data_analysis_pr

2025-05-08

```r
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.1     ✔ tibble    3.2.1
## ✔ lubridate 1.9.4     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(RColorBrewer)
library(ggbreak)
```

```
## Warning: пакет 'ggbreak' был собран под R версии 4.4.3
```

```
## ggbreak v0.1.4 Learn more at https://yulab-smu.top/
##
##
## If you use ggbreak in published research, please cite the following
## paper:
##
## S Xu, M Chen, T Feng, L Zhan, L Zhou, G Yu. Use ggbreak to effectively
## utilize plotting space to deal with large datasets and outliers.
## Frontiers in Genetics. 2021, 12:774846. doi: 10.3389/fgene.2021.774846
```

```r
library(vcd)
```

```
## Warning: пакет 'vcd' был собран под R версии 4.4.3
```

```
## Загрузка требуемого пакета: grid
```

```r
library(FSA)
```

```
## Warning: пакет 'FSA' был собран под R версии 4.4.3
```

```
## ## FSA v0.10.0. See citation('FSA') if used in publication.
## ## Run fishR() for related website and fishR('IFAR') for related book.
```

```r
library(nnet)
```

```
## Warning: пакет 'nnet' был собран под R версии 4.4.3
```

```r
library(effects)
```

```
## Warning: пакет 'effects' был собран под R версии 4.4.3
```

```
## Загрузка требуемого пакета: carData
```

```
## Warning: пакет 'carData' был собран под R версии 4.4.3
```

```
## lattice theme set by effectsTheme()
## See ?effectsTheme for details.
```

```r
library(ggeffects)
```

```
## Warning: пакет 'ggeffects' был собран под R версии 4.4.3
```

Introduction

In the world of fast-developing technologies, where the Internet has become the main source of communication and entertainment, people find it easier to express their opinion and believes openly, hidden behind the anonymity that the Internet provides. They express not only positive, but also negative attitude on open forums, on social media. It becomes increasingly harder to filter those comments to create a safe environment as there are a lot of ways to express the toxicity. For this purpose we created Rutoxicon to make the Internet a better place.

Rutoxicon is the corpus of toxic comments gathered from Internet forums such as Pikabu and Dvach. For this corpus the comments were collected and annotated manually by following criteria: - **text**: the text of the comment itself;

- **tox**: the sentence or a large phrase that contains toxic message;
- **tox_rate**: the rate of toxicity - how offensive and cruel the comment is based on the scale from 1 to 10, 1 being the lowest grade and 10 the highest, referring to the most insulting, hurtful comments;
- **response**: on what the toxic comment was written - author, person (as the commentator) or post, where in the post we also indicate was it a person or an object;
- **phrase**: the minimal toxic phrase;
- **phrase_types**: type of the phrase based on the way toxicity was explicited: direct and indirect - metaphors;
- **lex**: the toxic lexeme itself;
- **lex_counts**: the amount of lexemes in the toxic phrase;
- **Pos**: part of speech of the lexeme taken from Mystem;

In this research we would like to focus on the toxic lexemes. Russian language is known for its large variety of obscene words, thanks to well-developed morphology and word formation. In our research we would like to find out what parts of speech function as the source of toxicity in the speech more often than others. Also, we would like to analyse the specific in the usage of different types of toxicity.

Data

```
tox_data <- read_csv('rutoxicon_pos1.csv')
```

```
## Rows: 1883 Columns: 11
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (8): text, tox, response, tox_type, phrases, phrase_types, lex, Pos
## dbl (3): text_id, tox_rate, lex_counts
##
## 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.
```

```
head(tox_data)
```

```
## # A tibble: 6 × 11
##   text_id text       tox   tox_rate response tox_type phrases phrase_types lex
##     <dbl> <chr>      <chr>    <dbl> <chr>    <chr>    <chr>   <chr>        <chr>
## 1       0 "Кармодро… "Кар…        4 post: a… general… Кармод… direct       карм…
## 2       1 "Да блять… "Да …        5 post: a… profani… Да бля… direct       блять
## 3       1 "Да блять… "Да …        5 post: a… profani… Это до… direct       дохуя
## 4       1 "Да блять… "Да …        5 post: a… profani… Так чт… direct       ахуе…
## 5       2 "Вы там о… "Вы …        8 post: i… profani… Вы там… direct       охуе…
## 6       2 "Вы там о… "Вы …        8 post: i… profani… Это гд… direct       блять
## # ℹ 2 more variables: lex_counts <dbl>, Pos <chr>
```

```
tox_data$response <- as.factor(tox_data$response)
tox_data$tox_type <- as.factor(tox_data$tox_type)
tox_data$phrase_types <- as.factor(tox_data$phrase_types)
tox_data$Pos <- as.factor(tox_data$Pos)
summary(tox_data)
```

```
##     text_id          text               tox               tox_rate
##  Min.   :   0.0   Length:1883        Length:1883        Min.   : 1.000
##  1st Qu.: 277.5   Class :character   Class :character   1st Qu.: 5.000
##  Median : 519.0   Mode  :character   Mode  :character   Median : 7.000
##  Mean   : 532.9                                         Mean   : 6.678
##  3rd Qu.: 799.5                                         3rd Qu.: 8.000
##  Max.   :1097.0                                         Max.   :10.000
##
##             response                        tox_type      phrases
##  author         : 335   general_insult          :948   Length:1883
##  person         : 437   hate_speech: gender     :331   Class :character
##  post: animate  :1012   hate_speech: nationality:304   Mode  :character
##  post: inanimate:  99   profanity               :104
##                         threat                  : 87
##                         harassment              : 45
##                         (Other)                 : 64
##    phrase_types      lex              lex_counts         Pos
##  direct  :1276   Length:1883        Min.   :1.000   S      :1177
##  indirect: 607   Class :character   1st Qu.:1.000   V      : 357
##                  Mode  :character   Median :2.000   A      : 256
##                                     Mean   :1.686   ADV    :  61
##                                     3rd Qu.:2.000   INTJ   :  13
##                                     Max.   :5.000   (Other):   3
##                                                     NA's   :  16
```

```
sum(is.na(tox_data))
```

```
## [1] 16
```

For the sake of further analysis we transfered some columns like: type of response, type of toxicity, part of speech, type of phrase into factors. All in all, our dataset consists of 1097 unique sentences and 1883 lexemes. AS the annotation of parts of speech was done with the help of the library Mystem on Python, not all of the words have part of speech, for example different abbreviations*(ТП, РСП)*.

