Introduction to Package.10880416.10873945.10831920.10819616 and Russian-Ukraine Conflict Analysis

This vignette contain introduction to Introduction to Package.10880416.10873945.10831920.10819616 as well as discussion on the analysis of Russia_Ukraine Conflicts.

On 24 February 2022, Russia invaded Ukraine in a major escalation of the Russo_Ukrainian War that began in 2014.

The two csv files used for this analysis are collections of news articles from the New York Times (NYT) and the Guardian, related to the ongoing conflict between Russia and Ukraine. The publishing date of articles ranges from Feb 1st, 2022 to Jul 31st, 2022. The titles of both files indicate whether the news articles were published on the NYT or the Guardian.

Both files include the following variables:

- **published:** the date in which the article was published,
- **headlines:** the journal headlines,
- **articles:** the text of the article.

Data Preparation

We merge the two csv files provided from NYT and Guardian new articles to obtain a joint dataset called "ukraine_joint_dataset". The newly created dataset include a new variable indicating the journal where the articles were published on. The code below shows how the joint dataset was obtained.

```
## import the two datasets
NYT <- read.csv("NYT_Russia_Ukraine.csv")
guardian <- read.csv("Guardians_Russia_Ukraine.csv")

## Convert the date column of the two datasets to date format
guardian$published <- as.Date(guardian$published)
NYT$published <- as.Date(NYT$published)

## Concatenate (stack) the two datasets vertically
merged_data <- rbind(guardian, NYT)

## Create a new column "journal" based on the source of each row
merged_data$journal <- ifelse(row.names(merged_data) %in% row.names(guardian),
                              "Guardian", "NYT")
```

Let's explore the joint dataset by printing it out:

```
## Define a function to truncate text and add "..."
truncate_text <- Vectorize(function(text, max_chars) {
  ifelse(nchar(text) > max_chars, paste0(str_sub(text, end = max_chars), "..."), text)
}, vectorize.args = c("text", "max_chars"))

## Apply the truncation function to the specified columns
data_truncated <- merged_data
data_truncated$headlines <- truncate_text(data_truncated$headlines, 20)
data_truncated$articles <- truncate_text(data_truncated$articles, 25)

head(data_truncated, n = 15)
#>     published               headlines                     articles  journal
#> 1  2022-08-01 Sanctions against Ru... Simon Jenkins (The rouble... Guardian
#> 2  2022-07-26 Can Ukrainian forces... In the first phase of the... Guardian
#> 3  2022-08-05 Nightlands review - ... Who exactly is the enemy ... Guardian
#> 4  2022-08-02 Russia claims US 'di... The role of American inte... Guardian
#> 5  2022-07-27 Is Russia killing of... The International Space S... Guardian
#> 6  2022-06-21 Russia blocks Telegr... Russia has blocked the we... Guardian
#> 7  2022-07-28 Brittney Griner lawy... Brittney Griner's defence... Guardian
#> 8  2022-05-22 Russia bans 963 Amer... Russia on Saturday releas... Guardian
#> 9  2022-07-15 Russia escalating at... A top Ukrainian official ... Guardian
#> 10 2022-06-14 Russia bans 29 UK jo... Russia has banned 29 memb... Guardian
#> 11 2022-06-12 McDonald's restauran... Former McDonald's restaur... Guardian
#> 12 2022-07-08 Putin claims Russia ... Vladimir Putin has issued... Guardian
#> 13 2022-07-31 Ukrainian offensive ... Russia is moving large nu... Guardian
#> 14 2022-07-26 Russia seeks to play... The Kremlin has insisted ... Guardian
#> 15 2022-07-19 Germany worries abou... Germans are fretting abou... Guardian
```

