])/7
+ }
> for (i in 1:3){
+   country.week[15 , i] <- sum(country[69:75 , i])/7
+ }
> for (i in 1:3){
+   country.week[16 , i] <- sum(country[76:82 , i])/7
+ }
> for (i in 1:3){
+   country.week[17 , i] <- sum(country[83:89 , i])/7
+ }
> # exclude first day
> country.week <- country.week[-1 , ]
> ebola.who.death <- ebola.who.death[-1 , ]
> # plot 3 countries - weekly average
> plot(country.week[ , 1] ,
+      main = "relative occurrence of countries - weekly average" ,
+      ylab = "frequency in relation to all tweets" , type = "b" ,
+      ylim = c(0 , max(country.week)) ,
+      xlab = "week", cex.lab = 1.2 , cex.main = 1.5 , pch = 0)
> lines(country.week[ , 2] , col = "red" , type = "b", pch = 1)
> lines(country.week[ , 3] , col = "blue" , type = "b", pch = 2)
> legend("topleft" , cex = 0.7 , legend = colnames(country) , fill =
+        c("black" , "red" , "blue") , bty = "n")
> # examine correlation between official who death data and weekly
> # average twitter data
> par(mfrow = c(3,1))
> plot(country.week[ , 1] , ebola.who.death[ , 1] , ylab = "WHO Deaths" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Guinea")
> lines(lowess(country.week[ , 1] , ebola.who.death[ , 1])$x ,
+       lowess(country.week[ , 1] , ebola.who.death[ , 1])$y ,
+       col = "red" , lwd = 3)
> plot(country.week[ , 2] , ebola.who.death[ , 2] , ylab = "WHO Deaths" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Liberia")
> lines(lowess(country.week[ , 2] , ebola.who.death[ , 2])$x ,
+       lowess(country.week[ , 2] , ebola.who.death[ , 2])$y ,
+       col = "red" , lwd = 3)
> plot(country.week[ , 3] , ebola.who.death[ , 3] , ylab = "WHO Deaths" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Sierra Leone")
> lines(lowess(country.week[ , 3] , ebola.who.death[ , 3])$x ,
+       lowess(country.week[ , 3] , ebola.who.death[ , 3])$y ,
+       col = "red" , lwd = 3)
> # extend who death data constantly to 92 days
> ebola.who.death.extended <- data.frame(matrix(ncol = 3, nrow = 92))
> colnames(ebola.who.death.extended) <- c("Guinea" , "Liberia" , "Sierra Leone")
> ebola.who.death.extended[1:4 , ] <- c(0,0,0)
> for (i in 1:3){
+   ebola.who.death.extended[5:6 , i] <- ebola.who.death[1 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[7:11 , i] <- ebola.who.death[2 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[12:13 , i] <- ebola.who.death[3 , i]
+ }
> for (i in 1:3){
```

```
+   ebola.who.death.extended[14:18 , i] <- ebola.who.death[4 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[19:20 , i] <- ebola.who.death[5 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[21:25 , i] <- ebola.who.death[6 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[26:32 , i] <- ebola.who.death[7 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[33:39 , i] <- ebola.who.death[8 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[40:46 , i] <- ebola.who.death[9 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[47:53 , i] <- ebola.who.death[10 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[54:60 , i] <- ebola.who.death[11 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[61:67 , i] <- ebola.who.death[12 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[68:74 , i] <- ebola.who.death[13 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[75:81 , i] <- ebola.who.death[14 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[82:88 , i] <- ebola.who.death[15 , i]
+ }
> for (i in 1:3){
+   ebola.who.death.extended[89:92 , i] <- ebola.who.death[16 , i]
+ }
> # examine correlation between extended who death data and daily twitter data
> par(mfrow = c(3,1))
> plot(country[ , 1] , ebola.who.death.extended[ , 1] , ylab = "WHO Deaths" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Guinea")
> lines(lowess(country[ , 1] , ebola.who.death.extended[ , 1])$x ,
+       lowess(country[ , 1] , ebola.who.death.extended[ , 1])$y ,
+       col = "red" , lwd = 3)
> plot(country[ , 2] , ebola.who.death.extended[ , 2] , ylab = "WHO Deaths" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Liberia")
> lines(lowess(country[ , 2] , ebola.who.death.extended[ , 2])$x ,
+       lowess(country[ , 2] , ebola.who.death.extended[ , 2])$y ,
+       col = "red" , lwd = 3)
> plot(country[ , 3] , ebola.who.death.extended[ , 3] , ylab = "WHO Deaths" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Sierra Leone")
> lines(lowess(country[ , 3] , ebola.who.death.extended[ , 3])$x ,
+       lowess(country[ , 3] , ebola.who.death.extended[ , 3])$y ,
+       col = "red" , lwd = 3)
> # examine correlation between official  who cases data and weekly
> # average twitter data
> ebola.who.cases <- ebola.who.cases[-1 , ]
> par(mfrow = c(3,1))
> plot(country.week[ , 1] , ebola.who.cases[ , 1] , ylab = "WHO Cases" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Guinea")
> lines(lowess(country.week[ , 1] , ebola.who.cases[ , 1])$x ,
+       lowess(country.week[ , 1] , ebola.who.cases[ , 1])$y ,
+       col = "red" , lwd = 3)
> plot(country.week[ , 2] , ebola.who.cases[ , 2] , ylab = "WHO Cases" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Liberia")
> lines(lowess(country.week[ , 2] , ebola.who.cases[ , 2])$x ,
+       lowess(country.week[ , 2] , ebola.who.cases[ , 2])$y ,
+       col = "red" , lwd = 3)
> plot(country.week[ , 3] , ebola.who.cases[ , 3] , ylab = "WHO Cases" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Sierra Leone")
> lines(lowess(country.week[ , 3] , ebola.who.cases[ , 3])$x ,
+       lowess(country.week[ , 3] , ebola.who.cases[ , 3])$y ,
+       col = "red" , lwd = 3)
```

```
> # extend who cases data constantly to 92 days
> ebola.who.cases.extended <- data.frame(matrix(ncol = 3, nrow = 92))
> colnames(ebola.who.cases.extended) <- c("Guinea" , "Liberia" , "Sierra Leone")
> ebola.who.cases.extended[1:4 , ] <- c(0,0,0)
> for (i in 1:3){
+   ebola.who.cases.extended[5:6 , i] <- ebola.who.cases[1 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[7:11 , i] <- ebola.who.cases[2 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[12:13 , i] <- ebola.who.cases[3 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[14:18 , i] <- ebola.who.cases[4 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[19:20 , i] <- ebola.who.cases[5 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[21:25 , i] <- ebola.who.cases[6 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[26:32 , i] <- ebola.who.cases[7 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[33:39 , i] <- ebola.who.cases[8 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[40:46 , i] <- ebola.who.cases[9 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[47:53 , i] <- ebola.who.cases[10 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[54:60 , i] <- ebola.who.cases[11 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[61:67 , i] <- ebola.who.cases[12 , i]
+ }
```

```
> for (i in 1:3){
+   ebola.who.cases.extended[68:74 , i] <- ebola.who.cases[13 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[75:81 , i] <- ebola.who.cases[14 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[82:88 , i] <- ebola.who.cases[15 , i]
+ }
> for (i in 1:3){
+   ebola.who.cases.extended[89:92 , i] <- ebola.who.cases[16 , i]
+ }
> # examine correlation between extended who cases data and daily twitter data
> par(mfrow = c(3,1))
> plot(country[ , 1] , ebola.who.cases.extended[ , 1] , ylab = "WHO Cases" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Guinea")
> lines(lowess(country[ , 1] , ebola.who.cases.extended[ , 1])$x ,
+       lowess(country[ , 1] , ebola.who.cases.extended[ , 1])$y ,
+       col = "red" , lwd = 3)
> plot(country[ , 2] , ebola.who.cases.extended[ , 2] , ylab = "WHO Cases" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Liberia")
> lines(lowess(country[ , 2] , ebola.who.cases.extended[ , 2])$x ,
+       lowess(country[ , 2] , ebola.who.cases.extended[ , 2])$y ,
+       col = "red" , lwd = 3)
> plot(country[ , 3] , ebola.who.cases.extended[ , 3] , ylab = "WHO Cases" ,
+      xlab = "Relative Appearance on Twitter" ,
+      main = "Relation between WHO and Twitter data - Sierra Leone")
> lines(lowess(country[ , 3] , ebola.who.cases.extended[ , 3])$x ,
+       lowess(country[ , 3] , ebola.who.cases.extended[ , 3])$y ,
+       col = "red" , lwd = 3)
> lmguineadeath <- lm(formula = ebola.who.death.extended[,1] ~ country[,1])
> summary.lm(lmguineadeath)

