hw3_test

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(syuzhet)
```

```
Warning: package 'syuzhet' was built under R version 4.4.2
```

```
library(lubridate)
library(tm)
```

```
Warning: package 'tm' was built under R version 4.4.2
```

```
Loading required package: NLP
```

```
Warning: package 'NLP' was built under R version 4.4.2
```

```
Attaching package: 'NLP'

The following object is masked from 'package:ggplot2':

    annotate
```

```
library(wordcloud)
```

```
Warning: package 'wordcloud' was built under R version 4.4.2

Loading required package: RColorBrewer
```

```
spider_news <- read_delim(here::here('data', 'Data_spider_news_global.csv'), delim = '\t')
```

```
Rows: 6204 Columns: 41
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (25): ID, URL, Language, Country_search, Newspaper, Type_of_newspaper , ...
dbl (16): yr, Year_event, lon, lat, Bite, Death, Figure_species, Figure_bite...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
population <- read_csv(here::here('data', 'world_population.csv'))
```

```
Rows: 234 Columns: 17
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): CCA3, Country/Territory, Capital, Continent
dbl (13): Rank, 2022 Population, 2020 Population, 2015 Population, 2010 Popu...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
wealth <- read_csv(here::here('data', 'wealth', 'API_NY.GDP.MKTP.CD_DS2_en_csv_v2_76261.csv')
```

```
New names:
Rows: 266 Columns: 69
-- Column specification
-------------------------------------------------------- Delimiter: "," chr
(4): Country Name, Country Code, Indicator Name, Indicator Code dbl (64): 1960,
1961, 1962, 1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970, ... lgl (1): ...69
i Use `spec()` to retrieve the full column specification for this data. i
Specify the column types or set `show_col_types = FALSE` to quiet this message.
* `` -> `...69`
```

weigh the bias

```
# spider_news_weighed <- spider_news
#
#
#
# spider_news_weighed[27:38] <- spider_news_weighed %>%
#
#   select(Bite:Photo_error) %>%
#
#   replace_na()



spider_news_weighed <- spider_news %>%

  # impacts to bias score assigned arbitrarily
  mutate(

         # account for NAs
         Bite = replace_na(Bite, 1),
         Death = replace_na(Death, 1),
         Figure_species = replace_na(Figure_species, 1),
         Figure_bite = replace_na(Figure_bite, 1),

         # having an expert will reduce bias score
         Expert_arachnologist = replace_na(Expert_arachnologist, 1),
         Expert_doctor = replace_na(Expert_doctor, 1),
         Expert_others = replace_na(Expert_others, 1),

         # sensationalism give sever bias penalty
         Sensationalism = replace_na(Sensationalism, 1),
```

```
        # multiple types of error will compound
        Taxonomic_error = replace_na(Taxonomic_error, 1),
        Venom_error = replace_na(Venom_error, 1),
        Anatomy_error = replace_na(Anatomy_error, 1),
        Photo_error = replace_na(Photo_error, 1),

        Bite = Bite * 1,
        Death = Death * 1,
        Figure_species = Figure_species * 1,
        Figure_bite = Figure_bite * 2,

        # having an expert will reduce bias score
        Expert_arachnologist = Expert_arachnologist * -2,
        Expert_doctor = Expert_doctor * -1,
        Expert_others = Expert_others * -1,

        # sensationalism give sever bias penalty
        Sensationalism = Sensationalism * 5,

        # multiple types of error will compound
        Taxonomic_error = case_when(
          Taxonomic_error != 0 ~ Taxonomic_error * 2,
          Taxonomic_error == 0 | is.na(Taxonomic_error) == TRUE ~ 1),
        Venom_error = case_when(
          Venom_error != 0 ~ Venom_error * 2,
          Venom_error == 0 | is.na(Venom_error) == TRUE ~ 1),
        Anatomy_error = case_when(
          Anatomy_error != 0 ~ Anatomy_error * 2,
          Anatomy_error == 0 | is.na(Anatomy_error) == TRUE ~ 1),
        Photo_error = case_when(
          Photo_error != 0 ~ Photo_error * 2,
          Photo_error == 0 | is.na(Photo_error) == TRUE ~ 1),