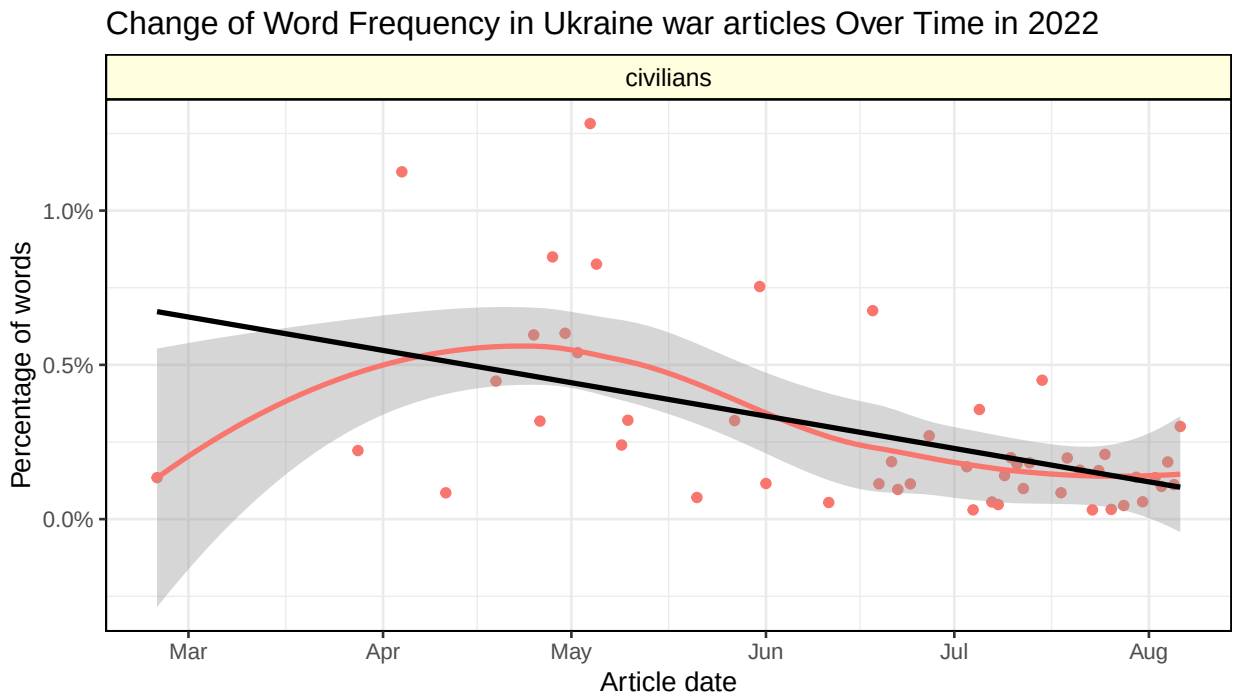
Change in the frequency of given words in Ukraine Conflict dataset over time

Using the function **plot_word_proportion** from our package, we will describe the change in the frequency of some words from the Ukraine conflict dataset over time. As an example, the function can take a single words or a list of words as the only argument and produce a vizualization of the change in the proportion of the given words over time.

The graph is a scatterplot which display the trend in the proportion of the words and over time. A smoothed line was added with confidence interval to aid the eyes in seeing patterns in the points. In addition, a linear regression model was fitted on the plot to describe the direction of the trend.

Let's explore some examples below:

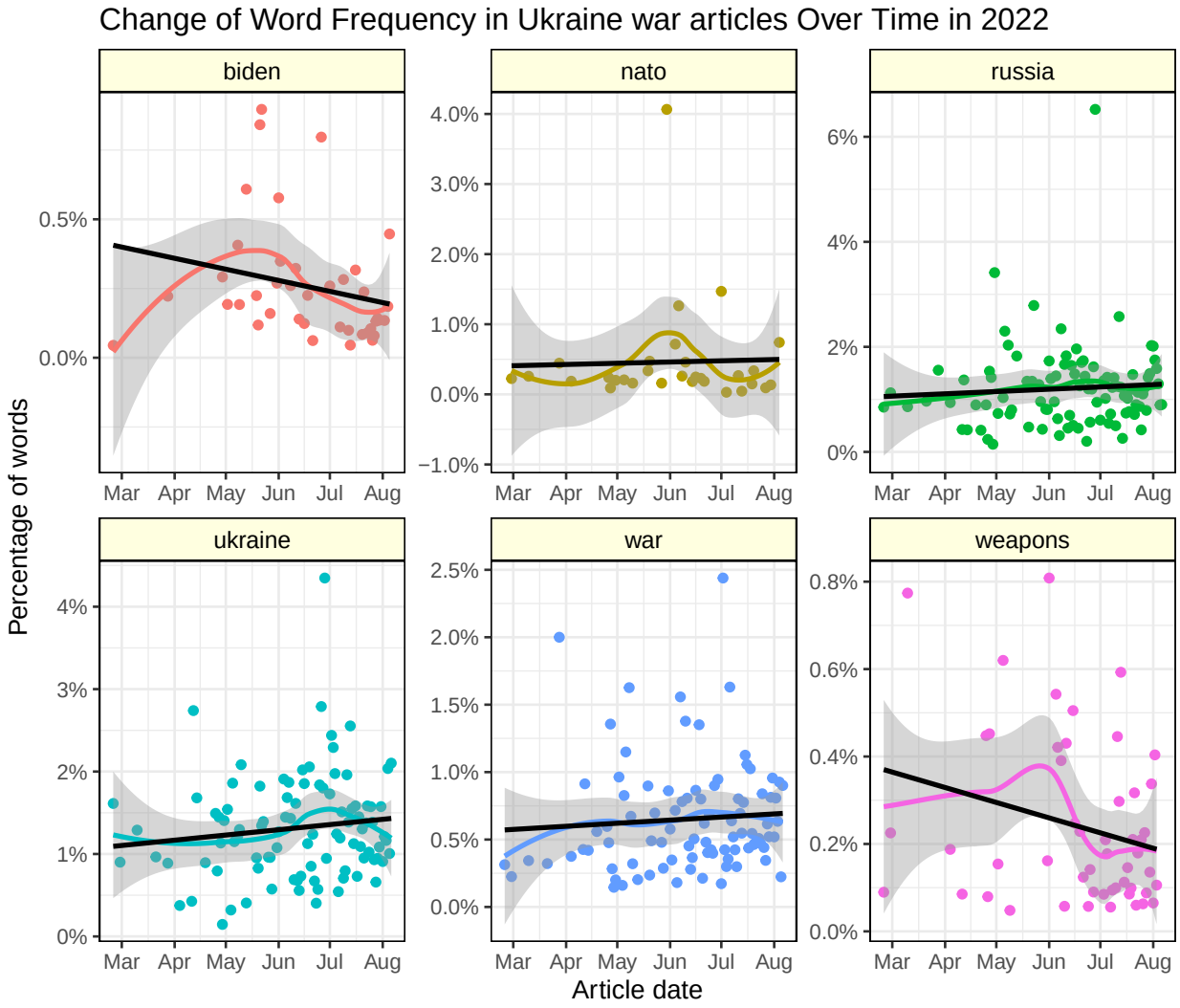
```
## function example with one given word
plot_word_proportion("civilians")
```



