Text mining ChatGPT

Abishan, Josua, Lars, Luca

2023-03-27

##Introduction Beschriebung einfügen über Projekt. ## Load packages and data

```
library (syuzhet)
```

```
## Warning: Paket 'syuzhet' wurde unter R Version 4.2.3 erstellt
```

```
library (stringr)
library (tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   1.0.0
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v forcats 0.5.2
## v readr   2.1.3
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library (ggplot2)
library(scales)
```

```
##
## Attache Paket: 'scales'
##
## Das folgende Objekt ist maskiert 'package:purrr':
##
##     discard
##
## Das folgende Objekt ist maskiert 'package:readr':
##
##     col_factor
##
## Das folgende Objekt ist maskiert 'package:syuzhet':
##
##     rescale
```

```
library(stringi)
library(lubridate)
```

```
## Lade nötiges Paket: timechange
##
## Attache Paket: 'lubridate'
##
## Die folgenden Objekte sind maskiert von 'package:base':
##
```

```
##     date, intersect, setdiff, union
```

```
library(dplyr)
options(scipen=999)

load("ChatGPT.rda")
```

1. Question: What can you tell us about the users that tweet about ChatGPT?

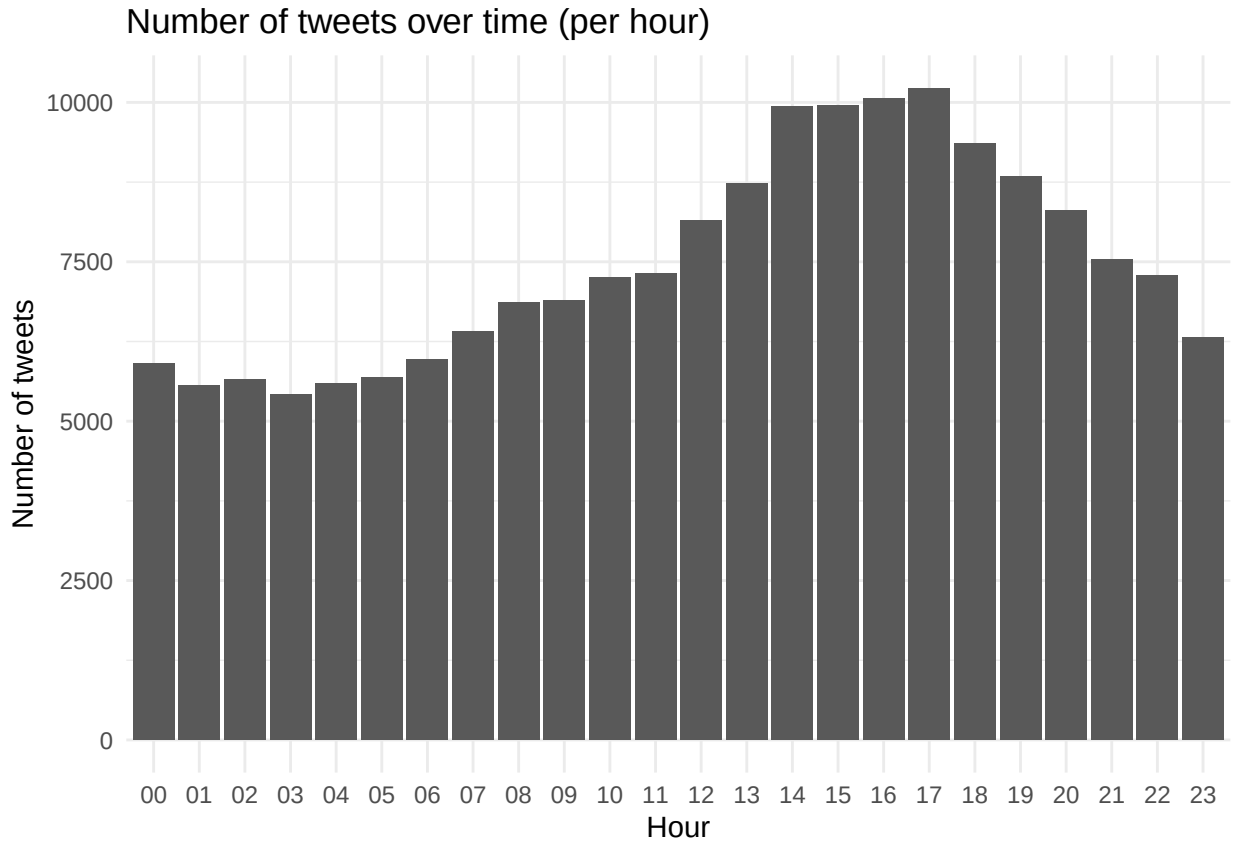
```
# Creating a copy of tweets
tweets_orig <- tweets

# take unique users
Users <- tweets[4:10]
Users = Users[!duplicated(Users$User),]

# Calculating average lenght of tweet
char_counts <- nchar(tweets$Tweet)
av_char_count <- mean(char_counts)
rounded_avg_char_count <- round(av_char_count, 2)
# tabelle einfügen mit rounded_avg_char_count!!!!!

#create median
retweets_median = median(Users$Retweets)
retweets_mean = mean(Users$Retweets)

likes_median = median(Users$Likes)
likes_mean = mean(Users$Likes)

Friends_median = median(Users$UserFriends)
Friends_mean = mean(Users$UserFriends)

Followers_median = median(Users$UserFollowers)
Followers_mean = mean(Users$UserFollowers)

verified_median = median(Users$UserVerified)
verified_mean = mean(Users$UserVerified)

# Create a tibble with the values
my_table <- tibble(
  Statistik = c("Retweets", "Likes", "Friends", "Followers", "Verified"),
  Median = c(retweets_median, likes_median, Friends_median, Followers_median, verified_median),
  Average = c(retweets_mean, likes_mean, Friends_mean, Followers_mean, verified_mean)
)

print(my_table)
```

```
## # A tibble: 5 x 3
##   Statistik Median   Average
##   <chr>      <dbl>     <dbl>
## 1 Retweets       0    0.833
## 2 Likes          1    4.61
## 3 Friends      402 1142.
## 4 Followers    285 5134.
```

```
## 5 Verified       0    0.0226
```