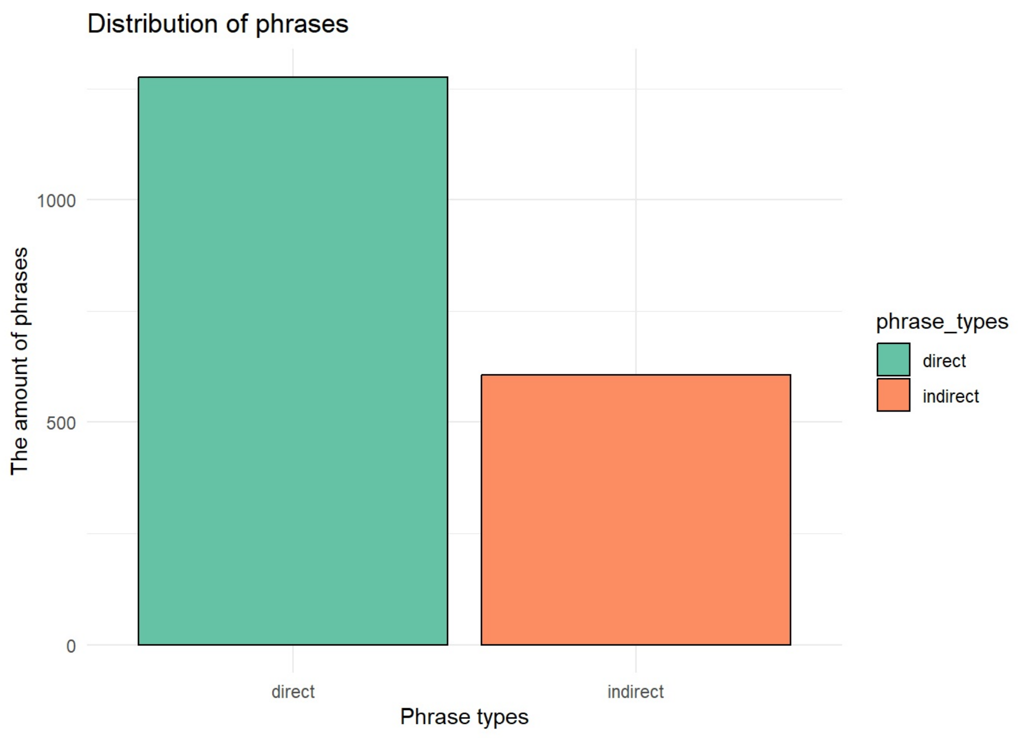
As we see now, our classes are not quite balanced, the mean toxic rate is 6.678, which is quite high as scores 6-7 usually describe increased toxicity (implicit threats, unacceptable hints) and the median value of toxicity rate is 7, even higher than the mean value.

```
glimpse(tox_data)
```

```
## Rows: 1,883
## Columns: 11
## $ text_id      <dbl> 0, 1, 1, 1, 2, 2, 3, 4, 5, 5, 6, 6, 6, 7, 8, 8, 8, 8, 9, …
## $ text         <chr> "Кармодрочеры", "Да блять !  Он не взламывал игры, а сим…
## $ tox          <chr> "Кармодрочеры", "Да блять !  Он не взламывал игры, а сим…
## $ tox_rate     <dbl> 4, 5, 5, 5, 8, 8, 5, 1, 7, 7, 5, 5, 5, 6, 8, 8, 8, 8, 6, …
## $ response     <fct> post: animate, post: animate, post: animate, post: animat…
## $ tox_type     <fct> general_insult, profanity, profanity, profanity, profanit…
## $ phrases      <chr> "Кармодрочеры", "Да блять !", "Это дохуя разные вещи", "Т…
## $ phrase_types <fct> direct, direct, direct, direct, direct, direct, direct, d…
## $ lex          <chr> "кармодрочеры", "блять", "дохуя", "ахуенной", "охуели", "…
## $ lex_counts   <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1, 2, 2, 2, 2, 2, …
## $ Pos          <fct> S, V, V, A, V, V, V, S, S, S, S, S, V, S, V, S, A, S, A, …
```

```
tox_data%>%
  ggplot(aes(phrase_types, fill=phrase_types))+
  geom_histogram(stat="count", color = 'black')+
  xlab('Phrase types')+
  ylab('The amount of phrases')+
  ggtitle('Distribution of phrases')+
  theme_minimal()+
  scale_fill_brewer(palette = "Set2")
```

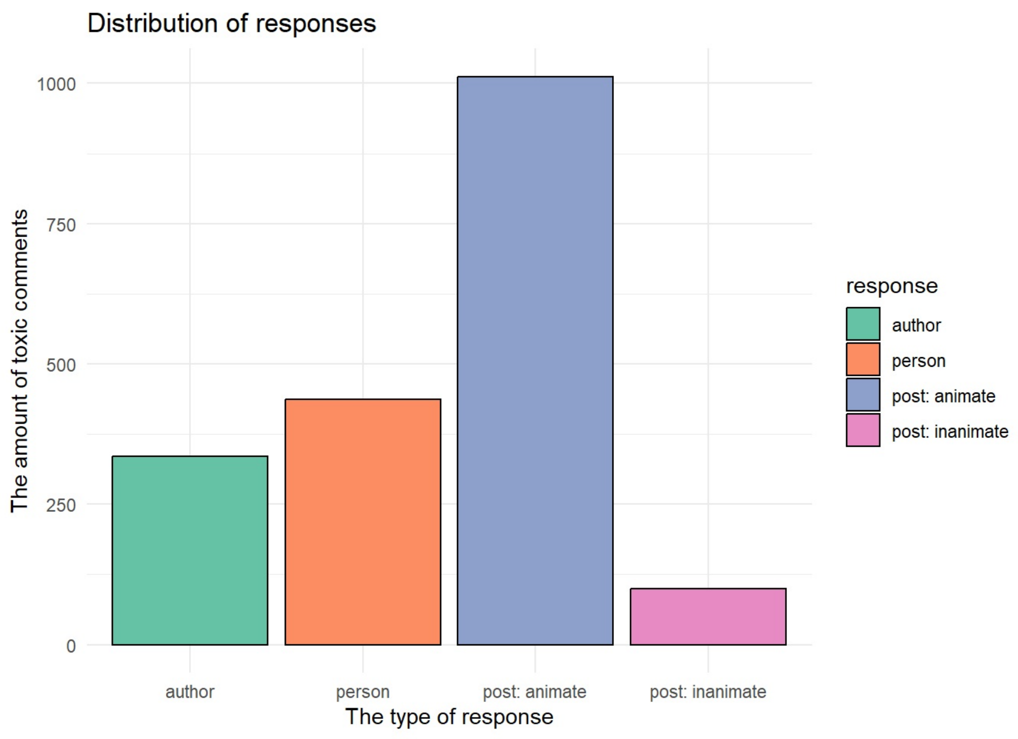
```
## Warning in geom_histogram(stat = "count", color = "black"): Ignoring unknown
## parameters: `binwidth`, `bins`, and `pad`
```



The following graphic depicts the proportion of different classes of phrases. We can see there is a disbalance of classes, indirect group being two times smaller than direct group. The probable explanation for such proportion lies partially behind the specifics of this research, we used only phrases that contained at least 1 word, that is why the amount of indirect phrases was reduced. Another reason for such proportion would be the uniqueness and creativeness required from the speaker in order to create metaphoric and other types of tropes.

```
tox_data%>%
  ggplot(aes(response, fill=response))+
  geom_histogram(stat="count", color = "black")+
  theme_minimal()+
  xlab('The type of response')+
  ylab('The amount of toxic comments')+
  ggtitle('Distribution of responses')+
  scale_fill_brewer(palette = "Set2")
```

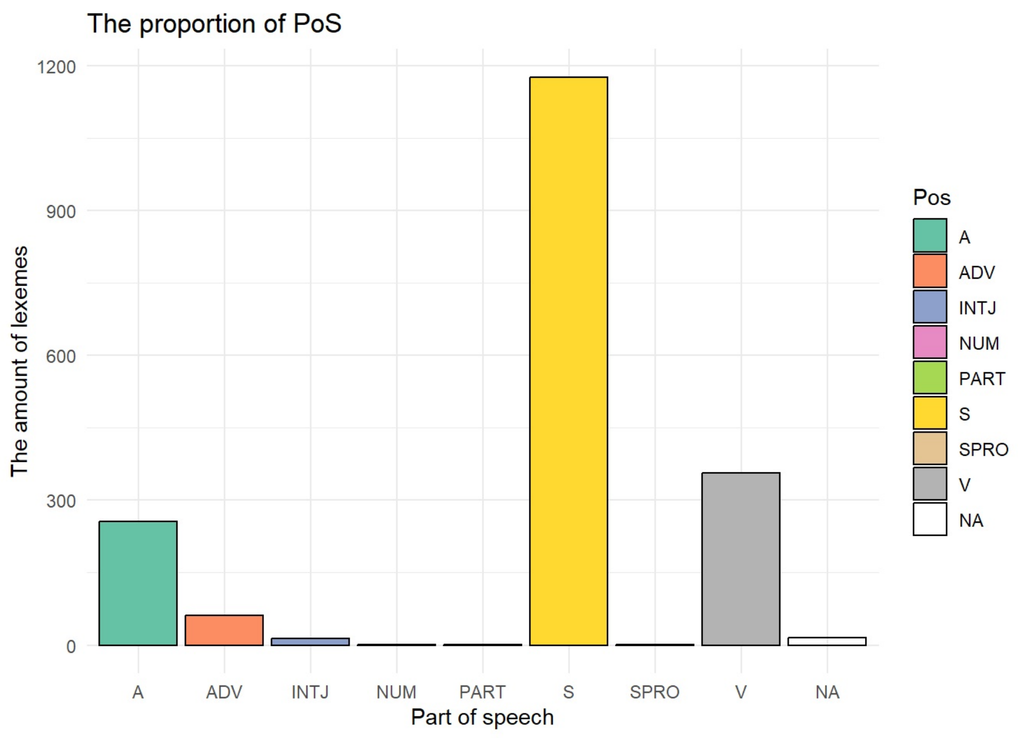
```
## Warning in geom_histogram(stat = "count", color = "black"): Ignoring unknown
## parameters: `binwidth`, `bins`, and `pad`
```