Call:
lm(formula = ebola.who.death.extended[, 1] ~ country[, 1])

Residuals:
    Min      1Q  Median      3Q     Max 
-61.525 -26.987   5.045  25.307  49.573 
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    62.951      4.764  13.213   <2e-16 ***
country[, 1] -927.960    645.650  -1.437    0.154
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 31.79 on 90 degrees of freedom
Multiple R-squared:  0.02244,        Adjusted R-squared:  0.01158
F-statistic: 2.066 on 1 and 90 DF,  p-value: 0.1541

> lmliberiadeath <- lm(formula = ebola.who.death.extended[,2] ~ country[,2])
> summary.lm(lmliberiadeath)

Call:
lm(formula = ebola.who.death.extended[, 2] ~ country[, 2])

Residuals:
    Min      1Q  Median      3Q     Max
-98.352 -35.582  -5.315  35.893 222.566

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    45.911      9.979   4.601 1.37e-05 ***
country[, 2] 1585.680    537.868   2.948  0.00407 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 55.64 on 90 degrees of freedom
Multiple R-squared:  0.08806,        Adjusted R-squared:  0.07793
F-statistic: 8.691 on 1 and 90 DF,  p-value: 0.004073

> lmsierraleonedeath <- lm(formula = ebola.who.death.extended[,3] ~ country[,3])
> summary.lm(lmsierraleonedeath)

Call:
lm(formula = ebola.who.death.extended[, 3] ~ country[, 3])

Residuals:
    Min      1Q  Median      3Q     Max
```

```
-206.26  -82.88  -23.52   36.77  384.48

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)     97.62      26.01   3.753 0.000309 ***
country[, 3]  2839.51    1157.77   2.453 0.016113 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 126.5 on 90 degrees of freedom
Multiple R-squared:  0.06265,        Adjusted R-squared:  0.05223
F-statistic: 6.015 on 1 and 90 DF,  p-value: 0.01611

> lmguineacase <- lm(formula = ebola.who.cases.extended[,1] ~ country[,1])
> summary.lm(lmguineacase)

Call:
lm(formula = ebola.who.cases.extended[, 1] ~ country[, 1])

Residuals:
   Min     1Q Median     3Q    Max
-81.57 -38.78 -10.13  36.30 104.40

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)     83.14       7.02  11.844   <2e-16 ***
country[, 1] -1023.27     951.33  -1.076    0.285
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 46.84 on 90 degrees of freedom
Multiple R-squared:  0.01269,        Adjusted R-squared:  0.001722
F-statistic: 1.157 on 1 and 90 DF,  p-value: 0.285

> lmliberiacase <- lm(formula = ebola.who.cases.extended[,2] ~ country[,2])
> summary.lm(lmliberiacase)

Call:
lm(formula = ebola.who.cases.extended[, 2] ~ country[, 2])

Residuals:
    Min      1Q  Median      3Q     Max
-135.40  -54.89  -27.58   26.82  349.19

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    106.24      19.73   5.385 5.73e-07 ***
country[, 2]  1567.82    1063.41   1.474    0.144
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 110 on 90 degrees of freedom
Multiple R-squared:  0.02358,        Adjusted R-squared:  0.01273
F-statistic: 2.174 on 1 and 90 DF,  p-value: 0.1439

> lmsierraleonecase <- lm(formula = ebola.who.cases.extended[,3] ~ country[,3])
> summary.lm(lmsierraleonecase)

Call:
lm(formula = ebola.who.cases.extended[, 3] ~ country[, 3])