        Total_error = Bite + Death + Figure_species + (Taxonomic_error * Venom_error * Anato

        )
```

clean and subset population dataset

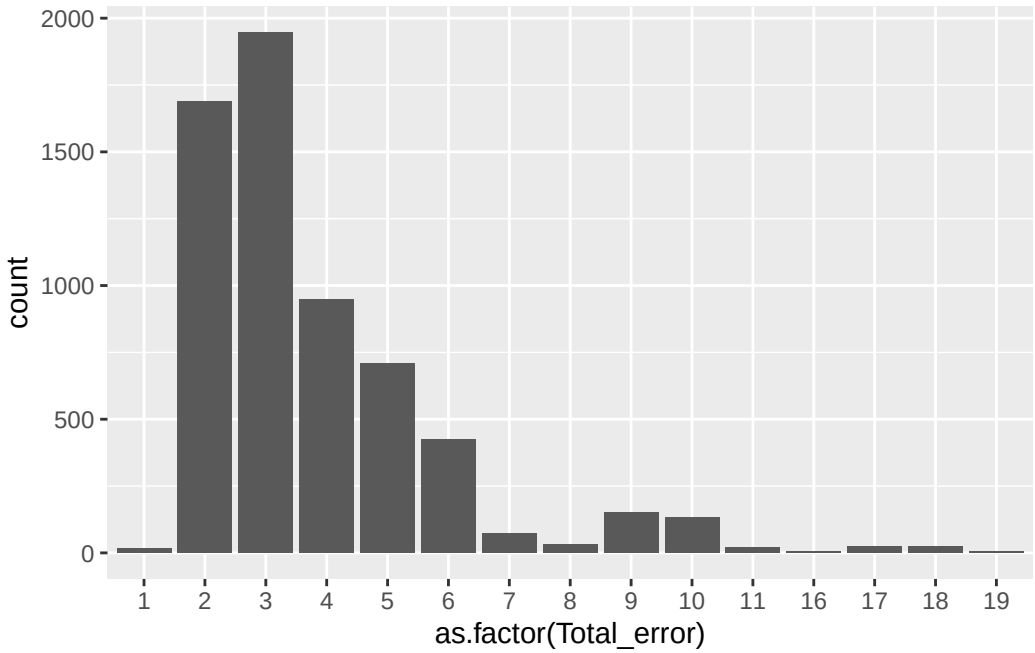
```
population_sub <- population %>%
  janitor::clean_names() %>%
```

```
  select(country_territory, x2022_population) %>%
  mutate(country_territory = case_match(country_territory,
    'Bosnia and Herzegovina' ~ 'Bosnia',
    'United Kingdom' ~ 'UK',
    'United States' ~ 'USA',
    .default = country_territory))
```

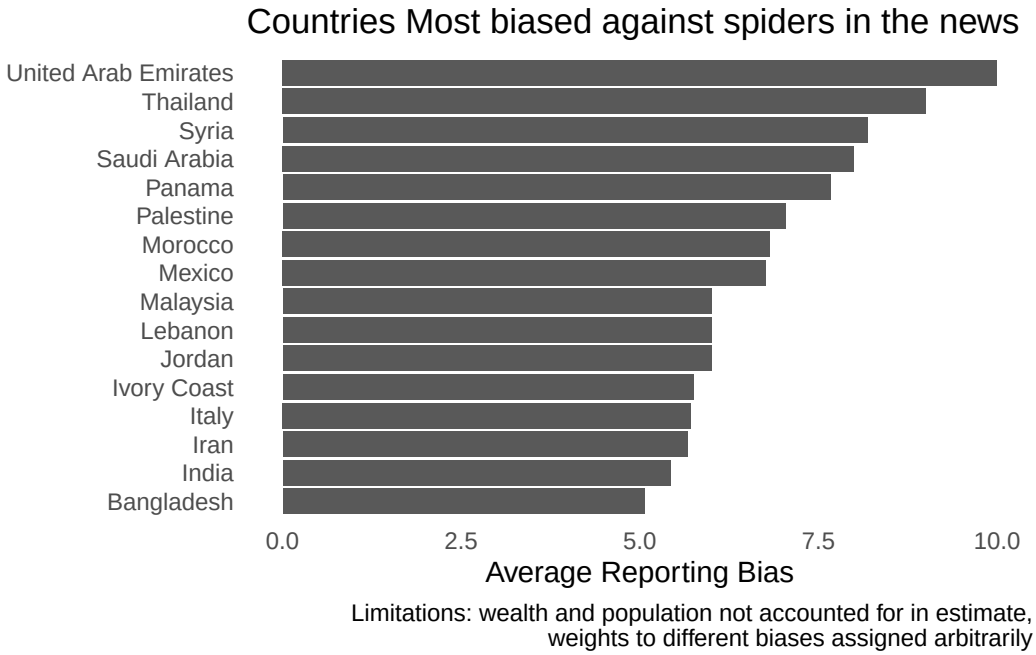
clean and subset wealth dataset