In this figure we can see the proportion of the toxic comments according to the type of the response. There is a disbalance in classes, where insults and toxicity towards inanimate objects are less common rather than cruelty towards animate objects. It is visible that people in the Internet tend to express their negativity towards the people described in the post, those who can not react and respond to the hate.

```
tox_data%>%
  ggplot(aes(x=Pos, fill = Pos))+
  geom_histogram(stat='count', color= 'black')+
  ggtitle('The proportion of PoS')+
  xlab('Part of speech') +
  ylab('The amount of lexemes')+
  scale_fill_brewer(palette = "Set2")+
  theme_minimal()
```

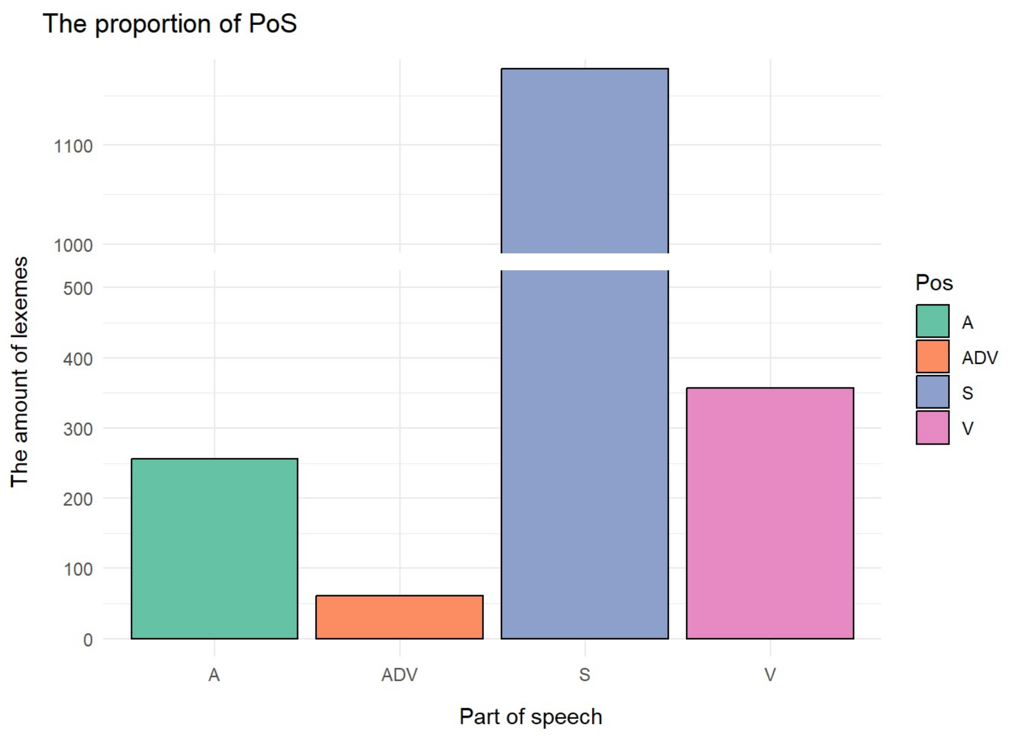
```
## Warning in geom_histogram(stat = "count", color = "black"): Ignoring unknown
## parameters: `binwidth`, `bins`, and `pad`
```



In this figure the proportion of parts of speech is presented. As we see some parts of speech are barely presented, so for the sake of better visualization, we are going to filter parts of speech whose occurrence is less than 20.

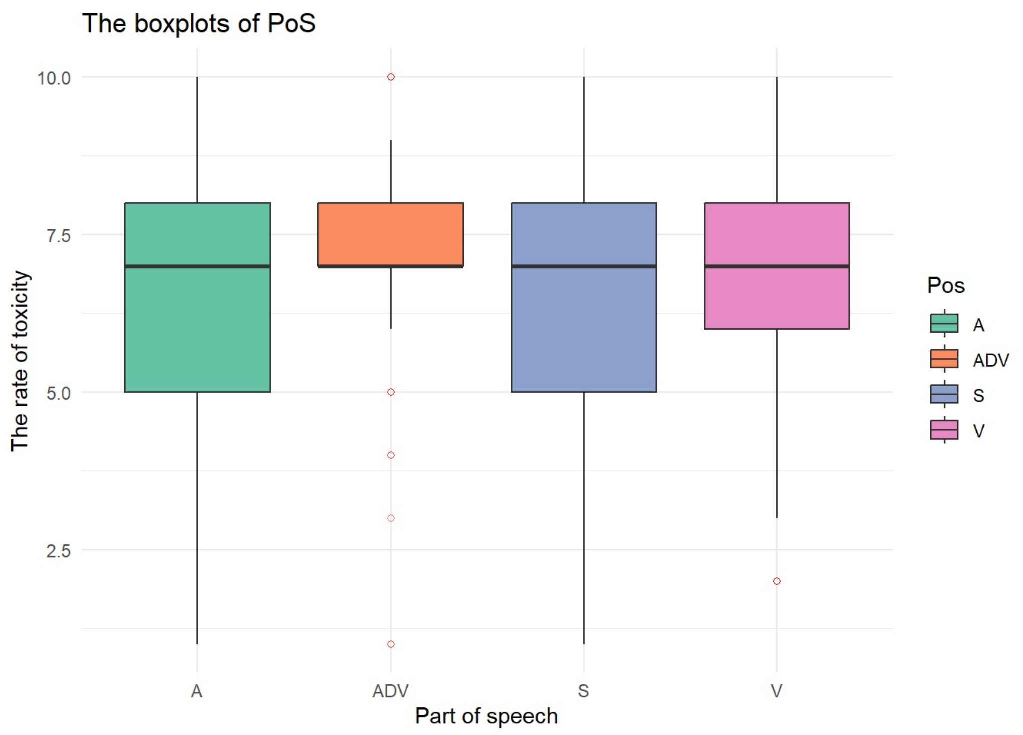
```
tox_data%>%
  group_by(Pos)%>%
  filter(n()>20)%>%
  ggplot(aes(x=Pos, fill = Pos))+
  geom_histogram(stat='count', color= 'black')+
  scale_y_break(c(500, 1000), scales = 0.5, ticklabels = c(seq(0, 500, by=50), seq(1000, 1500, by=100)), space =
0.2)+
  ggtitle('The proportion of PoS')+
  xlab('Part of speech') +
  ylab('The amount of lexemes')+
  scale_fill_brewer(palette = "Set2")+
  theme_minimal()
```

```
## Warning in geom_histogram(stat = "count", color = "black"): Ignoring unknown
## parameters: `binwidth`, `bins`, and `pad`
```



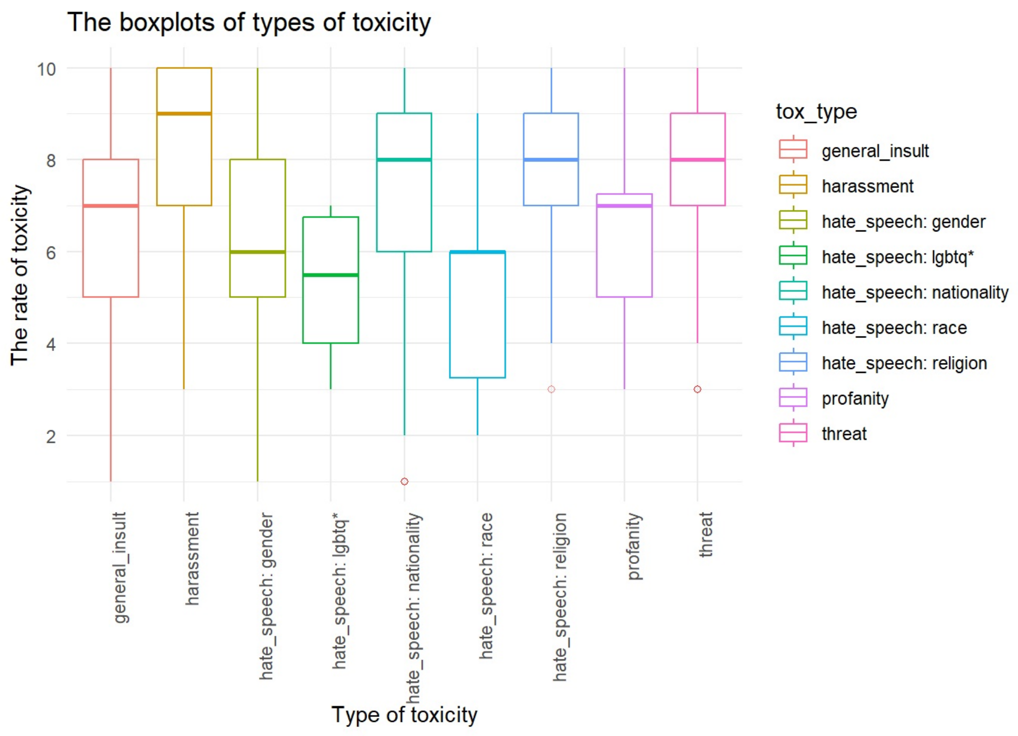
Now it is easier to see that the difference between the classes. Nouns present the largest group of the toxic lexemes with more than 1100 words. Such difference could be explained by the large paradig of words formation in the Russian language and the speakers' desires to hide the obscene words by changing 1-2 letters or using a homonyms. Verbs are the second largest class with less than 400 toxic words, the third class is presented by adjectives, with less than 300 words, as they are usually used in noun phrases to increase the expressiveness of the phrase.

```
tox_data %>%
  group_by(Pos) %>%
  filter(n()>20)%>%
  ggplot(aes(x = Pos, y = tox_rate, fill = Pos)) +
  geom_boxplot(
    outlier.colour = "red",
    outlier.shape = 1,
    outlier.alpha = 0.5) +
  ggtitle('The boxplots of PoS')+
  xlab('Part of speech') +
  ylab('The rate of toxicity')+
  scale_fill_brewer(palette = "Set2")+
  theme_minimal()
```