Residuals:
   Min     1Q Median     3Q    Max
-395.1 -153.4   18.8  119.3  400.7

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    249.07      39.61   6.288 1.13e-08 ***
country[, 3]  6186.52    1762.84   3.509 0.000704 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 192.6 on 90 degrees of freedom
Multiple R-squared:  0.1204,        Adjusted R-squared:  0.1106
F-statistic: 12.32 on 1 and 90 DF,  p-value: 0.0007036
```

Appendix C

Plots

C.1 Development of most frequent terms in the tweets of November 2014

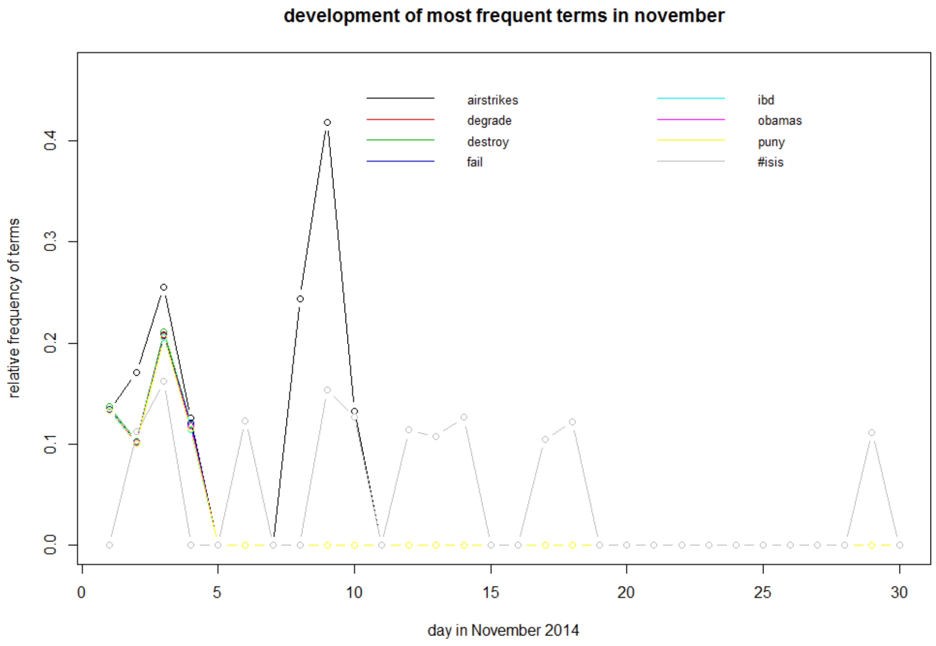


Figure C.1: Development of several top terms of Islamic State tweets in November 2014 I

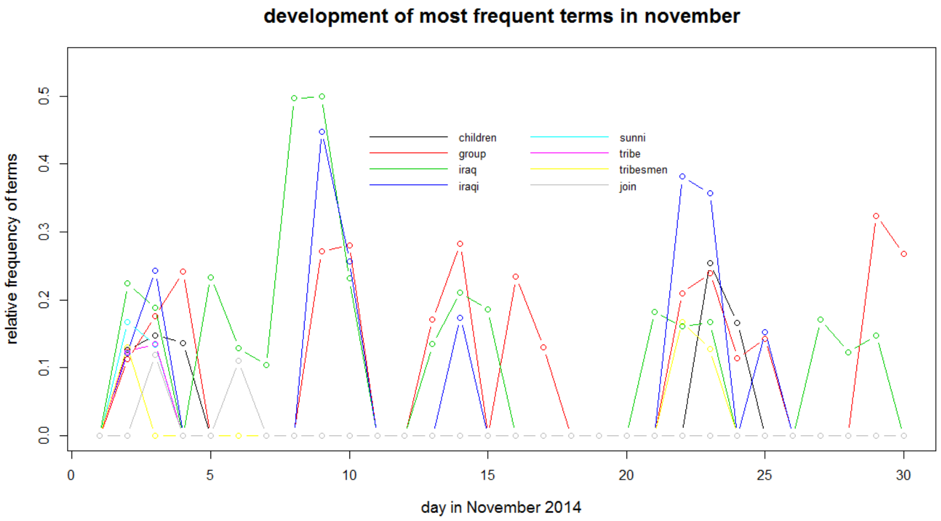


Figure C.2: Development of several top terms of Islamic State tweets in November 2014 II

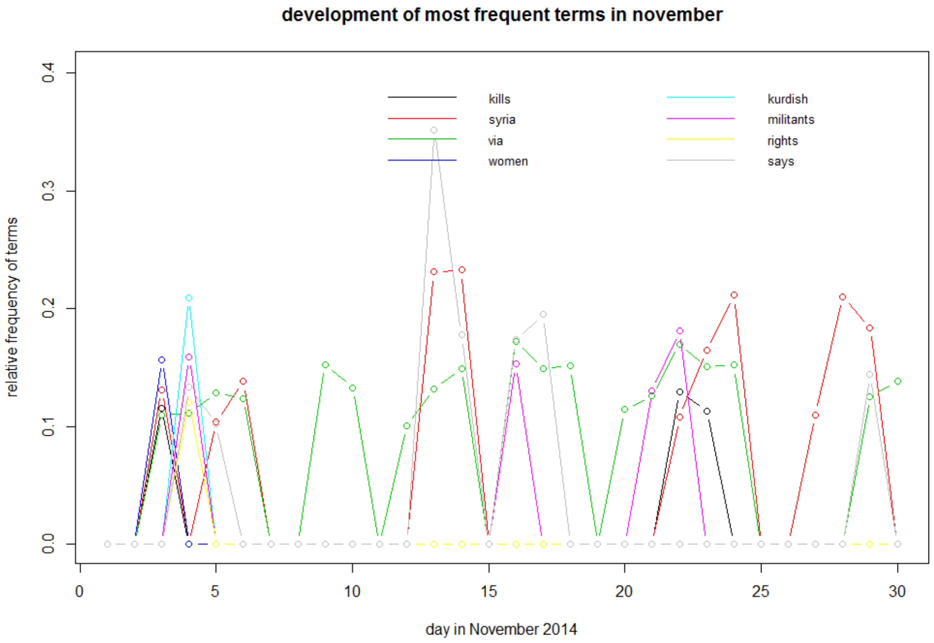


Figure C.3: Development of several top terms of Islamic State tweets in November 2014 III

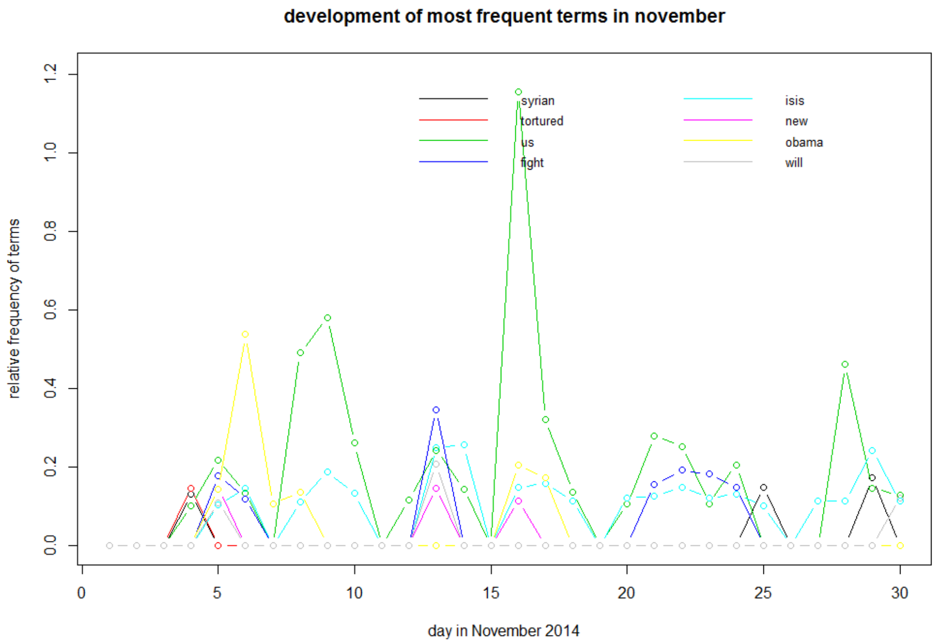


Figure C.4: Development of several top terms of Islamic State tweets in November 2014 IV

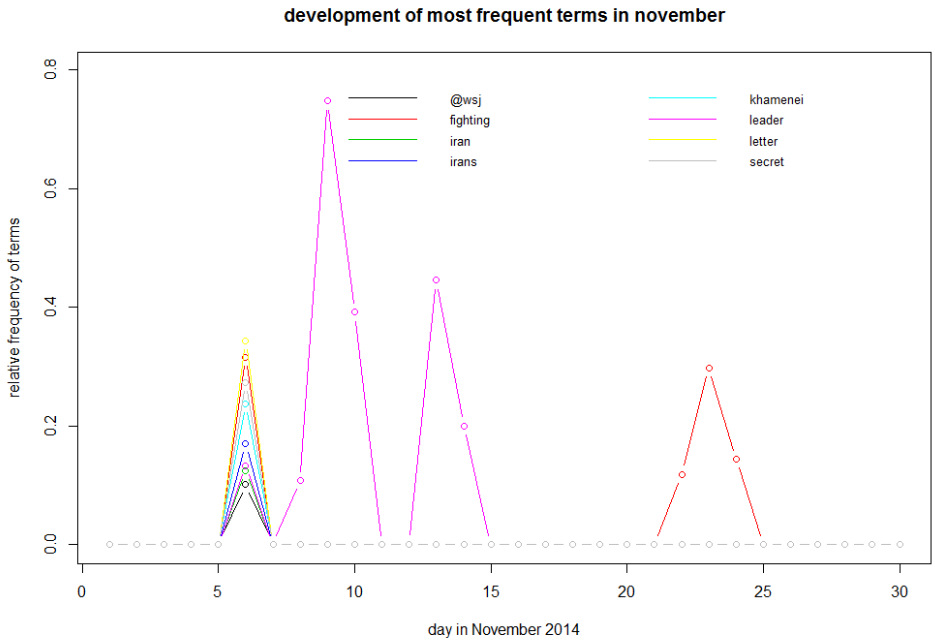


Figure C.5: Development of several top terms of Islamic State tweets in November 2014 V

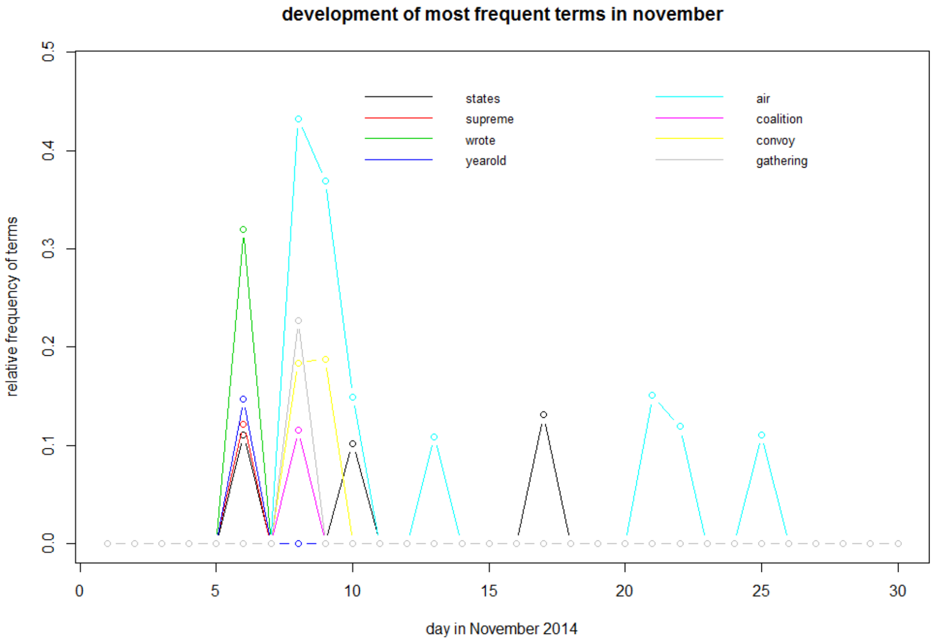


Figure C.6: Development of several top terms of Islamic State tweets in November 2014 VI

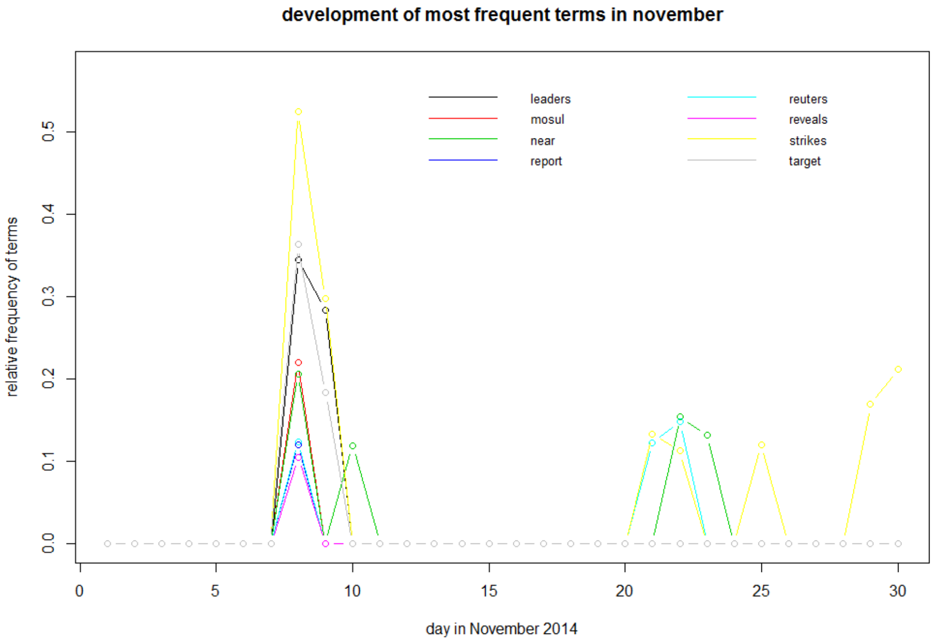


Figure C.7: Development of several top terms of Islamic State tweets in November 2014 VII

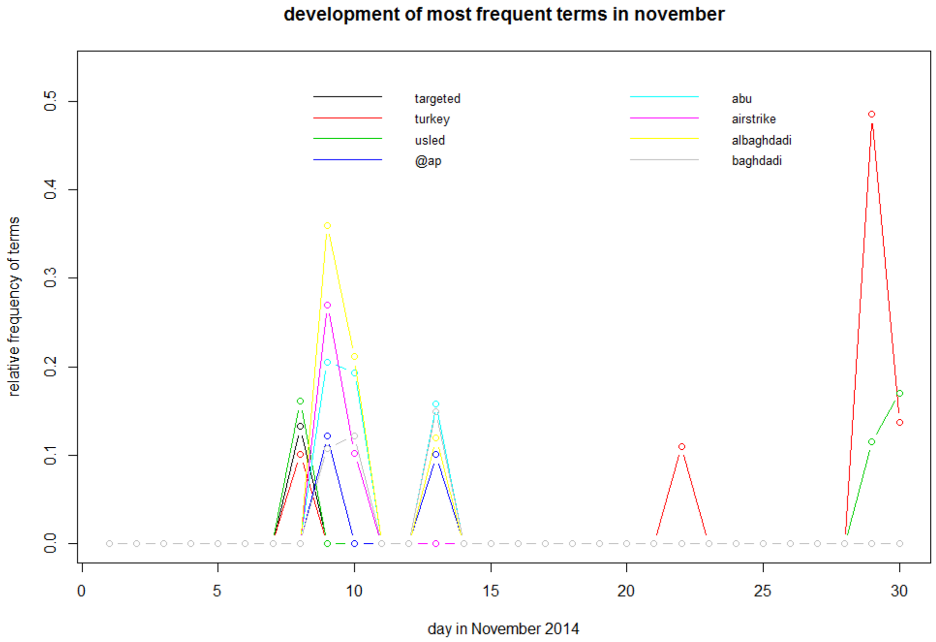


Figure C.8: Development of several top terms of Islamic State tweets in November 2014 VIII

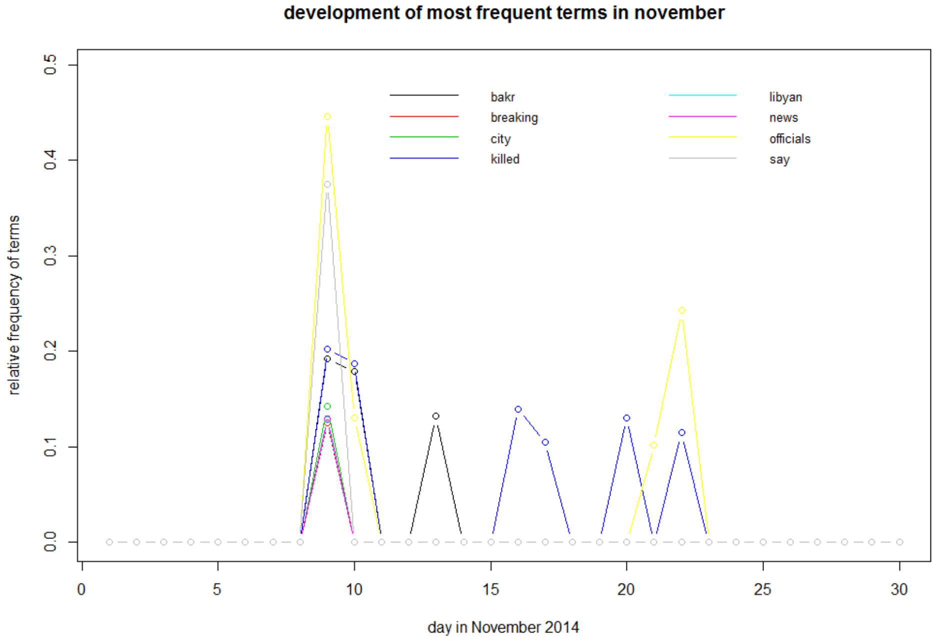


Figure C.9: Development of several top terms of Islamic State tweets in November 2014 IX

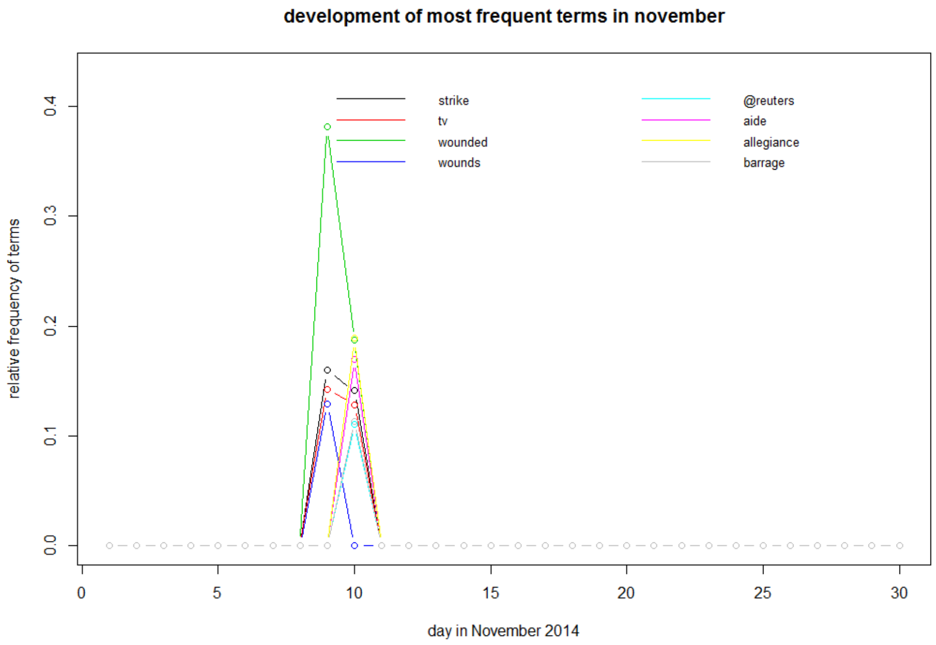


Figure C.10: Development of several top terms of Islamic State tweets in November 2014 X

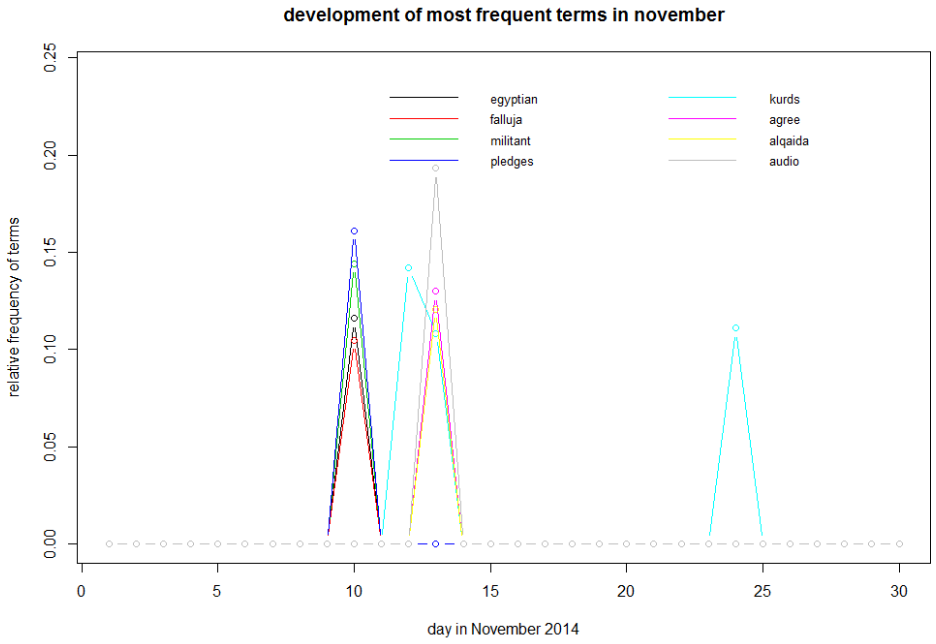


Figure C.11: Development of several top terms of Islamic State tweets in November 2014 XI

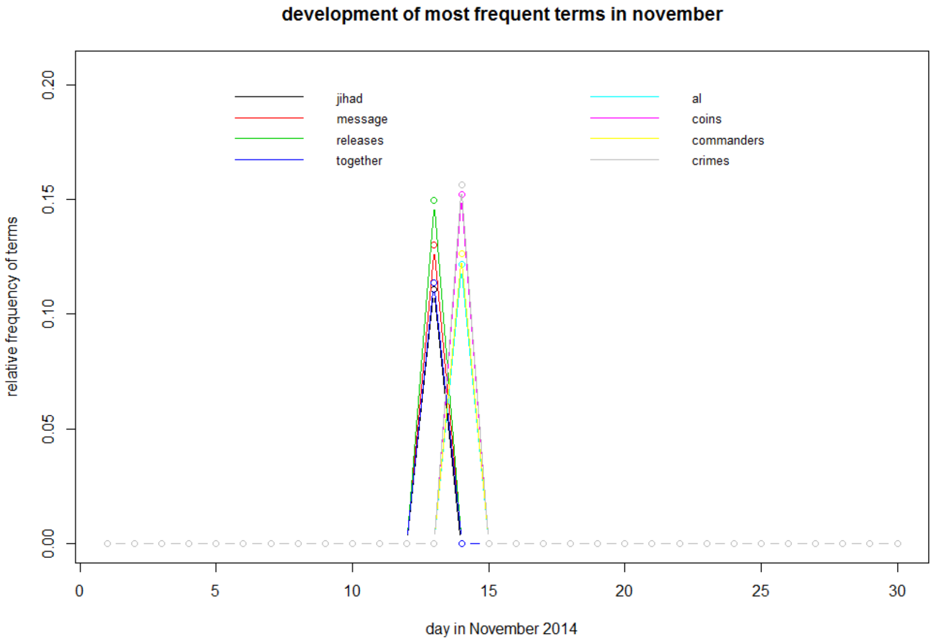


Figure C.12: Development of several top terms of Islamic State tweets in November 2014 XII