As shown in the example above, the use of word **"civilians"** shows a negative trend from March to August. This indicated that the mention of the word civilians reduces as we shift from March to August.

In the example below, a list of words will be provided to the function:

```
## function example with a given list of words
given_words <- c("war", "russia", "ukraine", "weapons", "biden", "nato")
plot_word_proportion(given_words)
```



The example above shows how the graph appear when a list of words is given. The plot for each word was wrapped in a facet wrap with a free scale. Notably, some words like Russia, Ukraine and war have positive trend over time. This indicated that the frequency of these words in the Russia_Ukraine conflict articles increased over time. In contrast, the words weapons and biden are shown to follow a negative trend as their frequencies decrease over time. NATO shows a slight positive trend.

Conclusively, the graph shown a positive and negative trends in the use of certain words during Russia_Ukraine conflicts. Words with positive trends are more commonly used in articles towards the end of the articles collection period while those with negative trends are common during the early period of the conflict.

Details into how the package works

The steps for obtaining the joint dataset used in the package was discussed earlier under data preparation.

The articles column in the joint dataset was further cleaned by removing numbers and stripping whitespaces using *removeNumbers()* and *stripWhitespace()* functions respectively from **tm** package. The articles were then tokenized using the **unnest_tokens** function from **tidytext package** and stop words are removed.

We then work out the proportion of each word in the articles and group the words by the article date. The data after the processing and working out proportion looks like this:

```
#> # A tibble: 6 x 4
#> # Groups:   published [1]
#>   published  word          n        p
#>   <date>     <chr>     <int>    <dbl>
#> 1 2022-02-24 abide         1 0.000448
#> 2 2022-02-24 access        1 0.000448
#> 3 2022-02-24 according     4 0.00179
#> 4 2022-02-24 accused       2 0.000896
#> 5 2022-02-24 achieved      1 0.000448
#> 6 2022-02-24 action        1 0.000448
```

The data was then filtered based on the given words. The output would be a dataframe containing the proportion of the given words grouped by article date such as the one given in the example below given the word **civilians**:

```
#> # A tibble: 6 x 4
#> # Groups:   published [6]
#>   published  word          n        p
#>   <date>     <chr>     <int>    <dbl>
#> 1 2022-02-24 civilians     3 0.00134
#> 2 2022-03-28 civilians     1 0.00222
#> 3 2022-04-04 civilians     6 0.0113
#> 4 2022-04-11 civilians     1 0.000853
#> 5 2022-04-19 civilians     4 0.00447
#> 6 2022-04-25 civilians     4 0.00597
```