The figure above depicts the boxplots of parts of speech. As we see the verbs and the adverbs are used in the sentences with higher toxicity rate in comparison to adjectives and nouns, only outliers are found among low values. Though it is very important to note that adverbs are not particularly common in extremely toxic comments. The boxplots of adjectives and nouns are rather similar which could be explained by the usage of adjectives in noun phrases.

```
tox_data %>%
  ggplot(aes(x=tox_type, y = tox_rate))+
  geom_boxplot(aes(color=tox_type), outlier.colour = "red", outlier.shape = 1, outlier.alpha = 0.5)+
  ggtitle('The boxplots of types of toxicity')+
  xlab('Type of toxicity')+
  ylab('The rate of toxicity')+
  scale_y_continuous(breaks =c(0,2,4,6,8,10))+
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

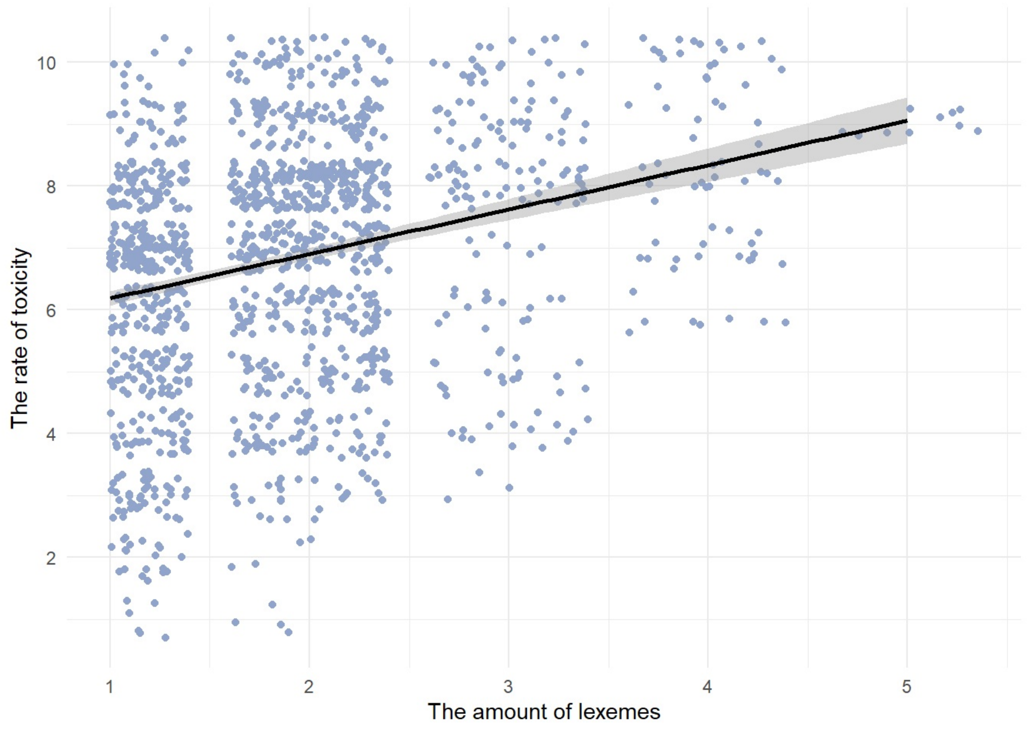


The figure above depicts the toxic rate dependence on the type of toxicity. The toxicity type, *harassment*, is shown to be one of the most toxic, which could be explained by the harsh and mostly obscene language used in these comments. Among other groups with high rate of toxicity are *hate_speech: nationality*, *hate_speech: religion* and *threat*. The least toxic group here is *hate_speech: race* as it is not quite of the current interest among the users of the Internet forums.

```
tox_data%>%
  ggplot(aes(x=lex_counts, y=tox_rate))+
  geom_jitter(stat = "identity", colour='#90a4cc')+
  scale_y_continuous(breaks=c(2, 4, 6, 8, 10))+
  #scale_x_continuous(breaks=c(1, 2, 3, 4, 5))+
  scale_x_continuous(breaks = c(1, 2, 3, 4, 5), limits = c(1, NA))+
  geom_smooth(method = "lm", color = "black", se = TRUE, fullrange = TRUE)+
  xlab('The amount of lexemes')+
  ylab('The rate of toxicity')+
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 456 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

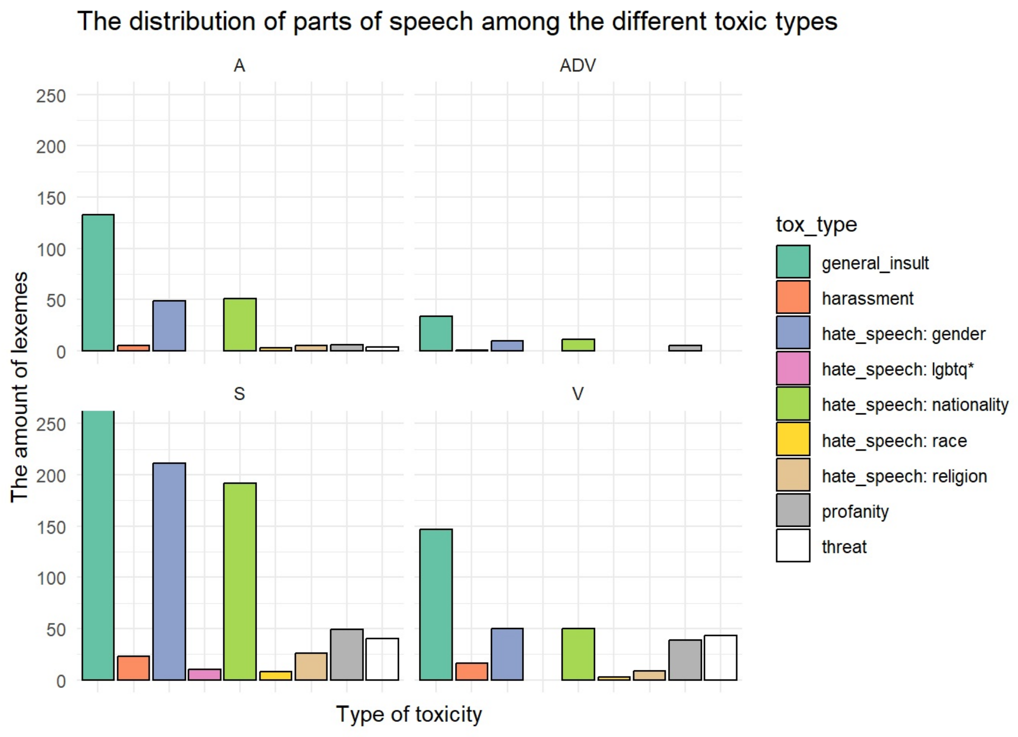


The following figure shows the relationship between the amount of lexemes and the rate of toxicity. As we can see, the more toxic words appear in the sentence, the higher the probability of the sentence to be count extremly toxic (8-10).

```
tox_data %>%
  group_by(Pos) %>%
  filter(n()>20)%>%
  ggplot(aes(x=tox_type, fill = tox_type))+
  geom_histogram(stat='count', color= 'black')+
  theme_minimal()+
  ggtitle('The distribution of parts of speech among the different toxic types')+
  xlab('Type of toxicity')+
  ylab('The amount of lexemes')+
  scale_fill_brewer(palette = "Set2")+
  theme(axis.text.x = element_blank())+
  facet_wrap(~Pos)+
  coord_cartesian(ylim = c(0, 250))
```

```
## Warning in geom_histogram(stat = "count", color = "black"): Ignoring unknown
## parameters: `binwidth`, `bins`, and `pad`
```

```
## Warning in RColorBrewer::brewer.pal(n, pal): n too large, allowed maximum for palette Set2 is 8
## Returning the palette you asked for with that many colors
```