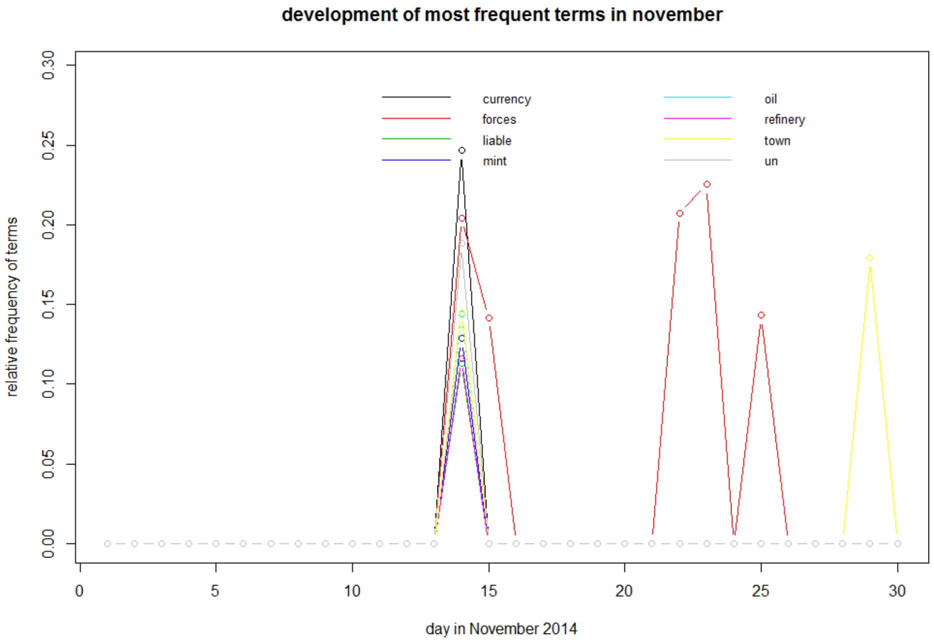


Figure C.13: Development of several top terms of Islamic State tweets in November 2014 XIII

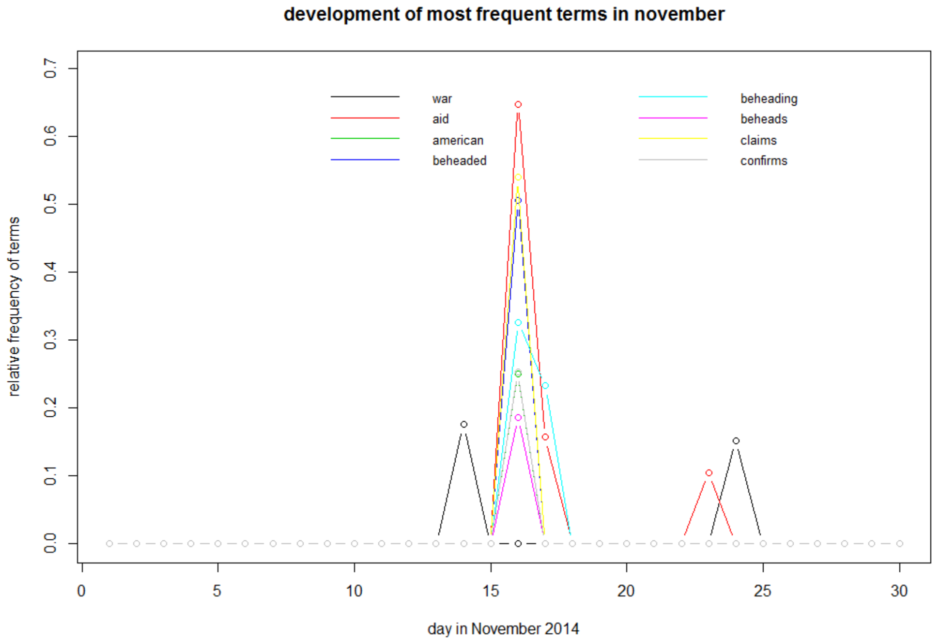


Figure C.14: Development of several top terms of Islamic State tweets in November 2014 XIV

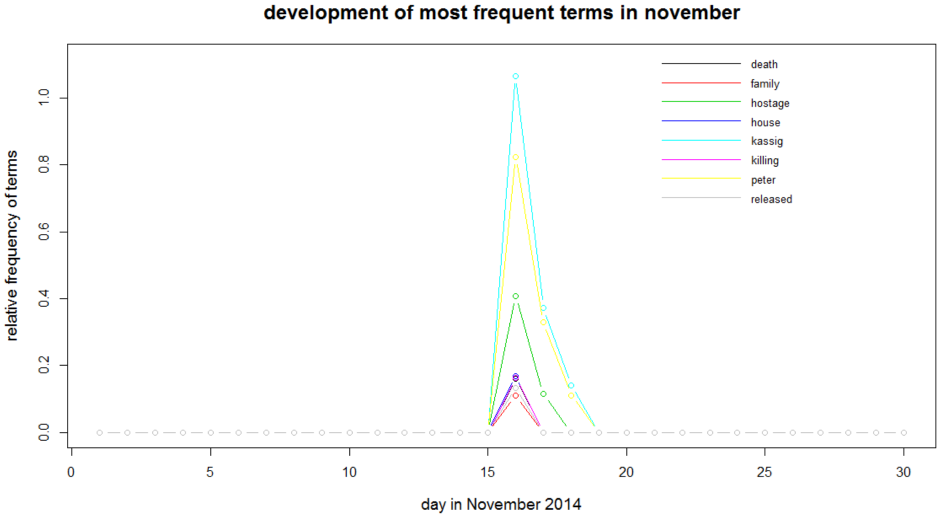


Figure C.15: Development of several top terms of Islamic State tweets in November 2014 XV

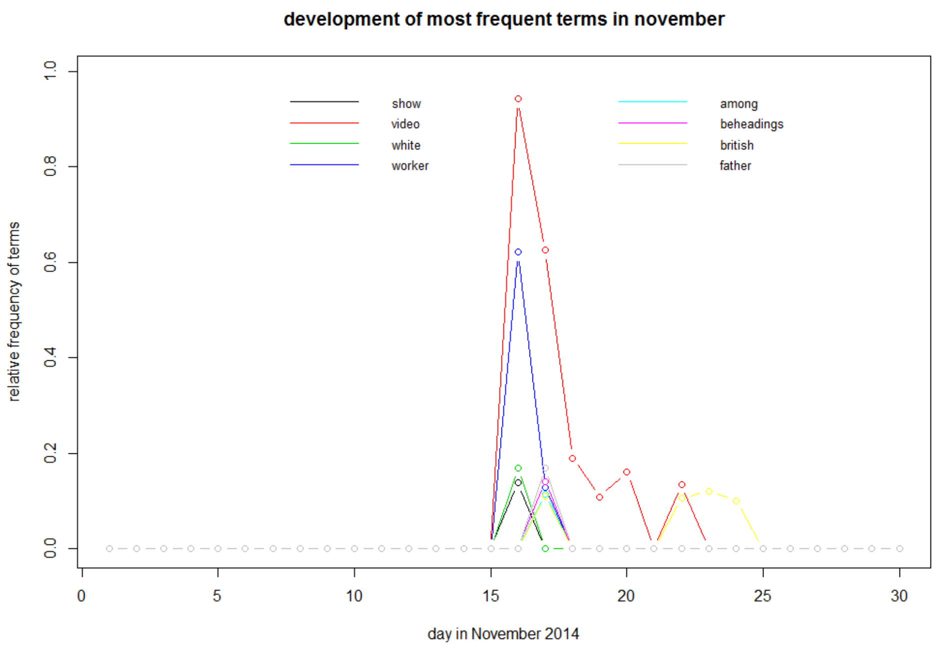


Figure C.16: Development of several top terms of Islamic State tweets in November 2014 XVI

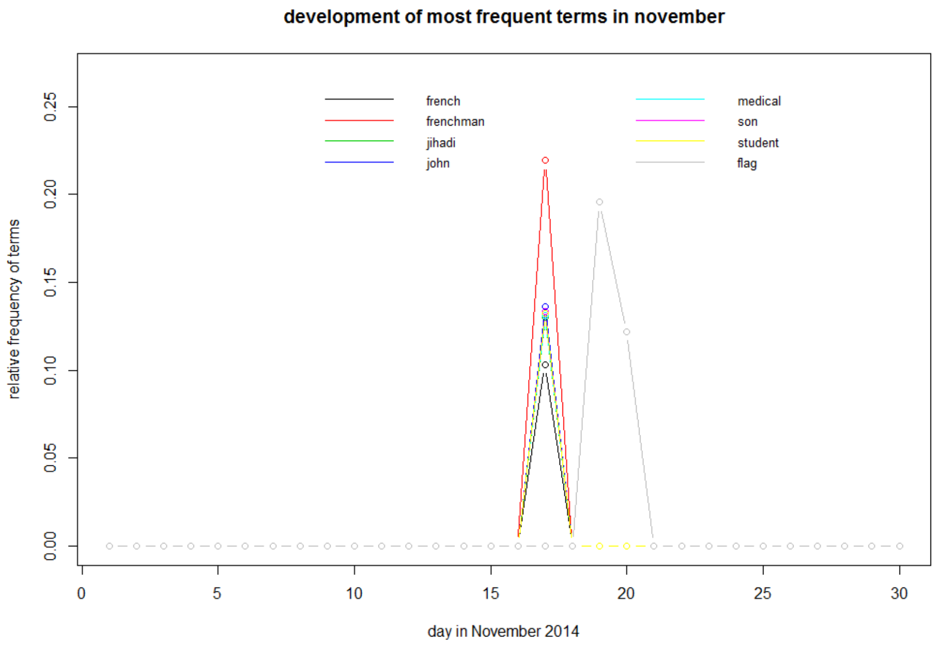


Figure C.17: Development of several top terms of Islamic State tweets in November 2014 XVII

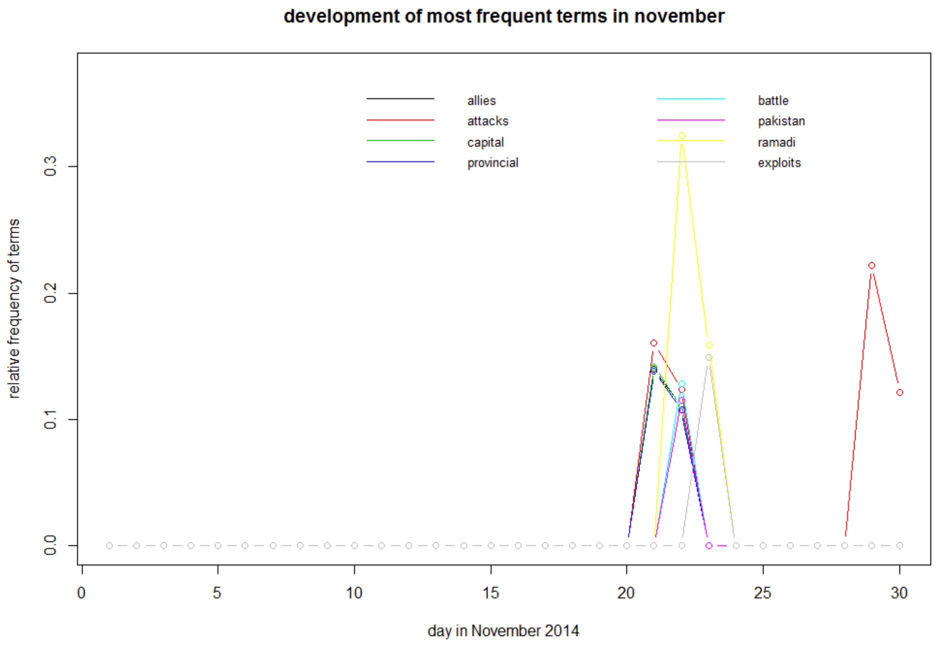


Figure C.18: Development of several top terms of Islamic State tweets in November 2014 XVIII

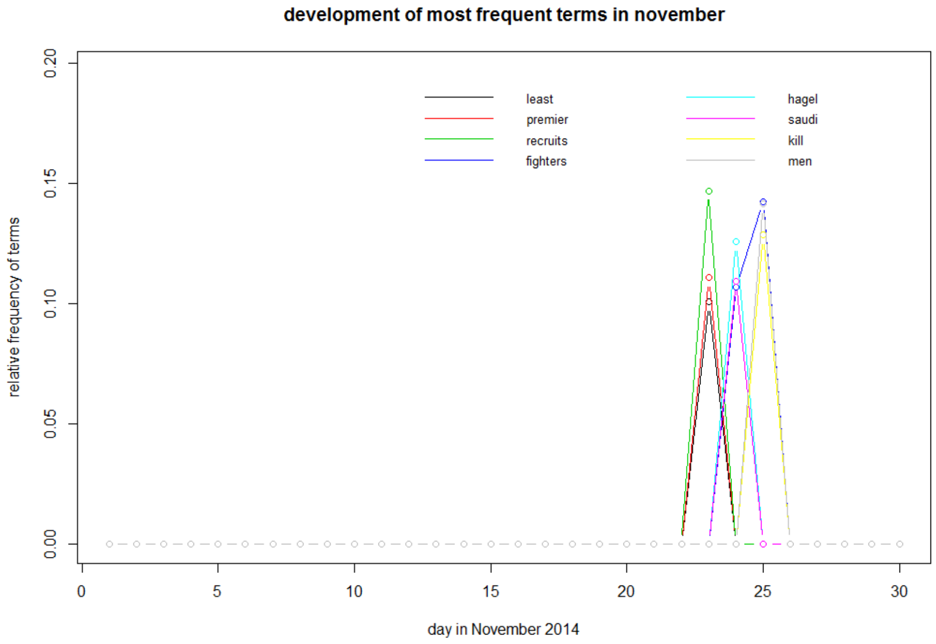


Figure C.19: Development of several top terms of Islamic State tweets in November 2014 XIX

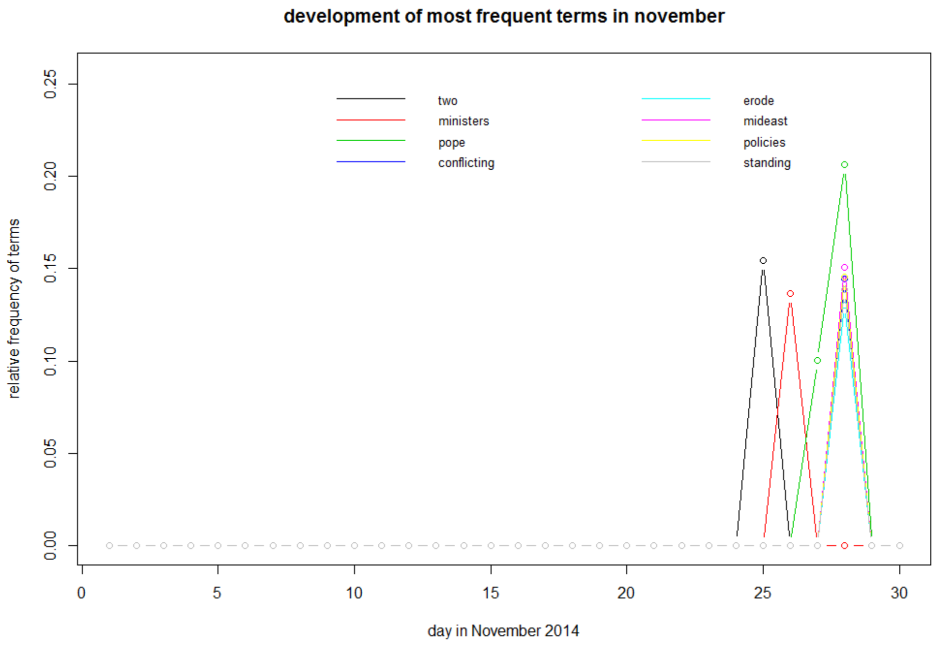


Figure C.20: Development of several top terms of Islamic State tweets in November 2014 XX

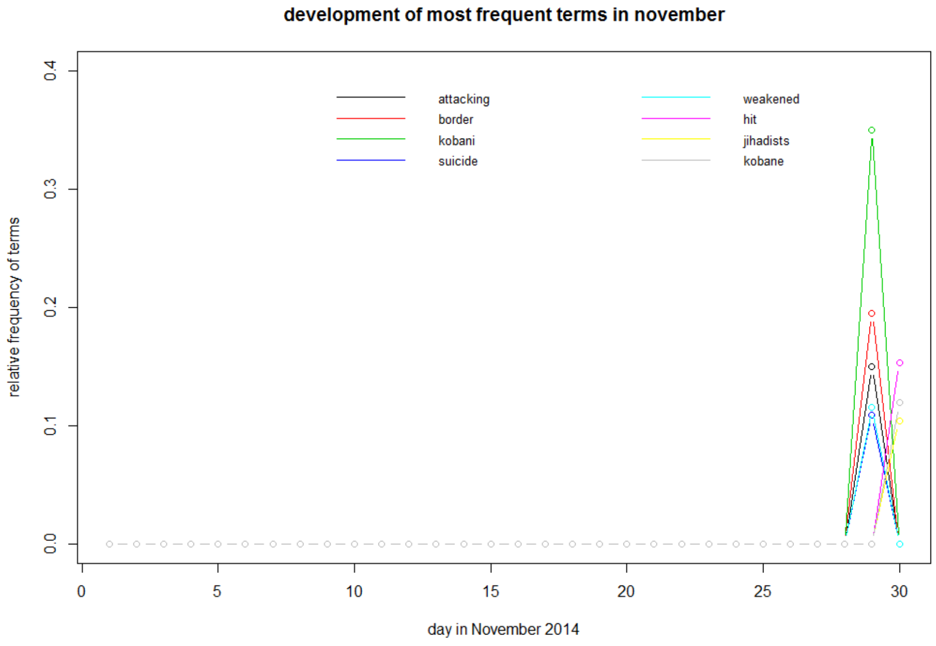


Figure C.21: Development of several top terms of Islamic State tweets in November 2014 XXI

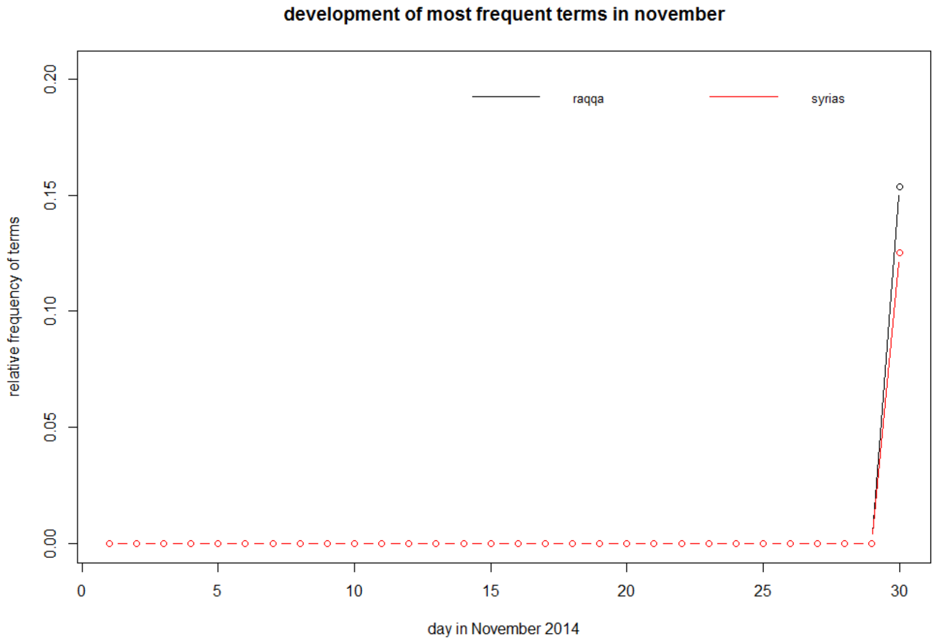


Figure C.22: Development of several top terms of Islamic State tweets in November 2014 XXII

Bibliography

[1] **Matthew Wall (2014):**
Big Data: Are you ready for blast-off?
http://www.bbc.com/news/business-26383058

[2] **Twitter - Wikipedia:**
http://en.wikipedia.org/wiki/Twitter

[3] **Twitter and status updating:**
http://www.pewinternet.org/2009/02/12/twitter-and-status-updating/

[4] **Twitter Report Second Quarter 2014 - Results:**
https://investor.twitterinc.com/releasedetail.cfm?ReleaseID=862505

[5] **Twitter stock - registration statement:**
http://www.sec.gov/Archives/edgar/data/1418091/000119312513390321/d564001ds1.htm#toc564001_1

[6] **The Engineering Behind TwitterâĂŹs New Search Experience:**
https://blog.twitter.com/2011/engineering-behind-twitter%E2%80%99s-new-search-experience

[7] **Twitter Figures - Alexa:**
http://www.alexa.com/siteinfo/twitter.com