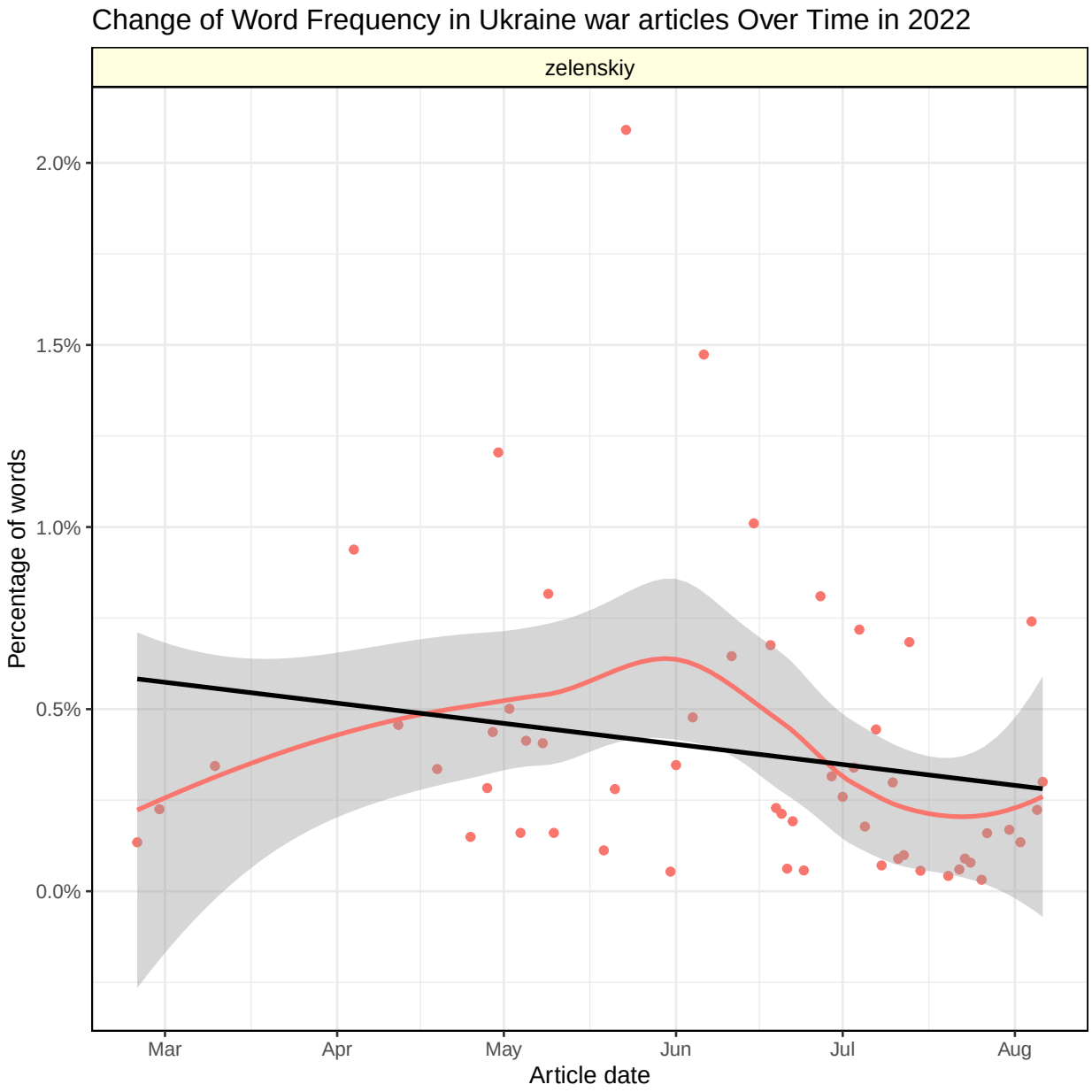
The filtered data containing the given words were then plotted as seen in the examples provided earlier. Provided that the given word(s) is not present in the Ukraine conflict dataset, the function outputs a message saying **"The word you provided is not present in the ukraine conflict joint dataset."**

For example, let's say a particular word "vignette" is given which is not present in the dataset, the output of the function will be:

```
plot_word_proportion("vignette")
#> [1] "The word you provided is not present in the ukraine conflict joint dataset."
```

For a word that is present in the dataset, the output of the filtered data will be a graph showing the change in the frequency of the word over time as shown below:

```
plot_word_proportion("zelenskiy")
```


Change of Word Frequency in Ukraine war articles Over Time in 2022
zelenskiy
2.0%
1.5%
1.0%
0.5%
0.0%
Percentage of words
Mar
Apr
May
Jun
Jul
Aug
Article date

Plotting the words with the highest tf_idf index for each journal

The tf_idf index computes the frequency of a term adjusted for how rarely it is used. The tf_idf index is calculated as the product of the term frequency (tf) and the inverse document frequency (idf):

- tf_idf = tf × idf.

The term frequency (tf) identifies how frequently a word occurs in a document. However, many common words such as "the", "is", "for", etc. typically achieve the highest tf values. The inverse document frequency (idf) decreases the weight for commonly used words and increases the weight for words that are not used very much in a collection of documents.

In the illustration below, we calculate the tf, idf and tf_idf indexes for the Ukraine Conflict dataset, and extracts the top 10 tf_idf words for each journal.