This figure illustrates the distribution of parts of speech across different types of toxic language. For the purpose of better illustrating the y-axis was cut to the 250, because of the big amount of nouns which was mentioned above. Nouns appear to be the only type of part of speech that are presented in every type of toxicity, even in *hate_speech: lgbtq* where other parts of speech are not found. Among all parts of speech such types as *general insult*, *hate_speech: gender*, *hate_speech: nationality* are presented. It is interesting to note that even though there are almost 3 times less verb, their amount in the comments involving *threats* is higher than for nouns, and in comments with *profanity* is just tiny less. Also, it is important to highlight how different distributions of adverbs is in comparison to other parts of speech. Adverbs appear to be a very small group, used only in common types of toxicity.

Data Analysis

Chi-square 1

For testing our future hypotheses we need to verify how our data is distributed as it could affect the methods that we are going to use. In order to check whether our data falls into normal distribution or not, we decided to use Shapiro-Wilcox test. Our null hypothesis is that the data is distributed normally.

```
shapiro.test(tox_data$tox_rate)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  tox_data$tox_rate
## W = 0.94896, p-value < 2.2e-16
```

The results of the Shapiro test (W = 0.94896, p-value < 2.2e-16) test showed that on the interval of 99% we could reject our null hypothesis of data being distributed normally, as such in further analysis we have to be careful with the results of the tests and use the alternative methods of testing.

As the toxic rate is discrete variable and type of toxicity is a categorical variable, we are going to use chi-squared test. But in order to do it, we have to know that at least 80% of expected values should be more than 5. There are following hypothesis that we are going to check:

H0: Type of toxic comment has no influence on the toxic rate of the comment

H1: Type of toxic comment has influence on the toxic rate of the comment

```
tox.tabulated <- tox_data %>%
  select(tox_rate, tox_type) %>%
  table()
tox.tabulated
```

```
##         tox_type
## tox_rate general_insult harassment hate_speech: gender hate_speech: lgbtq*
##       1              10          0                   1                   0
##       2              29          0                   8                   0
##       3              58          2                  25                   2
##       4              75          3                  43                   2
##       5             112          4                  40                   1
##       6             108          0                  49                   2
##       7             208          6                  68                   3
##       8             207          7                  75                   0
##       9              99         10                  20                   0
##       10             42         13                   2                   0
##         tox_type
## tox_rate hate_speech: nationality hate_speech: race hate_speech: religion
##       1                         6                 0                     0
##       2                         3                 1                     0
##       3                        12                 3                     1
##       4                         8                 1                     5
##       5                        24                 0                     0
##       6                        42                 6                     2
##       7                        40                 0                     7
##       8                        88                 1                    14
##       9                        37                 2                     2
##       10                       44                 0                     9
##         tox_type
## tox_rate profanity threat
##       1          0      0
##       2          0      0
##       3          1      5
##       4          8      5
##       5         18      6
##       6         13      3
##       7         38     12
##       8         15     14
##       9          5     24
##       10         6     18
```

As we can see a lot of classes do not have all the presented ratings scores. So, in order to calculate the chi-score correctly we are going to delete certain columns: hate_speech: race, hate_speech: religion, harassment, and the rows with low toxicity rate.

```
tox.tab_short <- tox.tabulated[-c(1,2,3), -c(2, 4, 6, 7)]
tox.tab_short
```

```
##         tox_type
## tox_rate general_insult hate_speech: gender hate_speech: nationality profanity
##       4              75                  43                        8         8
##       5             112                  40                       24        18
##       6             108                  49                       42        13
##       7             208                  68                       40        38
##       8             207                  75                       88        15
##       9              99                  20                       37         5
##       10             42                   2                       44         6
##         tox_type
## tox_rate threat
##       4       5
##       5       6
##       6       3
##       7      12
##       8      14
##       9      24
##       10     18
```

After removing the disturbing columns the table looks better. Now let’s compare the results of chi-square for both tables.

```
tox.chisq <- chisq.test(tox.tabulated)
```

```
## Warning in chisq.test(tox.tabulated): аппроксимация на основе хи-квадрат может
## быть неправильной
```

```
tox.chisq
```

```
##
##  Pearson's Chi-squared test
##
## data:  tox.tabulated
## X-squared = 316.29, df = 72, p-value < 2.2e-16
```

```
round(tox.chisq$expected, 1)
```

```
##         tox_type
## tox_rate general_insult harassment hate_speech: gender hate_speech: lgbtq*
##       1             8.6        0.4                 3.0                 0.1
##       2            20.6        1.0                 7.2                 0.2
##       3            54.9        2.6                19.2                 0.6
##       4            75.5        3.6                26.4                 0.8
##       5           103.2        4.9                36.0                 1.1
##       6           113.3        5.4                39.6                 1.2
##       7           192.3        9.1                67.1                 2.0
##       8           212.0       10.1                74.0                 2.2
##       9           100.2        4.8                35.0                 1.1
##       10           67.5        3.2                23.6                 0.7
##         tox_type
## tox_rate hate_speech: nationality hate_speech: race hate_speech: religion
##       1                       2.7               0.1                   0.4
##       2                       6.6               0.3                   0.9
##       3                      17.6               0.8                   2.3
##       4                      24.2               1.1                   3.2
##       5                      33.1               1.5                   4.4
##       6                      36.3               1.7                   4.8
##       7                      61.7               2.8                   8.1
##       8                      68.0               3.1                   8.9
##       9                      32.1               1.5                   4.2
##       10                     21.6               1.0                   2.8
##         tox_type
## tox_rate profanity threat
##       1        0.9    0.8
##       2        2.3    1.9
##       3        6.0    5.0
##       4        8.3    6.9
##       5       11.3    9.5
##       6       12.4   10.4
##       7       21.1   17.6
##       8       23.3   19.5
##       9       11.0    9.2
##       10       7.4    6.2
```

Our concerns about the unsuitability of the first table proved to be correct. In the deleted rows and columns the expected values are less than 5.0 which results in the warning sign, while calculating the chi-square.

```
tox.chisq1 <- chisq.test(tox.tab_short)
tox.chisq1
```

```
##
##  Pearson's Chi-squared test
##
## data:  tox.tab_short
## X-squared = 185.24, df = 24, p-value < 2.2e-16
```

```
round(tox.chisq1$expected, 1)
```

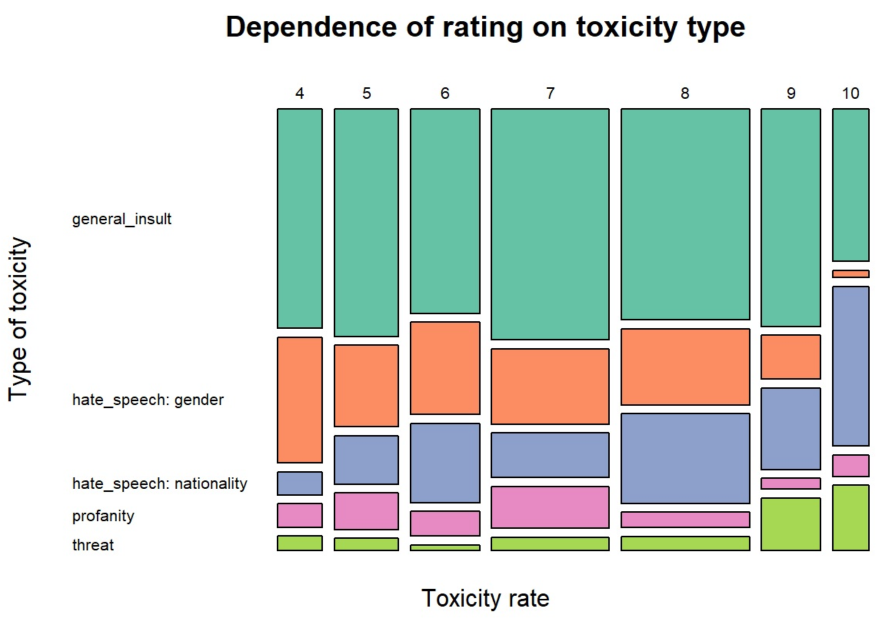
```
##         tox_type
## tox_rate general_insult hate_speech: gender hate_speech: nationality profanity
##       4           73.2                25.5                     24.3       8.9
##       5          105.3                36.8                     35.0      12.7
##       6          113.2                39.5                     37.7      13.7
##       7          192.7                67.3                     64.1      23.3
##       8          210.1                73.3                     69.9      25.4
##       9           97.4                34.0                     32.4      11.8
##       10          59.0                20.6                     19.6       7.1
##         tox_type
## tox_rate threat
##       4     7.1
##       5    10.1
##       6    10.9
##       7    18.6
##       8    20.2
##       9     9.4
##       10    5.7
```