Hier Erklärung Tabelle einfügen. (Joshi)

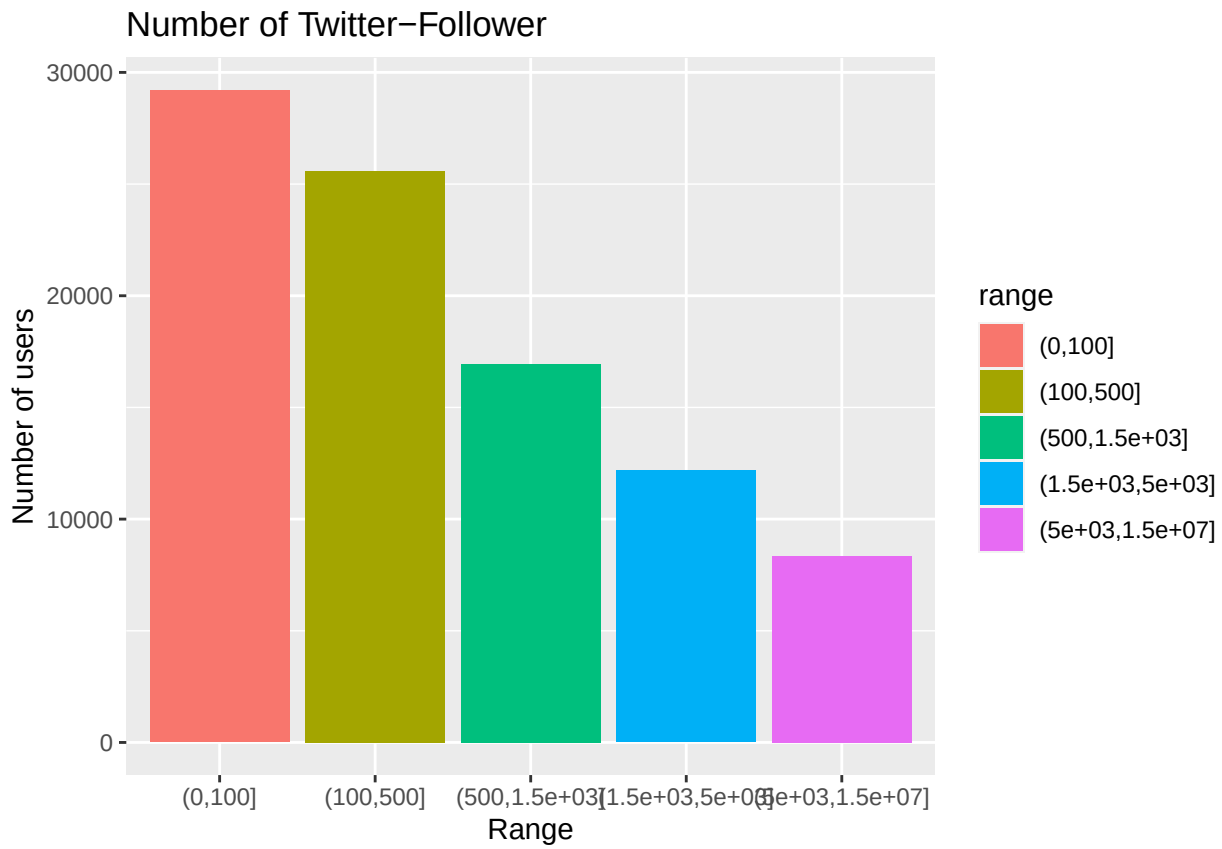
```
#Average/ median of Hour when to tweet, Nr of Retweets, Likes, Followers, Friends, verified

#create Histogramm for Tweettime
plot_dataHour <- tweets %>%
  group_by (timeofday_hour) %>%
  count()

ggplot (plot_dataHour,
        aes (x=timeofday_hour, y=n)) +
  geom_bar(stat = "identity")+
  theme_minimal () +
  ggtitle("Number of tweets over time (per hour)") +
  xlab("Hour") +
  ylab("Number of tweets")
```



Hier Erklärung Grafik einfügen. (Joshi)

```
#Number of tweets tweeted of an user
#range breaks
range_breaks <- c(0, 100, 500, 1500, 5000, 15000000)

#Appling cut() on follower-data
Users$range <- cut(Users$UserFollowers, breaks = range_breaks)
```