```
wealth_sub <- wealth %>%
  janitor::clean_names() %>%
  select(country_name, x2023, x2022, x2014) %>%
  mutate(country_name = case_match(country_name,
    'Bosnia and Herzegovina' ~ 'Bosnia',
    'Czechia' ~ 'Czech Republic',
    'Egypt, Arab Rep.' ~ 'Egypt',
    'Iran, Islamic Rep.' ~ 'Iran',
    "Cote d'Ivoire" ~ 'Ivory Coast',
    'Kyrgyz Republic' ~ 'Kyrgyzstan',
    'Russian Federation' ~ 'Russia',
    'Korea, Rep.' ~ 'South Korea',
    'Turkiye' ~ 'Turkey',
    'United Kingdom' ~ 'UK',
    'United States' ~ 'USA',
    'Venezuela, RB' ~ 'Venezuela',
    'Syrian Arab Republic' ~ 'Syria',
    .default = country_name)) %>%
  filter(country_name != 'Venezuela')

# NOTE: palestine, taiwan, venezuela omitted from world bank dataset
```

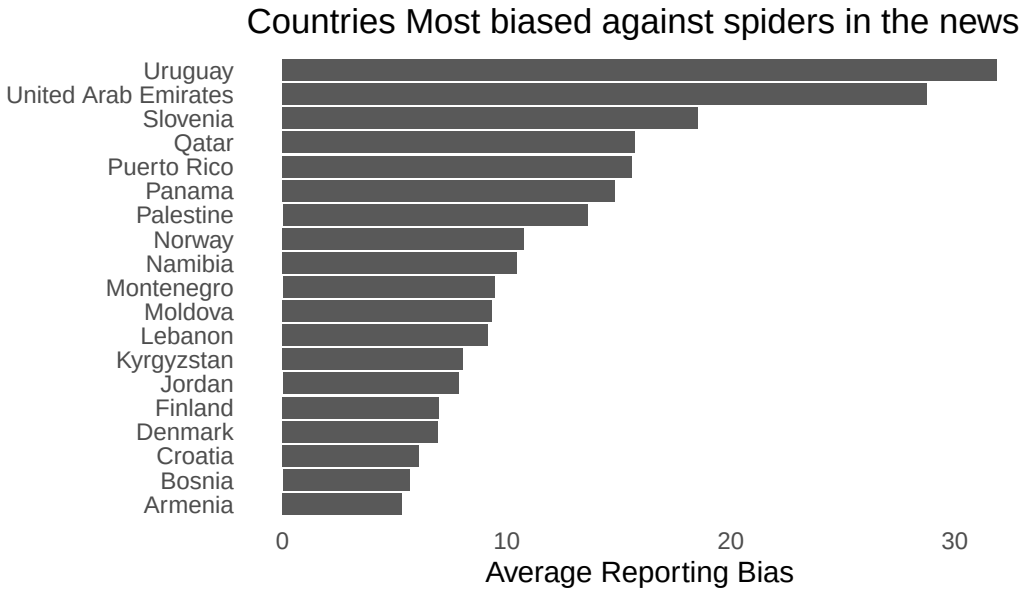
```
spider_news_weighed %>%
  group_by(Total_error) %>%
  summarize(count = n()) %>%
  ggplot(aes(x = as.factor(Total_error), y = count)) +
  geom_col()
```