When it comes to the new table, the expected values are higher than 5.0, so we could trust the results of the chi-square.

Our results (X-squared = 185.24, df = 24, p-value < 2.2e-16) allow us to reject the null hypothesis about the type of toxic comment having no influence on the toxic rate of the comment as p-value is less than 1%. From this follows that certain types of toxic comment tend to have certain rating scores, for example “profanity” can be usually interpret as middle-toxicity, having the toxic ratings around 5-8.

```
colours <- brewer.pal(n = ncol(tox.tab_short), name = 'Set2')
mosaicplot(tox.tab_short,main="Dependence of rating on toxicity type",
           legend = TRUE,
           xlab = 'Toxicity rate',
          ylab = 'Type of toxicity',
          col = colours,
          las = 1)
```

```
## Warning: В mosaicplot.default(tox.tab_short, main = "Dependence of rating on toxicity type",
##     legend = TRUE, xlab = "Toxicity rate", ylab = "Type of toxicity",
##     col = colours, las = 1) :
##  дополнительный аргумент 'legend' не будет учтен
```



This figure depicts that the types of toxicity like *hate_speech: nationality* and *threat* tend to be interpreted as highly rude and offensive with scores 8-10. This could be explained by the cruelness of these comments, in which calls for death and violence can be found. *Profanity* is mainly used in the comments with middle levels of toxicity, as it is more about expliciting emotions like annoyance and shock rather than insulting and threats towards a person or situation.

Mann–Whitney U test

Here we would like to test the following hypotheses:

H0: Number of words has no influence on the toxic rate of the comment

H1: Number of words has influence on the toxic rate of the comment

As the toxic rate is a discrete variable and number of words is also a discrete variable, we are going to use Welch’s t-test or Mann–Whitney U test. But in order to do it, we have to check whether our values are distributed normally.

```
shapiro_test1 <- shapiro.test(tox_data$lex_counts)
shapiro_test2 <- shapiro.test(tox_data$tox_rate)
shapiro_test1
```

```
##
##  Shapiro-Wilk normality test
##
## data:  tox_data$lex_counts
## W = 0.75728, p-value < 2.2e-16
```

```
shapiro_test2
```

```
##
##  Shapiro-Wilk normality test
##
## data:  tox_data$tox_rate
## W = 0.94896, p-value < 2.2e-16
```

Both p-values (W = 0.75728, p-value < 2.2e-16; W = 0.94896, p-value < 2.2e-16) are less than 0.05, so we can reject the null hypothesis about the normal distribution of the data. Thus, we are going to use unparametric analog of t-test, Mann–Whitney U test.

```
wilcox_test_result <- wilcox.test(tox_data$lex_counts, tox_data$tox_rate, exact = FALSE)
wilcox_test_result
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  tox_data$lex_counts and tox_data$tox_rate
## W = 73700, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0
```

Our results (W = 73700, p-value < 2.2e-16) allow us to reject the null hypothesis about the amount of toxic words having no influence on the toxic rate of the comment on the 99% interval as the p-value is less than 1%.

Chi-square 2

We would like to test the following hypotheses:

H0: Part of speech has no influence on the number of words

H1: Part of speech has influence on the number of words

As the number of words is discrete variable and part of speech is a categorical variable, we are going to use chi-square test. But in order to do it, we have to know that at least 80% of expected values should be more than 5.0

```
lex.pos <- tox_data %>%
  select(lex_counts, Pos) %>%
  table()
lex.pos
```

```
##           Pos
## lex_counts   A ADV INTJ NUM PART   S SPRO   V
##          1  84  37    4   1    1 595    1 186
##          2 119  22    5   0    0 442    0 135
##          3  38   1    2   0    0  93    0  25
##          4  13   0    2   0    0  42    0   9
##          5   2   1    0   0    0   5    0   2
```

```
lex.pos_short <- lex.pos[-c(5), -c(3,4,5, 7)]
lex.pos_short
```

```
##           Pos
## lex_counts   A ADV   S   V
##          1  84  37 595 186
##          2 119  22 442 135
##          3  38   1  93  25
##          4  13   0  42   9
```

As we already did in the previous chi-square test, we reduced the table by deleting the columns with null values like num, part, spro, intj.

```
pos.chisq <- chisq.test(lex.pos_short)
```

```
## Warning in chisq.test(lex.pos_short): аппроксимация на основе хи-квадрат может
## быть неправильной
```

```
pos.chisq
```

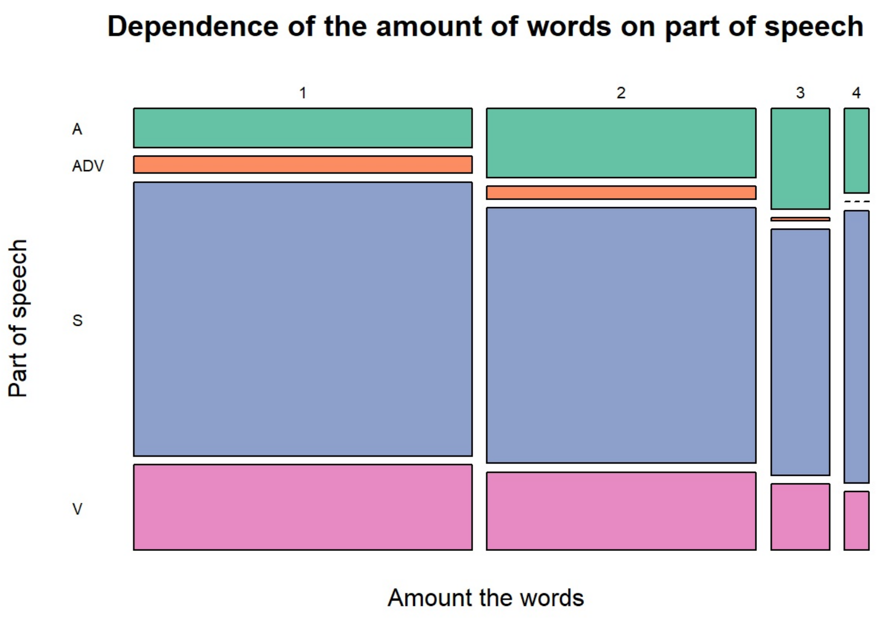
```
##
##  Pearson's Chi-squared test
##
## data:  lex.pos_short
## X-squared = 43.419, df = 9, p-value = 1.805e-06
```

```
round(pos.chisq$expected, 1)
```

```
##           Pos
## lex_counts     A  ADV     S     V
##          1 124.4 29.4 574.2 173.9
##          2  99.1 23.4 457.1 138.5
##          3  21.7  5.1  99.9  30.3
##          4   8.8  2.1  40.7  12.3
```

```
mosaicplot(lex.pos_short,main="Dependence of the amount of words on part of speech",
           legend = TRUE,
           xlab = 'Amount the words',
          ylab = 'Part of speech',
          col = colours,
          las = 1)
```

```
## Warning: В mosaicplot.default(lex.pos_short, main = "Dependence of the amount of words on part of speech",
##     legend = TRUE, xlab = "Amount the words", ylab = "Part of speech",
##     col = colours, las = 1) :
##  дополнительный аргумент 'legend' не будет учтен
```



Our results (X-squared = 43.419, df = 9, p-value = 1.805e-06) allow us to reject the null hypothesis about parts of speech having no influence on the number of words with p-value less than 1%. AS we see adjectives mainly appear in the sentences with 2 or more words, 3 being the most expected value. Nouns as the biggest class are used with any amount of words in the sentences, 1 and 4 being the most significant.