[8] **Market value of the largest internet companies worldwide - Statista:**
http://www.statista.com/statistics/277483/market-value-of-the-largest-internet-companies-worldwide/

[9] **Most retweeted tweet by Ellen DEGeneres:**
https://twitter.com/theellenshow/status/440322224407314432

[10] **OAuth - Overview:**
https://dev.twitter.com/oauth/overview

[11] **Twitter - Terms of service:**
https://twitter.com/tos

[12] **Partial correlation coefficient:**
http://www.encyclopediaofmath.org/index.php?title=Partial_correlation_coefficient

&oldid=24254
Encyclopedia of Mathematics

[13] **W. Ludwig-Mayerhofer (2004): Faktorenanalyse**
http://psydok.sulb.uni-saarland.de/volltexte/2004/260/html/ilm_f3.htm
ILMES - Internet-Lexikon der Methoden der empirischen Sozialforschung

[14] **Obama calls hostage's beheading by ISIS 'pure evil'**
http://edition.cnn.com/2014/11/16/world/meast/isis-kassig-killing-images/

[15] **Peter Kassig killed by ISIS, Obama confirms**
http://www.cbsnews.com/news/isis-video-claims-us-aid-worker-peter-kassig-beheaded-in-syria/

[16] **US air strikes near Mosul destroy Isis convoy**
http://www.theguardian.com/world/2014/nov/08/us-air-strikes-target-top-isis-leaders-in-iraq

[17] **Sweden betrays Islamic State victims**
http://www.yourmiddleeast.com/opinion/sweden-betrays-islamic-state-victims_27976

[18] **Kassig family responds to Islamic State beheading video**
http://mashable.com/2014/11/16/kassig-family-islamic-state/

[19] **Offical website of WHO:**
http://apps.who.int/ebola/en/current-situation/ebola-situation-report

[nov3] **Death toll from ISIS' public executions of Iraqi Sunni tribesmen passes 200**
http://www.cbsnews.com/news/death-toll-from-isis-public-executions-of-iraqi-sunni-tribesmen-passes-200/

[nov4] **Islamic State Tortured Kurdish Child Hostages, Says Rights Group**
http://www.newsweek.com/islamic-state-tortured-kurdish-child-hostages-rights-group-281957

[nov5] **New Zealand won't send troops to fight Islamic State in Iraq**
http://www.haaretz.com/news/middle-east/middle-east-updates/1.624751

[nov6] **Obama Wrote Secret Letter to IranâĂŹs Khamenei About Fighting Islamic State**