```
spider_news_weighed %>%
  group_by(Country_search) %>%
  summarize(avg_bias = mean(Total_error)) %>%
  filter(avg_bias > 5) %>%
  na.omit() %>%
  left_join(population_sub, by = join_by(Country_search == country_territory)) %>%


  ggplot(aes(x = Country_search, y = sort(avg_bias))) +
  geom_col() +
  coord_flip() +
  labs(title = 'Countries Most biased against spiders in the news',
       y = 'Average Reporting Bias',
       caption = "Limitations: wealth and population not accounted for in estimate,\nweights
  theme_minimal() +
  theme(panel.grid = element_blank(),
        axis.title.y = element_blank())
```




```
spider_news_weighed %>%
  group_by(Country_search) %>%
  na.omit() %>%
  summarize(avg_bias = mean(Total_error)) %>%
  left_join(population_sub, by = join_by(Country_search == country_territory)) %>% mutate(bia
  filter(bias_per_cap > 5) %>%


  ggplot(aes(x = Country_search, y = sort(bias_per_cap))) +
  geom_col() +
  coord_flip() +
  labs(title = 'Countries Most biased against spiders in the news',
       y = 'Average Reporting Bias',
       caption = "Limitations: wealth and population not accounted for in estimate,\nweights
  theme_minimal() +
  theme(panel.grid = element_blank(),
        axis.title.y = element_blank())
```



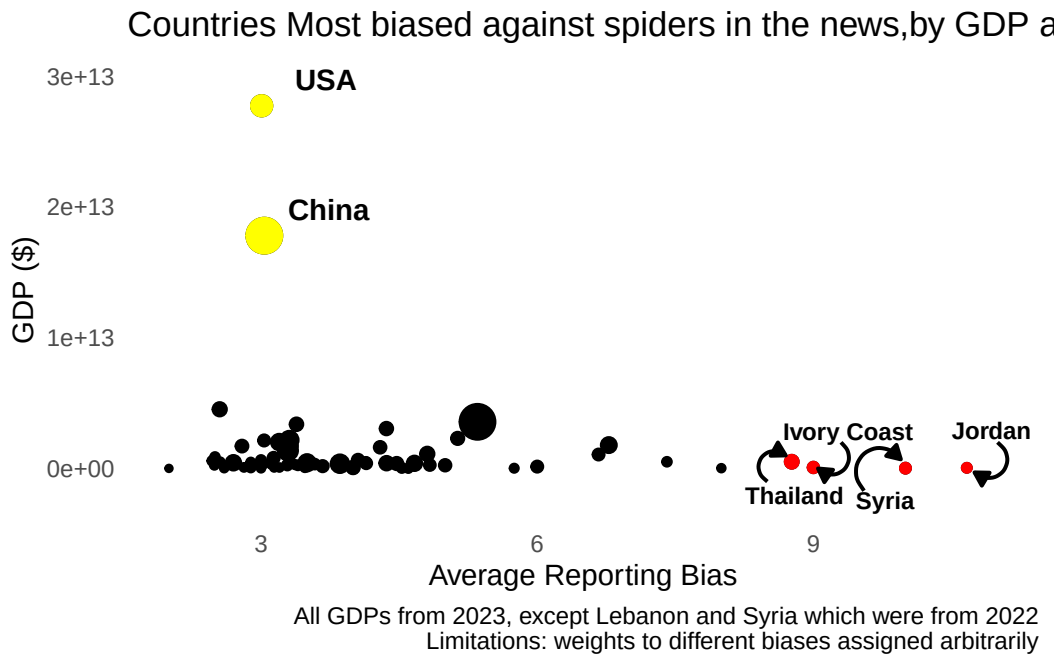

```
df_gdp <- spider_news_weighed %>%
  group_by(Country_search) %>%
  na.omit() %>%
  summarize(avg_bias = mean(Total_error)) %>%
  full_join(wealth_sub, by = join_by(Country_search == country_name)) %>%
#  filter(is.na(x2023) == TRUE) %>%
  filter(Country_search != 'Palestine' & Country_search != 'Taiwan') %>%
  mutate(gdp = case_when(
    is.na(x2023) == FALSE ~ x2023,
    is.na(x2023) == TRUE & is.na(x2022) == FALSE ~ x2022,
    is.na(x2023) == TRUE & is.na(x2022) == TRUE & is.na(x2014) == FALSE ~ x2014
  )) %>%
  select(Country_search, avg_bias, gdp) %>%
  filter(is.na(avg_bias) == FALSE)


df_pop <- spider_news_weighed %>%
  group_by(Country_search) %>%
  na.omit() %>%
  summarize(avg_bias = mean(Total_error)) %>%
  left_join(population_sub, by = join_by(Country_search == country_territory))

df_full <- left_join(df_gdp, df_pop, by = 'Country_search') %>%
```

```
  select(-avg_bias.y) %>%
  rename('avg_bias' = 'avg_bias.x',
         'population' = 'x2022_population',
         'country' = 'Country_search')
```