Multi models

One of the hardest part of our analysis was the manual annotation of toxic comments. There are a lot of parameters that influence the toxicity rate, including the perception of the annotator. We would like to see whether it is possible to predict the toxicity rates depending on the annotated data, without any help from human annotator. Also, it was interesting to find what kind of factors have more influence on the toxicity rate.For this purpose we chose to train **multinominal logistic regression**, because our target variable is a discrete variable with more than 2 categories and which could be interpreted as classification.

Firstly, we are going to train the model on all the columns to see which parameters are of use here:

Multimodel 1

```
multi_model <- multinom(tox_rate ~ tox_type + response + phrase_types + lex_counts + Pos, data = tox_data)
```

```
## # weights:  220 (189 variable)
## initial  value 4298.926369 
## iter  10 value 3680.021649
## iter  20 value 3483.727975
## iter  30 value 3451.415307
## iter  40 value 3445.372901
## iter  50 value 3441.877244
## iter  60 value 3440.441496
## iter  70 value 3439.570350
## iter  80 value 3439.243576
## iter  90 value 3439.088027
## iter 100 value 3438.923564
## final  value 3438.923564 
## stopped after 100 iterations
```

```
summary(multi_model)
```

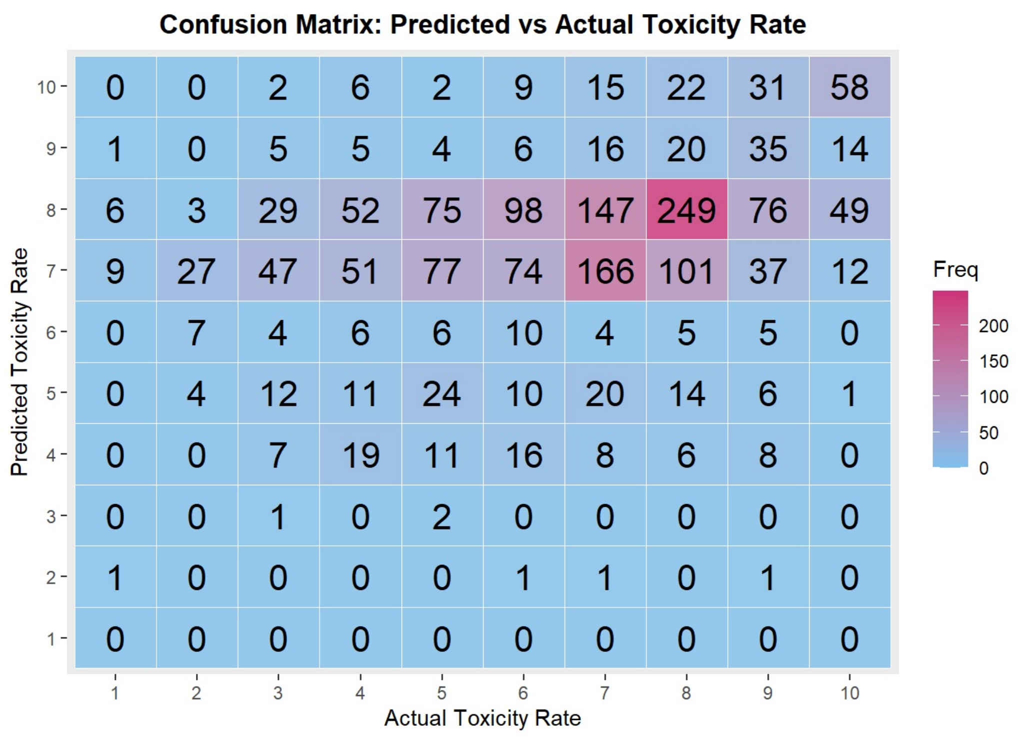
```
## Call:
## multinom(formula = tox_rate ~ tox_type + response + phrase_types + 
##     lex_counts + Pos, data = tox_data)
## 
## Coefficients:
##    (Intercept) tox_typeharassment tox_typehate_speech: gender
## 2   3.40879387          -5.585367                   1.2189367
## 3   2.31636003           2.688359                   1.3771180
## 4   1.57242784           2.741616                   1.4444135
## 5   1.66838851           2.805937                   1.1209346
## 6   1.05741336          -6.075365                   1.2314413
## 7   2.55779087           2.747703                   0.9732729
## 8   1.41712822           2.918834                   0.9395677
## 9  -0.01638656           4.340135                   0.4684312
## 10 -1.43977099           6.211571                  -1.5023777
##    tox_typehate_speech: lgbtq* tox_typehate_speech: nationality
## 2                    -3.021887                       -1.5891848
## 3                     4.347439                       -1.0920606
## 4                     4.037228                       -1.8942752
## 5                     3.055724                       -1.0690132
## 6                     3.802351                       -0.7209276
## 7                     3.599266                       -1.5578650
## 8                    -5.688917                       -0.7684281
## 9                    -4.551292                       -0.7769857
## 10                   -2.449272                       -0.2431739
##    tox_typehate_speech: race tox_typehate_speech: religion tox_typeprofanity
## 2                   4.023878                     -5.086757         -5.103614
## 3                   4.220280                      2.545402          3.404924
## 4                   2.554968                      3.852073          5.335340
## 5                  -5.603124                     -5.981678          5.122536
## 6                   4.111754                      2.360820          4.469037
## 7                  -6.404185                      2.965664          5.067974
## 8                   1.734459                      3.692072          4.278423
## 9                   3.405646                      2.491338          3.514568
## 10                 -4.053664                      4.538810          4.925922
##    tox_typethreat responseperson responsepost: animate responsepost: inanimate
## 2       -5.381333     -0.1071926           -0.56432568               -1.410607
## 3        3.220708     -0.1202448           -0.62752220               -3.781006
## 4        3.086488     -0.1279865           -0.62740744               -4.032546
## 5        2.849453     -0.0113674           -0.52808937               -1.896770
## 6        2.015583      0.1500117           -0.11096107               -1.556899
## 7        2.431123     -0.4931616           -0.09562474               -2.621442
## 8        2.683217     -0.2413265           -0.12711608               -2.793091
## 9        4.205344     -0.1266789            0.01561602               -1.058485
## 10       4.543593     -0.4672166            0.72268909              -13.077269
##    phrase_typesindirect lex_counts      PosADV   PosINTJ      PosNUM
## 2          -0.439894840 -1.0073545 -10.5407979 -3.605774 -1.17892932
## 3          -0.377882678  0.2655633  -2.6281028 -4.399176 -1.43583809
## 4          -0.001287526  1.0410043  -2.2776867  3.774655 -1.95458720
## 5          -0.239379032  0.9003559  -1.8315698  4.384638  9.49242044
## 6          -0.158303255  1.1907955  -1.0409386 -5.130995 -1.62896454
## 7          -1.020382354  0.9406294  -0.9167173  3.982089 -1.37129260
## 8          -0.885216661  1.5285874  -0.6211286  3.533122 -1.21394759
## 9          -0.951539188  1.8527971  -0.7678780 -5.329399 -0.49673783
## 10         -2.803922552  2.2798311  -0.9788785  4.377584  0.00703435
##        PosPART        PosS     PosSPRO     PosV
## 2  -0.05121189 -0.78879911 -0.94239474 4.759274
## 3  -0.34465520 -0.13883001 -1.16524047 4.510267
## 4  -2.67894531 -0.42450998  9.79285846 4.518325
## 5  -2.09291522  0.07146120 -1.40409115 5.100792
## 6   9.70925587 -0.11850521 -1.60363607 5.437062
## 7  -2.82533825 -0.09673473 -2.08801485 5.532162
## 8  -1.24903902 -0.01251043 -1.55424229 5.818341
## 9  -0.23552823 -0.24174588 -0.67691585 5.671088
## 10 -0.09616722 -0.52996163 -0.06052806 5.235363
##
## Std. Errors:
##    (Intercept) tox_typeharassment tox_typehate_speech: gender
## 2     1.475489          62.240368                    1.217844
## 3     1.326095           8.518859                    1.167672
## 4     1.298513           8.509386                    1.158046
## 5     1.291578           8.504044                    1.156813
## 6     1.288274          44.594425                    1.154641
## 7     1.273521           8.500018                    1.149798
## 8     1.272766           8.499118                    1.150233
## 9     1.297005           8.497982                    1.171280
## 10    1.346155           8.501140                    1.384302
##    tox_typehate_speech: lgbtq* tox_typehate_speech: nationality
## 2                     50.14779                        0.9304754
## 3                     16.11321                        0.7442049
## 4                     16.11411                        0.7670672
## 5                     16.12885                        0.7059077
## 6                     16.11364                        0.6929683
## 7                     16.10831                        0.6876043
## 8                     67.32497                        0.6794007
## 9                     66.27055                        0.7010928
## 10                    45.65436                        0.7118290
##    tox_typehate_speech: race tox_typehate_speech: religion tox_typeprofanity
## 2                   12.24807                     56.720740         68.302069
## 3                   12.21731                      7.580895          7.598656
## 4                   12.24583                      7.527911          7.540789
## 5                   56.82322                     49.738612          7.533880
## 6                   12.21100                      7.544967          7.536253
## 7                   59.75641                      7.522229          7.532299
## 8                   12.24710                      7.518951          7.535192
## 9                   12.22819                      7.546753          7.544098
## 10                  48.56905                      7.527830          7.546290
##    tox_typethreat responseperson responsepost: animate responsepost: inanimate
## 2       45.605376      1.0401812             1.0013218               1.1364119
## 3        6.774185      0.9759157             0.9256863               1.4058334
## 4        6.774001      0.9657155             0.9149867               1.2276115
## 5        6.771096      0.9602691             0.9093527               1.0078084
## 6        6.784030      0.9640728             0.9103923               1.0099499
## 7        6.765444      0.9529580             0.8971049               0.9983892
## 8        6.765038      0.9528099             0.8978181               1.0128266
## 9        6.764025      0.9754882             0.9171609               1.0105643
## 10       6.767596      1.0387935             0.9504287              90.0656300
##    phrase_typesindirect lex_counts    PosADV  PosINTJ     PosNUM      PosPART
## 2             0.6121105  0.7504160 61.352377 60.60467 13.2435046 3.227811e-03
## 3             0.5527956  0.5727660  1.463824 66.80631 12.8329407 1.602379e+01
## 4             0.5431710  0.5528121  1.279838 28.00661 11.2193068 1.610428e+01
## 5             0.5340569  0.5496124  1.207766 28.00350 88.8812568 2.010501e+01
## 6             0.5312575  0.5465864  1.119180 67.81116 12.3022118 1.064765e+02
## 7             0.5287286  0.5452946  1.082968 28.00181 12.3391244 1.580923e+01
## 8             0.5263150  0.5432198  1.077952 28.00469 12.2222887 2.356632e+01
## 9             0.5429062  0.5466167  1.130952 75.61178  8.3333199 9.612487e+00
## 10            0.6449043  0.5529085  1.204868 28.02294  0.5225661 5.117275e+00
##         PosS   PosSPRO     PosV
## 2  0.8546062 13.647938 8.539509
## 3  0.8092903 12.987725 8.534435
## 4  0.7888840 96.007327 8.530307
## 5  0.7869905 12.781780 8.528860
## 6  0.7795744 12.469272 8.527337
## 7  0.7726076 11.385181 8.526277
## 8  0.7710362 12.523398 8.525991
## 9  0.7849416 10.521372 8.527720
## 10 0.8001299  2.108253 8.530180
##
## Residual Deviance: 6877.847
## AIC: 7255.847
```