http://www.wsj.com/articles/obama-wrote-secret-letter-to-irans-khamenei-about-fighting-islamic-state-1415295291

[nov8] **US air strikes near Mosul destroy Isis convoy**
http://www.theguardian.com/world/2014/nov/08/us-air-strikes-target-top-isis-leaders-in-iraq

[nov9] **ISIS leader Abu Bakr al-Baghdadi wounded in airstrike, Iraq officials say**
http://www.cbc.ca/news/world/isis-leader-abu-bakr-al-baghdadi-wounded-in-airstrike-iraq-officials-say-1.2829360

[nov10] **Aide to Islamic State's Baghdadi killed near Falluja: Iraqi TV**
http://www.reuters.com/article/2014/11/11/us-mideast-crisis-iraq-idUSKCN0IU0V720141111

[nov12] **Kurds block an Islamic State supply route to Syria's Kobani**
http://news.yahoo.com/kurds-block-islamic-state-supply-route-syrias-kobani-131433506.html

[nov13] **Back from the dead: Audio message of ISIS leader ordering 'volcano of jihad' released by terror group days after reports he was killed in American airstrike on Iraq**
http://www.dailymail.co.uk/news/article-2833437/Days-rumoured-killed-audio-message-said-ISIS-s-leader-ordering-volcano-jihad-released-terror-group.html

[nov14] **UN: Islamic State Liable for 'Massive' War Crimes**
http://www.voanews.com/content/isis-committing-mass-war-crimes-un/2520861.html

[nov16] **Peter Kassig killed by ISIS, Obama confirms**
http://www.cbsnews.com/news/isis-video-claims-us-aid-worker-peter-kassig-beheaded-in-syria/

[nov17] **Frenchman seen in Islamic State video of beheadings**
http://www.dailymail.co.uk/wires/reuters/article-2837770/Briton-Frenchman-seen-Islamic-State-video-beheadings.html

[nov18] **Frenchman, Brit among ISIS thugs seen on tape of Peter Kassig beheading**
http://www.nydailynews.com/news/world/frenchman-brit-isis-thugs-peter-kassig-behead-tape-article-1.2013430

[nov20] **ISIS claims suicide bombings in IraqâĂŹs Erbil**
http://english.alarabiya.net/en/News/middle-east/2014/11/21/ISIS-claims-suicide-bombings-in-Iraq-s-Erbil-.html

[nov21] **Islamic State attacks Iraq provincial capital**
http://www.themalaysianinsider.com/world/article/islamic-state-attacks-iraq-provincial-capital