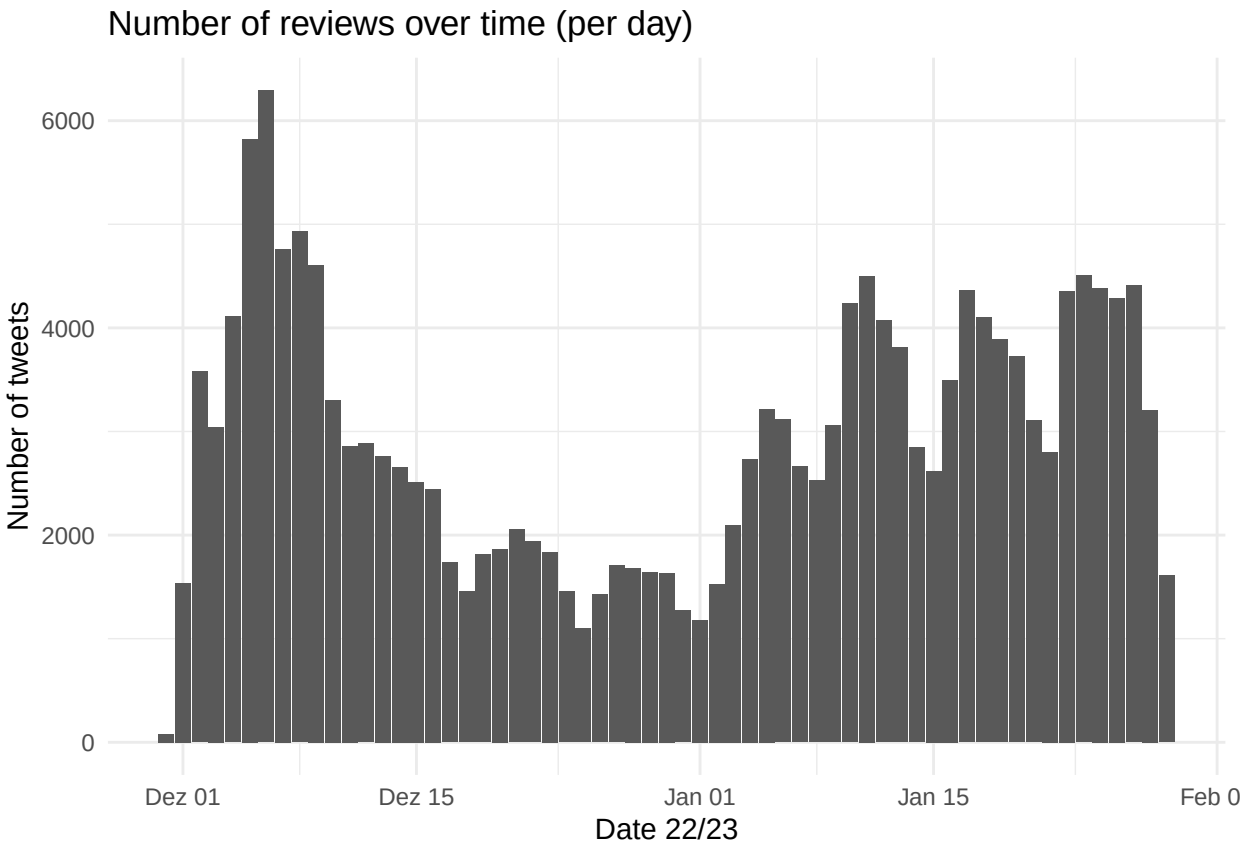
```
Users <- na.omit(Users)

# Creating Barplot
ggplot(Users, aes(x = range, fill = range)) + geom_bar() + labs(title = "Number of Twitter-Follower", x
```




```
# Beschriftung x Achse auf numerisch wechseln !!!!!!
```