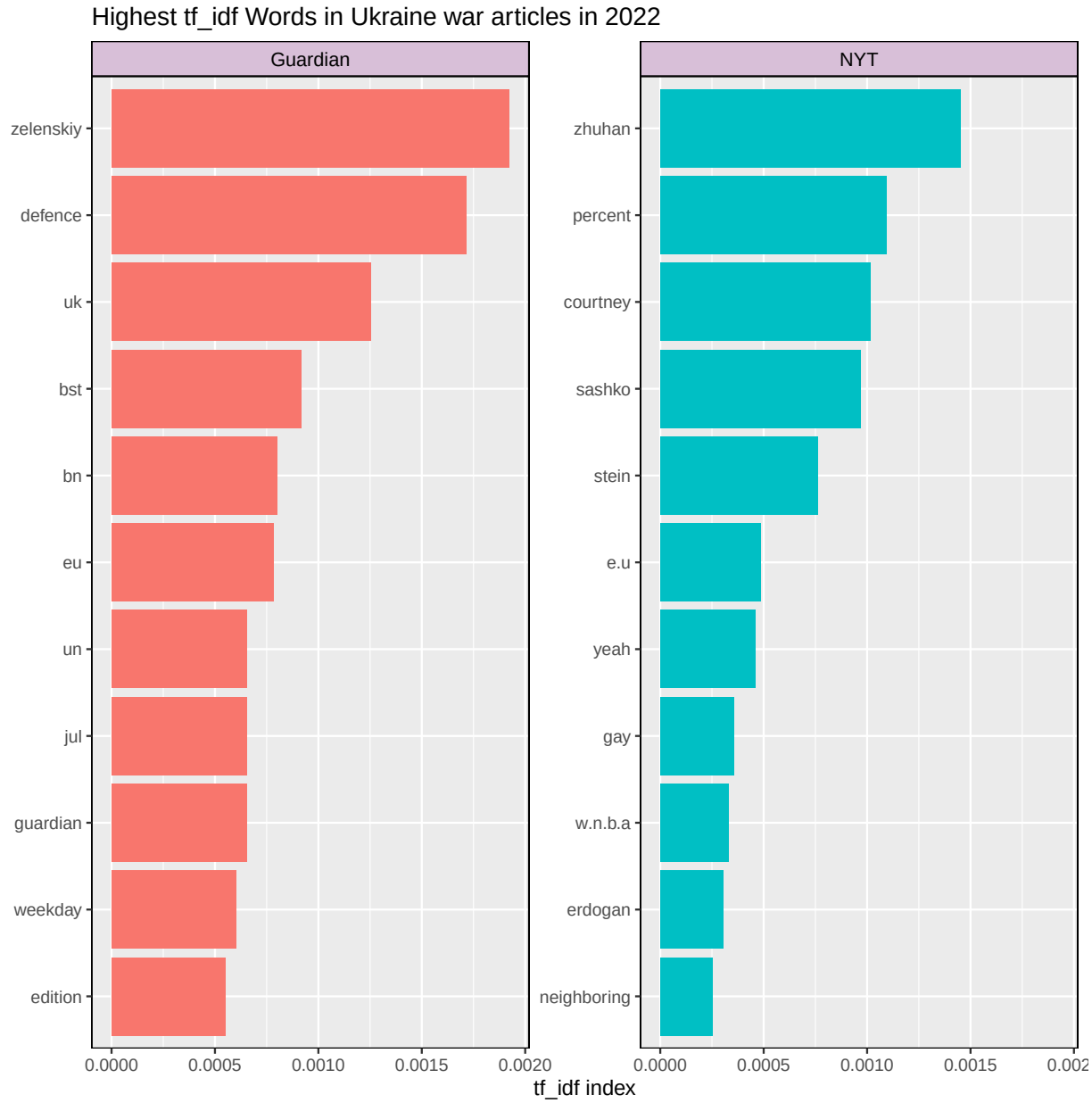
First, we prepare the dataset by removing numbers and whitespaces, followed by tokenizing the articles and remove stop words in a process similar to the one explained earlier in the package details. The *tf*, *idf* and *tf_idf* indexes were then calculated using the **bind_tf_idf** function from the tidytext package.

The R code below illustrates the process and its output:

```
## Create a copy of the joint dataset
df_copy <- merged_data

## remove whitespace and numbers from the articles
df_copy$articles <- stripWhitespace(df_copy$articles)
df_copy$articles <- removeNumbers(df_copy$articles)

## tokenize the articles and remove stop words
clean_data <- df_copy %>%
    unnest_tokens(word, articles) %>%
    anti_join(get_stopwords())
#> Joining with `by = join_by(word)`

## calculates the tf, idf and tf_idf indexes
df_tfidf <- clean_data %>%
  count(journal, word) %>%
  bind_tf_idf(word, journal, n) %>%
  arrange(desc(tf_idf))

head(df_tfidf)
#>    journal      word   n          tf       idf      tf_idf
#> 1 Guardian zelenskiy 224 0.002773822 0.6931472 0.001922667
#> 2 Guardian   defence 200 0.002476627 0.6931472 0.001716667
#> 3      NYT    zhuhan  57 0.002092741 0.6931472 0.001450578
#> 4 Guardian        uk 146 0.001807938 0.6931472 0.001253167
#> 5      NYT   percent  43 0.001578735 0.6931472 0.001094296
#> 6      NYT  courtney  40 0.001468591 0.6931472 0.001017949
```