```
  ggplot() +
  geom_point(data = df_full,
             mapping = aes(x = avg_bias,
                           y = gdp,
                           size = population),
             show.legend = FALSE) +

  geom_point(data = subset(df_full, avg_bias > 8),
             mapping = aes(x = avg_bias,
                           y = gdp,
                           size = population),
             color = 'red',
             show.legend = FALSE) +
  geom_text(data = subset(df_full, avg_bias > 8),
            mapping = aes(x = avg_bias,
                          y = gdp,
                          label = country),
            size = 3,
            position = position_jitter(0.5, 3e12, 20),
            fontface = 'bold') +

  geom_point(data = subset(df_full, gdp > 1e13),
             mapping = aes(x = avg_bias,
                           y = gdp,
                           size = population),
             color = 'yellow',
             show.legend = FALSE) +
  geom_text(data = subset(df_full, gdp > 1e13),
            mapping = aes(x = avg_bias,
                          y = gdp,
                          label = country),
            nudge_x = 0.7,
            nudge_y = 2e12,
            fontface = 'bold') +
  geom_curve(aes(x = c(8.5, 9.3, 9.55, 11),
                 y = c(-1e12, 2e12, -1.7e12, 2e12),
                 xend = c(8.7, 9.05, 9.95, 10.75),
```

```
                yend = c(1e12, 0, 7e11, -3e11)),
             curvature = c(-1.2),
             lwd = 0.7,
             arrow = arrow(type = 'closed', length = unit(0.2, 'cm'))) +
#  coord_flip() +
  labs(title = 'Countries Most biased against spiders in the news,by GDP and population',
       x = 'Average Reporting Bias',
       y = 'GDP ($)',
       caption = "All GDPs from 2023, except Lebanon and Syria which were from 2022\nLimitati
  theme_minimal() +
  theme(panel.grid = element_blank())
```

```
Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_point()`).
```

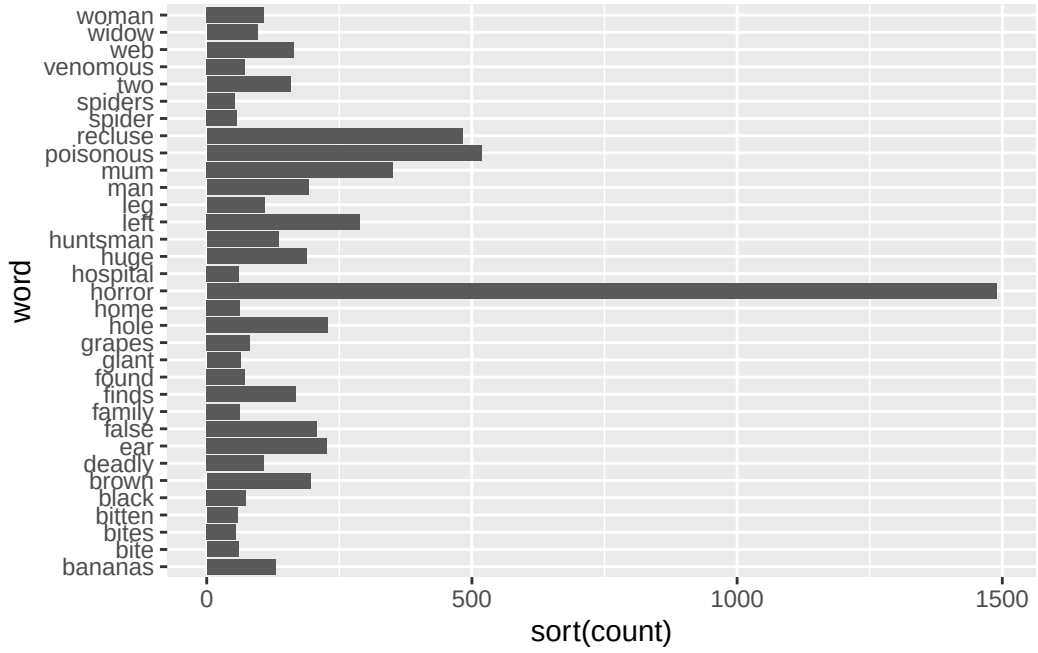