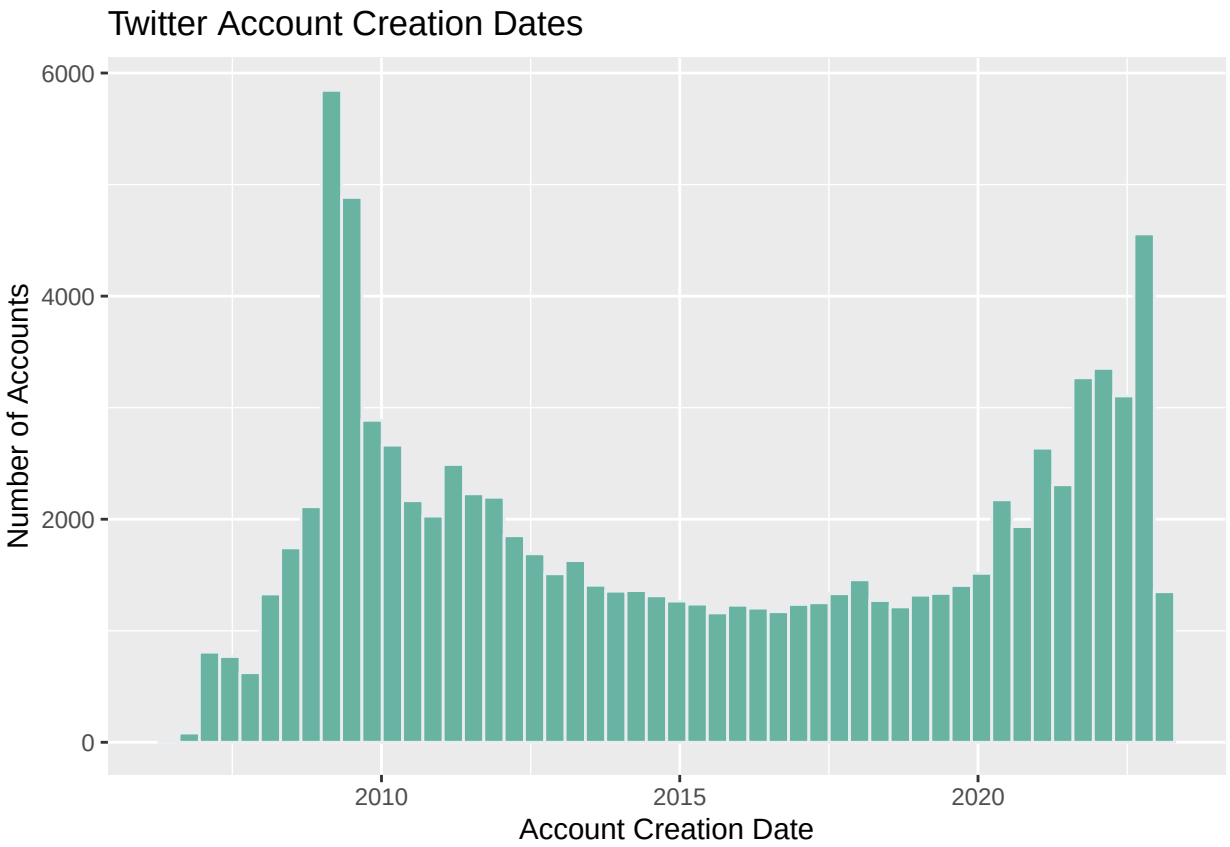
Hier Erklärung Grafik einfügen. (Joshi)

```
#number of tweets over time
plot_data <- tweets %>%
  group_by (tweet_date) %>%
  count()

ggplot (plot_data,
        aes (x=tweet_date, y=n)) +
  geom_bar(stat = "identity")+
  theme_minimal () +
  ggtitle("Number of reviews over time (per day)") +
  xlab("Date 22/23") +
  ylab("Number of tweets")
```



Hier Erklärung Grafik einfügen. (Joshi)

```
# Convert UserCreated to datetime format
Users$UserCreated <- ymd_hms(Users$UserCreated)

# Create the plot
ggplot(Users, aes(x = UserCreated)) +
  geom_histogram(bins = 50, fill = "#69b3a2", color = "#e9ecef") +
  labs(x = "Account Creation Date", y = "Number of Accounts") +
  ggtitle("Twitter Account Creation Dates")
```



Hier Erklärung Grafik einfügen. (Joshi)

2. What are the tweets about, what do users associated the new technology with (e.g. industries, specific applications, and also emotions)?

Pre processing

```
# Define a function to preprocess the text
preprocess_text <- function(text) {

  # Convert text to lower case
  text <- tolower(text)

  # Remove emojis and emoticons
  text <- gsub("[\U0001F600-\U0001F64F\U0001F910-\U0001F96F\U0001F300-\U0001F5FF\U0001F680-\U0001F6FF\U

  # Remove numbers
  text <- gsub("\\d+", "", text)

  # Remove punctuation
  text <- gsub("[[:punct:]]", "", text)

  # Remove whitespace
  text <- gsub("\\s+", " ", text)

  # Remove stopwords and other words to be removed
```

```
  words_to_remove <- c("the", "and", "in", "to", "a", "of")
  words_to_remove_pattern <- paste0("\\b(", paste(words_to_remove, collapse = "|"), ")\\b")
  text <- gsub(words_to_remove_pattern, "", text, ignore.case = TRUE)

  # Return the preprocessed text
  return(text)
}

# Apply the preprocessing function to the Tweet column
tweets$preprocessed_text <- sapply(tweets$Tweet, preprocess_text)
```

Erklärung zu pr Prozessing, bzw. was wir entfernt haben. (Absichtlich nicht buchstaben sonder Füllwörter entfernt)

```
### - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
### STEP 3: PERFORM AUTOMATED CONTENT ANALYSIS
### - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

#Select text column and create your custom dictionary
# dic1 = industries
#tweets$dic1 <- "NA"
tweets$dic1 <-str_count(tweets$preprocessed_text, "artificial intelligence|machine learning|automation|
#tweets$dic1_occurence<- "NA"
tweets$dic1_occurence<-ifelse(tweets$dic1>=2,1,0)
# dic2 = specific applications
tweets$dic2 <-str_count(tweets$preprocessed_text, "chatbot|language modeling|ai|artificial intelligence
tweets$dic2_occurence<-ifelse(tweets$dic2>=2,1,0)
# dic3 = emotions
tweets$dic3 <-str_count(tweets$preprocessed_text, "excited|happy|frustated|angry|sad|amused")
tweets$dic3_occurence<-ifelse(tweets$dic3>=2,1,0)
# dic4 = hype
tweets$dic4 <-str_count(tweets$preprocessed_text, "excited|hyped|thrilled|stoked|pumped")
tweets$dic4_occurence<-ifelse(tweets$dic4>=2,1,0)


sum (tweets$dic1_occurence)
```

```
## [1] 1911
```

```
sum(tweets$dic2_occurence)
```

```
## [1] 42945
```

```
sum(tweets$dic3_occurence)
```