The top 10 tf_idf words for each journal were then extracted and plotted. However, because of ties in the tf_idf index of two words from the Guardian journal, the top 11 tf_idf words were extracted and plotted. The R code and output is illustrated below:

```
## extract the top 11 tf_idf words for each journal
top_idf <- df_tfidf %>%
  group_by(journal) %>%
  arrange(desc(tf_idf)) %>%
  slice_head(n = 11)

## plot the extracted top 11 tf_idf words for each journal
ggplot(top_idf,
       aes(x = reorder(word, n), y = tf_idf, fill = journal, group = journal)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(title = "Highest tf_idf Words in Ukraine war articles in 2022", x = "",
       y = "tf_idf index") +
  facet_wrap(~ journal, scales = "free_y") +
  theme(legend.position = "none",
        strip.text = element_text(color = "black", hjust = 0.5, size = 10),
        strip.background = element_rect(fill = "thistle", linetype = "solid",
                                        color = "black", linewidth = 0.5),
        panel.border = element_rect(fill = "transparent",
                                    color = "black", linewidth = 0.5))
```



Conclusively, when identifying the words with the highest tf_idf scores in a journal, we are essentially pinpointing the terms that are both frequent in that specific journal and relatively rare in the overall corpus. These terms can provide insights into the unique and important themes or topics of the journal. The higher the tf_idf score, the more significant the term is within the document. It signifies that the term is not only present frequently but is also distinctive to the document.

As seen from the graph above, words like **Zelenskiy, defence, UK, BST, EU, UN, Guardian, weekday, and edition** are the most significant words which make up the themes of the Guardian journal. On the contrast, words such as **Zhuhan, percent, courtney, sashko, stein, EU, gay, WNBA, erdogan, and neighboring** are the most distinctive words specific to NYT journal.

Linear regression model and its application to describe Zipf's law

Introduction to Linear Regression

- Linear regression is a statistical modeling technique used to establish a relationship between a dependent variable and one or more independent variables.
- In the context of the Ukraine Conflict articles data, linear regression can be applied to understand how rank order may influence the word frequency in the articles.

Linear Regression Model for Zipf's Law

- Zipf's law is an empirical law stating that the frequency of a word is inversely proportional to its rank within a group or corpus of documents.Thus the most frequent word will occur approximately twice as often as the second most frequent word, three times as often as the third most frequent word, etc.
- Linear regression can be employed to model and analyze the relationship between word frequency (dependent variable) and word rank (independent variable) in the Ukraine Conflict articles data. The model aims to fit a straight line that best represents the observed distribution of word frequencies according to Zipf's law.

Application to Ukraine Conflict Articles Data

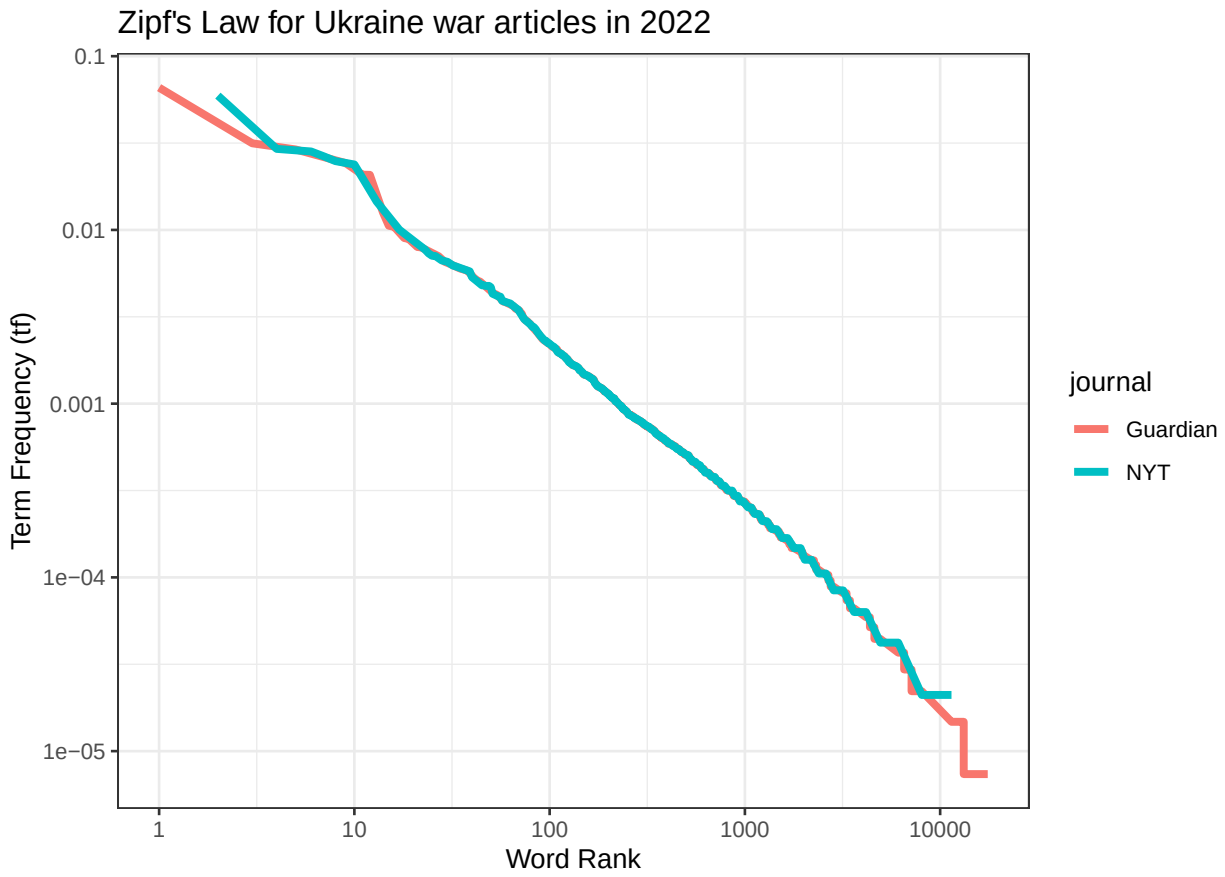
- In this project, the Ukraine Conflict articles data serves as the basis for the application of the linear regression model to describe Zipf's law.
- Word frequency data and corresponding ranks from the articles are utilized to perform the regression analysis.
- The linear regression model helps quantify the degree to which the observed word frequencies adhere to Zipf's law and provides insights into the specific characteristics of the language used in the context of the Ukraine Conflict.