[nov22] **Islamic State kills at least 25 Iraqi tribesmen near Ramadi: officials**
http://www.reuters.com/article/2014/11/22/us-mideast-crisis-ramadi-idUSKCN0J609O20141122

[nov23] **Islamic State Group Recruits, Exploits Children**
http://www.onebeak.com/world/2014/11/23/islamic-state-group-militants-recruit-exploit-children-in_s_148809289.html

[nov24] **Saudi Arabia says attackers behind Shiite killings linked to IS**
http://jordantimes.com/saudi-arabia-says-attackers-behind-shiite-killings-linked-to-is

[nov25] **Syrian government air strikes kill 63 in Raqqa - monitoring group**
http://uk.reuters.com/article/2014/11/25/uk-mideast-crisis-strikes-idUKKCN0J928X20141125

[nov27] **Middle East Updates / Explosion and gunfire heard in Kabul's diplomatic quarter**
http://www.haaretz.com/news/middle-east/middle-east-updates/1.628731

[nov28] **Fifteen Islamic State targets hit by U.S., allies since Wednesday: U.S.**
http://www.reuters.com/article/2014/11/28/us-mideast-crisis-usa-airstrikes-idUSKCN0JC1G420141128

[nov29] **ISIS attacks border town Kobani from Turkey**
http://www.cbc.ca/news/world/isis-attacks-border-town-kobani-from-turkey-1.2854694

[nov30] **Thirty U.S.-led strikes hit Islamic State in Syria's Raqqa: monitoring group**
http://www.reuters.com/article/2014/12/01/us-mideast-crisis-syria-raqqa-idUSKCN0JE07020141201

[Generous (2014)] **Nicholas Generous, Geoffrey Fairchild, Alina Deshpande, Sara Y. Del Valle, Reid Priedhorsky (2014):**
Global Disease Monitoring and Forecasting with Wikipedia
PLoS Comput Biol 10(11)

[Paul (2014)] **Michael J. Paul and Mark Dredze (2011):**
You Are What You Tweet: Analyzing Twitter for Public Health
Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media

[Gentry (2013)] **Jeff Gentry (2013):**
twitteR: R based Twitter client. R package version 1.1.7.
http://CRAN.R-project.org/package=twitteR

[Feinerer (2014)] **Ingo Feinerer and Kurt Hornik (2014):**
tm: Text Mining Package. R package version 0.5-10
http://CRAN.R-project.org/package=tm
Ingo Feinerer, Kurt Hornik, and David Meyer (2008):
Text Mining Infrastructure in R

Journal of Statistical Software 25(5): 1-54