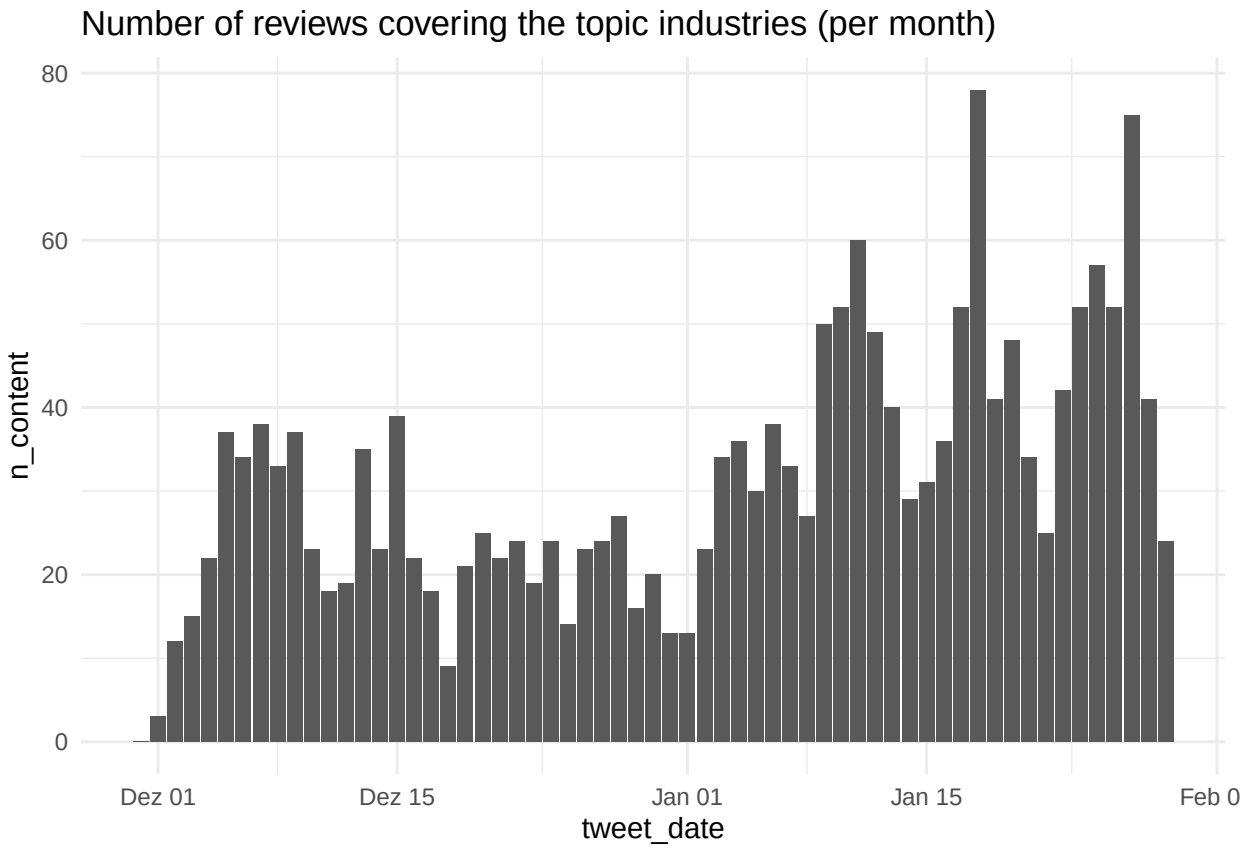
```
## [1] 129
```

```
sum(tweets$dic4_occurence)
```

```
## [1] 21
```