Illustration of R code for Zipf's law

Below is the R code that illustrates Zipf's law.

- First, the articles were tokenized, but, this time, stopwords are not removed since common words are important to illustrate Zipf's law.
- Then, the term frequency (or tf) can be easily calculated using the bind_tf_idf function.
- The rank can be obtained as the row number, after sorting the data in decreasing order of tf and grouping the data per journal.
- The Zipf's law was then illustrated by plotting the Ukraine Conflict articles data on a log_log graph, with log(term frequency) on the vertical axis and log(rank order) on the horizontal axis.

```r
## Create a copy of the joint dataset
copy_df <- merged_data

## remove whitespace and numbers from the articles
copy_df$articles <- stripWhitespace(copy_df$articles)
copy_df$articles <- removeNumbers(copy_df$articles)

## tokenize the articles
copy_df_2 <- copy_df %>%
  unnest_tokens(word, articles)

## calculates the tf indexes
tf_rank <- copy_df_2 %>%
  count(journal, word) %>%
  bind_tf_idf(word, journal, n) %>%
  arrange(desc(tf)) %>% # arrange the words in order of term frequency
  mutate(rank = row_number()) # create a rank variable from the row number

## plot the the data on a log_log graph to illustrates Zipf's law
ggplot(tf_rank, aes(x = rank, y = tf,
                    colour = journal, group = journal)) +
  geom_line(linewidth = 1.5) +
  scale_x_continuous(trans = "log10", breaks = 10^(0:4), labels = 10^(0:4)) +
  scale_y_continuous(trans = "log10", breaks = 10^(-5:1), labels = 10^(-5:1)) +
  labs(title = "Zipf's Law for Ukraine war articles in 2022",
       x = "Word Rank", y = "Term Frequency (tf)") +
  theme_bw()
```