URL: http://www.jstatsoft.org/v25/i05/

[Lubbe (1997)] **Jan C. A. Van der Lubbe (1997):**
Information Theory
Cambridge University Press

[Klenke (2006)] **Achim Klenke (2006):**
Wahrscheinlichkeitstheorie
Springer Verlag Berlin Heidelberg

[Bai (2005)] **Jushan Bai & Serena Ng (2005):**
Tests for Skewness, Kurtosis, and Normality for Time Series Data
Journal of Business & Economic Statistics, January 2005

[Mardia (1974)] **K.V. Mardia (1974):**
Applications of some measures of multivariate skewness and kurtosis in testing normality and robustness studies
Sankhya: The Indian Journal of Statistics, Volume 36, Series B, Pt. 2, pp. 115-128

[Mardia (1970)] **K.V. Mardia (1970):**
Measures of multivariate skewness and kurtosis with applications
Biometrika 57 (3): p. 519-530

[Axler (1997)] **S. Axler (1997):**
Linear Algebra Done Right
Second Edition, Springer Verlag Acknowledgments (Sonntag)

[Congfeng] **Liu Congfeng:**
Estimation of Random Variables
Xidian University

[Fahrmeir (1996)] **Ludwig Fahrmeir, Alfred Hamerle and Gerhard Tutz (1996):**
Multivariate statistische Verfahren
Walter de Gruyter

[Lilliefors (1967)] **Hubert W. Lilliefors (1967):**
On the Kolmogorov-Smirnov Test for Normality with Mean and Variance Unknown
Journal of the American Statistical Association, Volume 62, Issue 318, 1967

[Cochran (1989)] **Snedecor, George W. and Cochran, William G. (1989):**
Statistical Methods
Eighth Edition, Iowa State University Press

[Eichler (2007)] **M. Eichler (2007):**
Graphical modelling in multivariate statistics
Lecture Note

[Cureton (1983)] **E.E. Cureton & R.B. D'Agostino(1983):**
Factor analysis: an applied approach
Hillside, NJ: Lawrence Erlbaum Associates

[MacKay (2003)] **David J.C. MacKay (2003):**
Information Theory, Inference and Learning Algorithms
Cambridge University Press

[Park (2009)] **Hae-Sang Park, Chi-Hyuck Jun (2009):**
A simple and fast algorithm for K-medoids clustering
Elsevier

[Rousseeuw (1987)] **Peter J. Rousseeuw (1987):**
Silhouettes: A graphical aid to the interpretation and validation of cluster analysis
Computational and Applied Mathematics, Volume 20, pp. 53-65