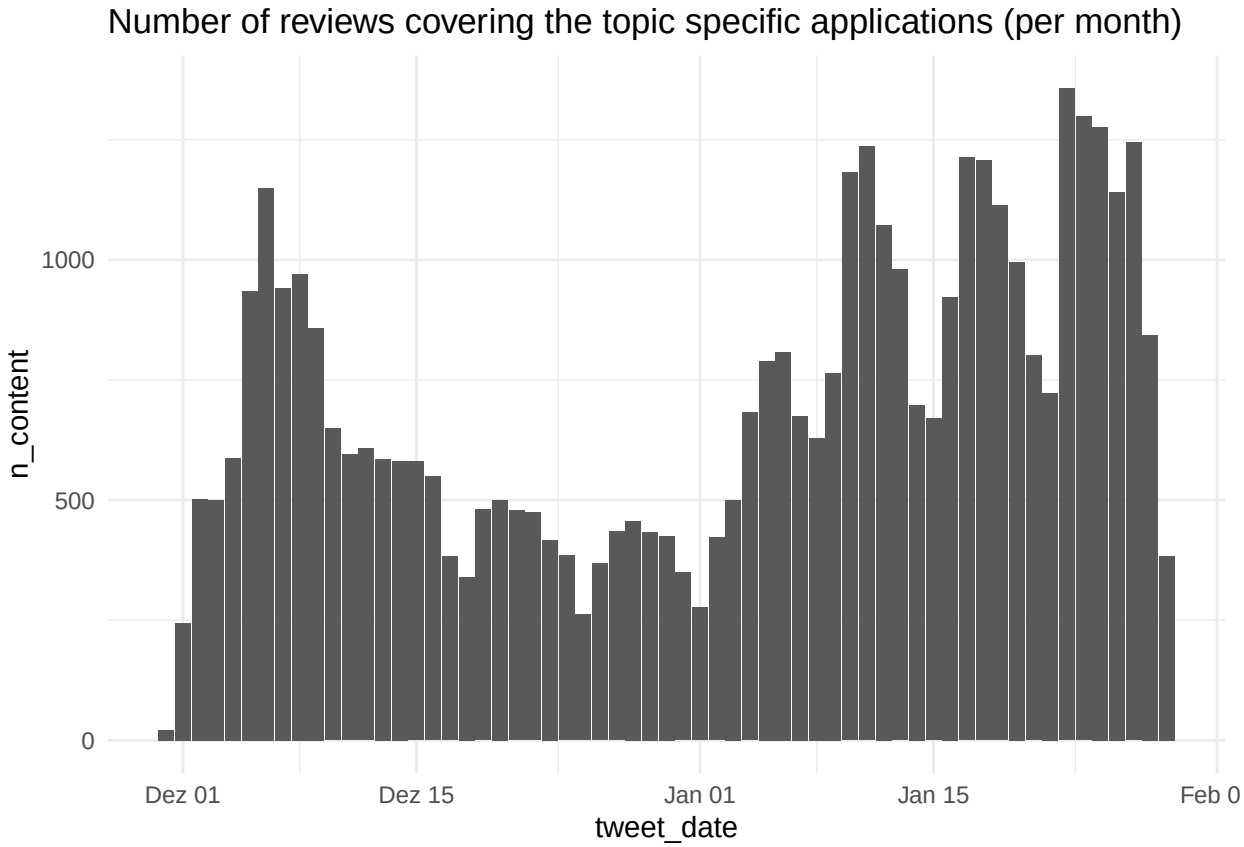
```
## VISUALIZE RESULTS dictionary 1 (industries)
#sum of reviews that cover topic per day
plot_content_data1 <- tweets %>%
  group_by (tweet_date) %>%
  summarise(n_content=sum(dic1_occurence))
```

```
ggplot (plot_content_data1, aes (x=tweet_date, y=n_content)) + geom_bar(stat = "identity")+ theme_minima
```



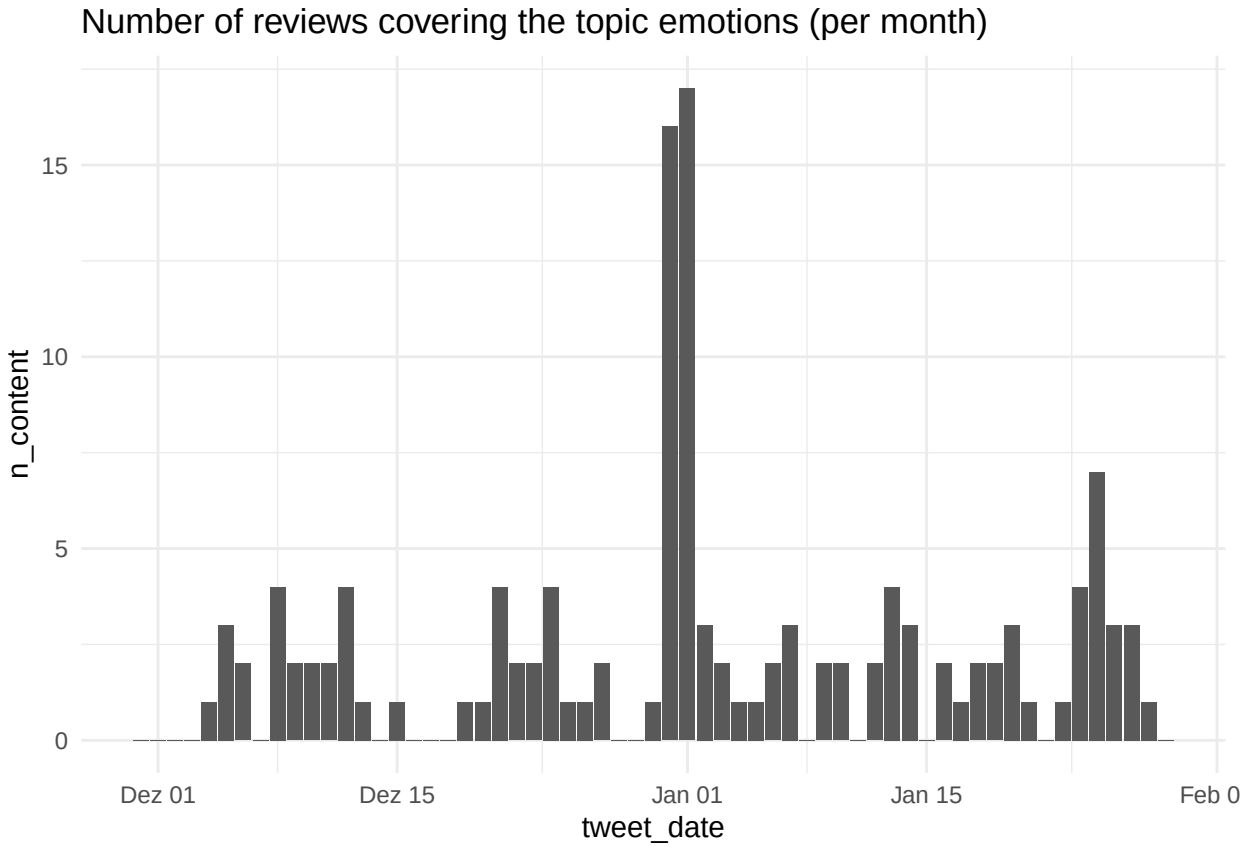
Erklärung einfügen Dictionary industries!!!

```
## VISUALIZE RESULTS dictionary 2 (specific applications)
#sum of reviews that cover topic per day
plot_content_data2 <- tweets %>%
  group_by (tweet_date) %>%
  summarise(n_content=sum(dic2_occurence))

ggplot (plot_content_data2, aes (x=tweet_date, y=n_content)) + geom_bar(stat = "identity")+ theme_minima
```