Conclusion from the graph

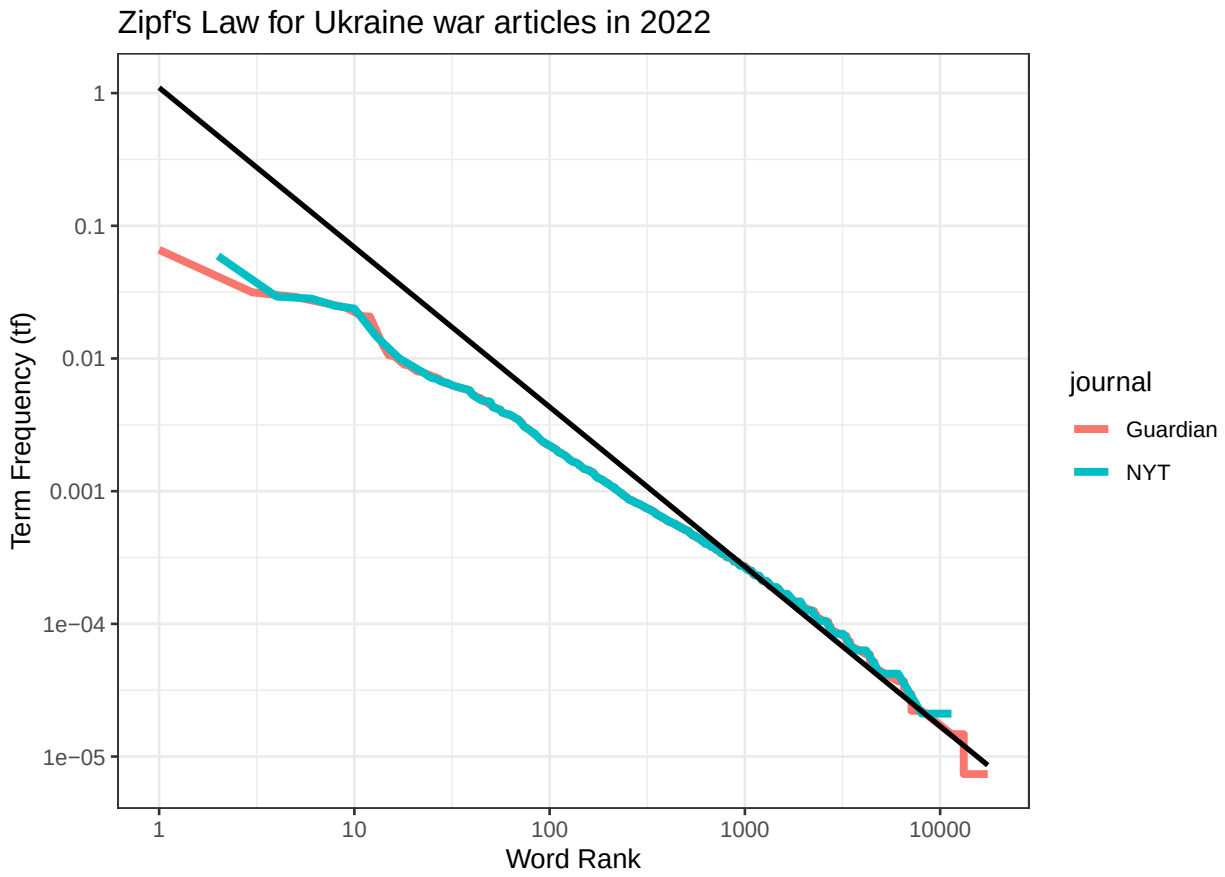
- In both Guardian and NYT journals, the word frequency distribution adheres closely to Zipf's law.
- High-frequency words dominate the rank_size relationship, with a predictable inverse relationship between word frequency and rank.
- The linguistic patterns in both journals exhibit consistency, emphasizing a similar structure in the distribution of word frequencies.
- The observed adherence to Zipf's law suggests a commonality in language use across the two journals.
- The fact that both journals exhibit a clear adherence to Zipf's law indicates that this model effectively characterizes the distribution of word frequencies in their articles.
- The uniform application of Zipf's law in both journals allows for generalizations about the rank-size relationship and word frequency distribution within the context of Ukraine conflicts articles.

Applying the Linear Regression Model to Zipf's law

We apply the linear regression model to describe the effect of the log(rank order) to the log(term frequency) visualized by Zipf's law earlier.

The R code below illustrates the use of **geom_smooth(method="lm")** to generate a linear regression model and fit the regression line on the Zipf's law plot produced earlier.

```
## Estimate the linear regression model
lm_model <- lm(log10(tf) ~ log10(rank), data = tf_rank)

## Extract the coefficients of the model
intercept <- coef(lm_model)[1]
slope <- coef(lm_model)[2]

## Print the coefficients
print(paste("Intercept = ", intercept))
#> [1] "Intercept =  0.0412390373999482"
print(paste("Slope = ", slope))
#> [1] "Slope =  -1.20325421908 53"

## apply regression model to the plot
ggplot(tf_rank, aes(x = rank, y = tf,
                    colour = journal, group = journal)) +
  geom_line(linewidth = 1.5) +
  geom_smooth(data = tf_rank, aes(x = rank, y = tf),
              method = "lm", se = FALSE, inherit.aes = FALSE, colour = "black") +
  scale_x_continuous(trans = "log10", breaks = 10^(0:4), labels = 10^(0:4)) +
  scale_y_continuous(trans = "log10", breaks = 10^(-5:1), labels = 10^(-5:1)) +
  labs(title = "Zipf's Law for Ukraine war articles in 2022",
       x = "Word Rank", y = "Term Frequency (tf)") +
  theme_bw()
```



Comments on the result of the regression model fitted on Zipf's law

- A regression model line was fitted to capture the relationship between word frequency and rank in accordance with Zipf's law for both Guardian and NYT journals.
- The fitted regression line in both journals give a good fit which demonstrate a strong alignment with Zipf's law, indicating that the observed word frequency distributions closely follow the expected inverse relationship with rank.
- The coefficients of the regression model quantify the strength and direction of the relationship between word frequency and rank, providing a numerical representation of how well Zipf's law characterizes the data in both journals. The slope of the model gives a value of **negative 1.20**. This indicated that for every one unit increase in the rank order, the term frequency is expected to decrease by more than one unit.
- This signifies a more rapid decline in frequency for less frequent words as we move down the rank.
- The regression model enhances the interpretability of Zipf's law, allowing for predictions and a clearer understanding of how changes in word rank correspond to changes in word frequency within the context of both journals.