```
all_english <- spider_news %>%
  filter(Language == 'English') %>%
  select(Title) %>% as_vector() %>% VectorSource() %>%
  SimpleCorpus()

all_english_senti <- all_english %>%
```

```
  tm_map(tolower) %>%
  tm_map(removePunctuation) %>%
  tm_map(removeNumbers) %>%
  tm_map(removeWords, stopwords('english')) %>%
#  tm_map(stemDocument) %>%
  tm_map(stripWhitespace)

tdm <- TermDocumentMatrix(all_english_senti) %>% as.matrix()

row_sums <- rowSums(tdm)
```

```
sentiment_tib <- enframe(row_sums, name = "word", value = "count")
```

```
ggplot(subset(sentiment_tib, count > 50), aes(word, sort(count))) +
  geom_col() +
  coord_flip()
```

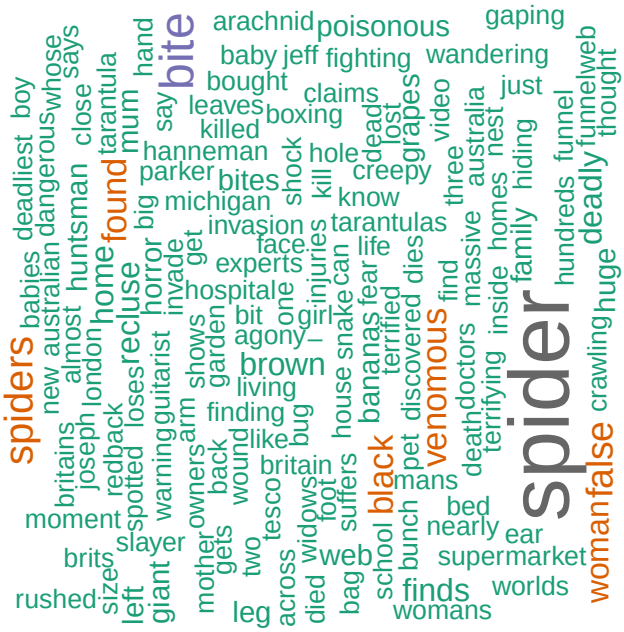



```
wordcloud(words = names(row_sums),
          freq = row_sums,
          max.words = 150,
          random.order = TRUE,
```

```
          min.freq = 5,
          colors = brewer.pal(8, 'Dark2'),
          scale = c(2.7, 0.8),
          rot.per = 0.7,
          use.r.layout = FALSE)
```

```
Warning in wordcloud(words = names(row_sums), freq = row_sums, max.words = 150,
: flesheating could not be fit on page. It will not be plotted.
```

```
Warning in wordcloud(words = names(row_sums), freq = row_sums, max.words = 150,
: widow could not be fit on page. It will not be plotted.
```




```
Warning in wordcloud(words = names(row_sums), freq = row_sums, max.words = 150,
: man could not be fit on page. It will not be plotted.
```

```
Warning in wordcloud(words = names(row_sums), freq = row_sums, max.words = 150,
: brazilian could not be fit on page. It will not be plotted.
```

```
Warning in wordcloud(words = names(row_sums), freq = row_sums, max.words = 150,
: horrific could not be fit on page. It will not be plotted.
```

```
Warning in wordcloud(words = names(row_sums), freq = row_sums, max.words = 150,
: bitten could not be fit on page. It will not be plotted.
```

```
sentiment <- row_sums %>%
  iconv() %>%
  get_nrc_sentiment()
```

```
sentiment2 = c()

for (i in row_sums) {

sentiment2 <- c(sentiment2, syuzhet::get_sent_values(i))
}
```