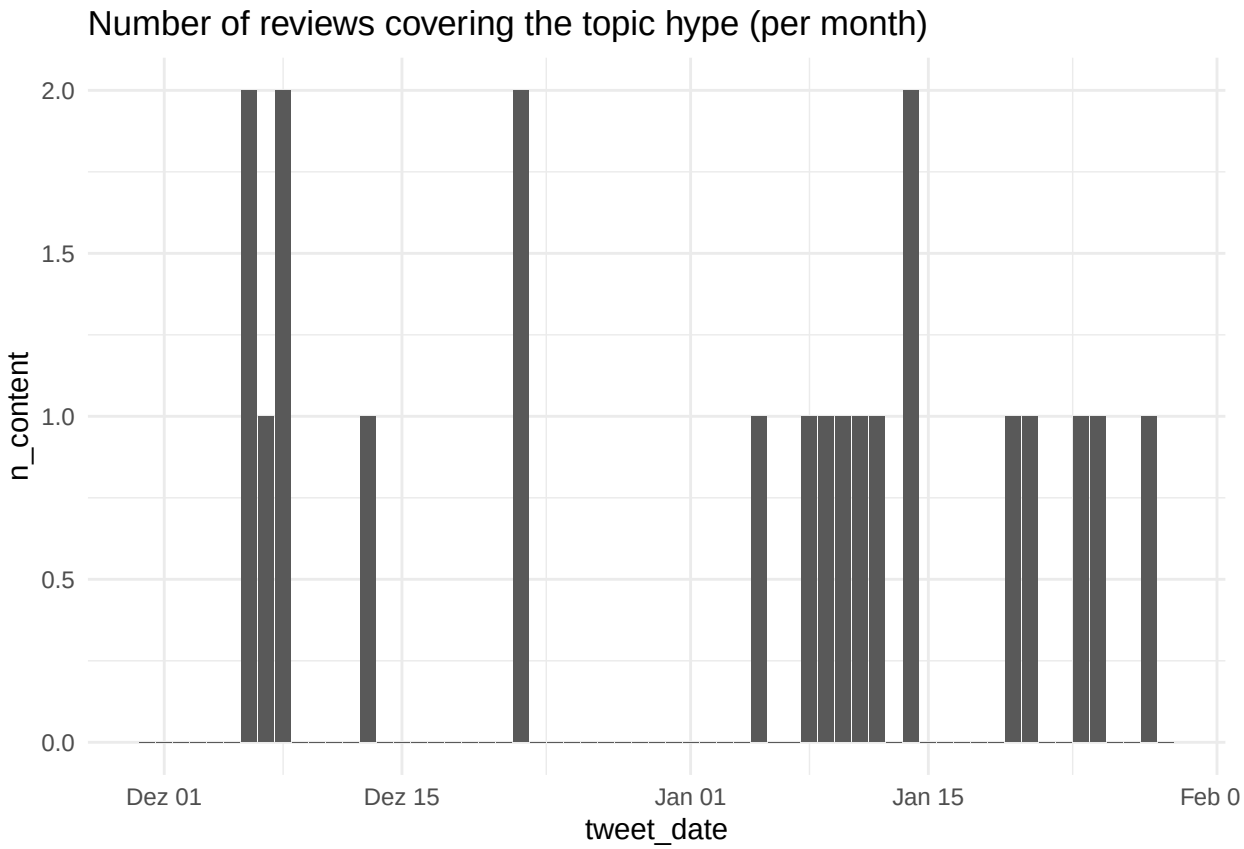
Erklärung einfügen Dictionary specific applications!!!

```
## VISUALIZE RESULTS dictionary 3 (emotions)
#sum of reviews that cover topic per day
plot_content_data3 <- tweets %>%
  group_by (tweet_date) %>%
  summarise(n_content=sum(dic3_occurence))

ggplot (plot_content_data3, aes (x=tweet_date, y=n_content)) + geom_bar(stat = "identity")+ theme_minima
```



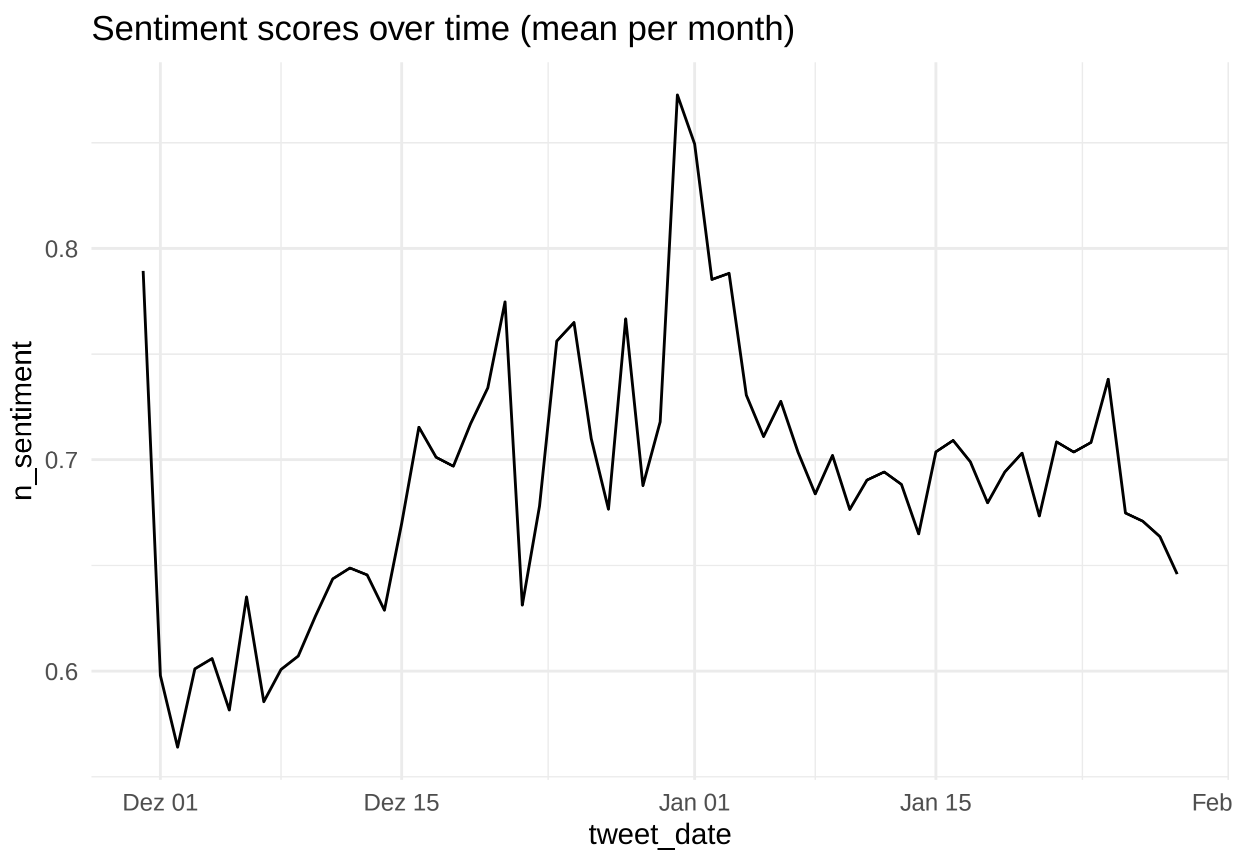
Erklärung einfügen Dictionary emotions!!!

```
## VISUALIZE RESULTS dictionary 4 (hype)
#sum of reviews that cover topic per day
plot_content_data4 <- tweets %>%
  group_by (tweet_date) %>%
  summarise(n_content=sum(dic4_occurence))

ggplot (plot_content_data4, aes (x=tweet_date, y=n_content)) + geom_bar(stat = "identity")+ theme_minima
```



Erklärung einfügen Dictionary hype!!!

```
# Perform sentiment analysis
#Select text column and calculate sentiment scores. You can change the method (e.g."syuzhet", "bing", "
tweets$sentiment <- "NA"
tweets$sentiment <- get_sentiment(tweets$preprocessed_text, method="syuzhet", lang="english")

## VISUALIZE RESULTS
# mean over time
plot_sentiment_data <- tweets %>%
  group_by (tweet_date) %>%
  summarise(n_sentiment=mean(sentiment))

ggplot (plot_sentiment_data, aes (x=tweet_date, y=n_sentiment)) + geom_line()+ theme_minimal () + ggtitl
```




```
### - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
### STEP 5: PERFORM ADDITIONAL ANALYSIS
### - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

#correlation of sentiment with custom dictionary
cor(tweets$dic1, tweets$Likes, method = "pearson")
```

```
## [1] 0.002319274
```

```
cor(tweets$dic1, tweets$sentiment, method = "pearson")
```

```
## [1] 0.0683316
```

```
cor(tweets$dic2, tweets$Likes, method = "pearson")
```

```
## [1] -0.00145943
```

```
cor(tweets$dic2, tweets$sentiment, method = "pearson")
```

```
## [1] 0.119084
```

```
cor(tweets$dic3, tweets$Likes, method = "pearson")
```

```
## [1] 0.002715025
```

```
cor(tweets$dic3, tweets$sentiment, method = "pearson")
```

```
## [1] 0.07935998
```

```
cor(tweets$dic4, tweets$Likes, method = "pearson")
```

```
## [1] -0.0002338922
```

```
cor(tweets$dic4, tweets$sentiment, method = "pearson")
```

```
## [1] 0.07891795
```