After training our model has scores (Residual Deviance: 6877.847, AIC: 7255.847) that are far from ideal. For some data: rare parts of speech like NUM, PRO, INTJ, rare toxicity types: hate_speech: religion, hate_speech: race, lgbtq*, the errors scores are quite high which could disturb the work of the model.

```
predicted_classes <- predict(multi_model, newdata = tox_data)
probs <- predict(multi_model, newdata = tox_data, "probs")

conf_matrix <- as.data.frame(table(predicted_classes, tox_data$tox_rate))
```

```
ggplot(conf_matrix, aes(x = Var2, y = predicted_classes, fill = Freq)) +
  geom_tile(color = "white", alpha = 0.8) +
  geom_text(aes(label = Freq), size = 6) +
  labs(title = "Confusion Matrix: Predicted vs Actual Toxicity Rate",
       x = "Actual Toxicity Rate",
       y = "Predicted Toxicity Rate") +
  scale_fill_gradient(low = "skyblue2", high = "violetred3") +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        panel.grid = element_blank())
```



The following figure presents the confusion matrix of the model predicted scores, as we see only ratings 7-8 are predicted with a good quality. The low rating scores (1-3) are not predicted at all, which could be explained by insufficient amount of data.

Multimodel 2

In order to improve our results we are going to do the feature engineering on the dataset. We are going to transform the ratings into categorical groups and delete the columns with very high error scores: hate_speech: religion, post: inanimate. Also, we are going to remove the extra amount of nouns, reducing their quantity to 500. We are creating 4 categories:

1. **low_tox** - sentences with low toxicity with scores 1-3
2. **mid_tox** - sentences with medium toxicity with scores 4-6
3. **high_tox** - sentences with high toxicity with scores 7-8
4. **extreme_tox** - sentences with extra toxicity with scores 9-10

```
tox_clean <- tox_data %>%
  select(-c(1, 2, 3, 7, 9)) %>%
  mutate(
    tox_group = factor(case_when(
      tox_rate %in% 1:3 ~ "low_tox",
      tox_rate %in% 4:6 ~ "mid_tox",
      tox_rate %in% 7:8 ~ "high_tox",
      tox_rate %in% 9:10 ~ "extreme_tox"
    ), levels = c("low_tox", "mid_tox", "high_tox", "extreme_tox")),
    across(tox_type, as.factor)
  )%>%
  filter(tox_type != "hate_speech: religion")%>%
  filter(response != "post: inanimate")%>%
  group_by(Pos)%>%
  mutate(keep = ifelse(Pos == "S", row_number() <= 500, TRUE)) %>%
  filter(keep) %>%
  select(-keep) %>%
  filter(n()>50)%>%
  droplevels()

glimpse(tox_clean)
```

```
## Rows: 1,125
## Columns: 7
## Groups: Pos [4]
## $ tox_rate     <dbl> 4, 5, 5, 5, 5, 7, 7, 6, 6, 6, 7, 7, 6, 4, 8, 3, 7, 7, 6, …
## $ response     <fct> post: animate, post: animate, post: animate, post: animat…
## $ tox_type     <fct> general_insult, profanity, profanity, profanity, harassme…
## $ phrase_types <fct> direct, direct, direct, direct, direct, direct, direct, d…
## $ lex_counts   <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 1, 1, 1, 1, 2, 2, 1, …
## $ Pos          <fct> S, V, V, A, V, S, S, S, A, S, S, S, S, A, A, S, S, S, V, …
## $ tox_group    <fct> mid_tox, mid_tox, mid_tox, mid_tox, mid_tox, high_tox, hi…
```

```
summary(tox_clean)
```

```
##     tox_rate              response                      tox_type
##  Min.   : 1.000   author       :269   general_insult          :510
##  1st Qu.: 5.000   person       :301   hate_speech: gender     :303
##  Median : 7.000   post: animate:555   hate_speech: nationality:146
##  Mean   : 6.612                       profanity               : 55
##  3rd Qu.: 8.000                       threat                  : 52
##  Max.   :10.000                       harassment              : 43
##                                       (Other)                 : 16
##    phrase_types   lex_counts     Pos            tox_group
##  direct  :716   Min.   :1.000   A  :238   low_tox    : 90
##  indirect:409   1st Qu.:1.000   ADV: 59   mid_tox    :375
##                 Median :2.000   S  :500   high_tox   :483
##                 Mean   :1.714   V  :328   extreme_tox:177
##                 3rd Qu.:2.000
##                 Max.   :4.000
##
```

```
multi_model_clean <- multinom(tox_group ~ tox_type + response + phrase_types + lex_counts + Pos, data = tox_clean
)
```

```
## # weights:  64 (45 variable)
## initial  value 1559.581156
## iter  10 value 1261.381578
## iter  20 value 1178.310728
## iter  30 value 1174.327788
## iter  40 value 1174.108008
## iter  50 value 1173.952695
## final  value 1173.951691
## converged
```

```
summary(multi_model_clean)
```

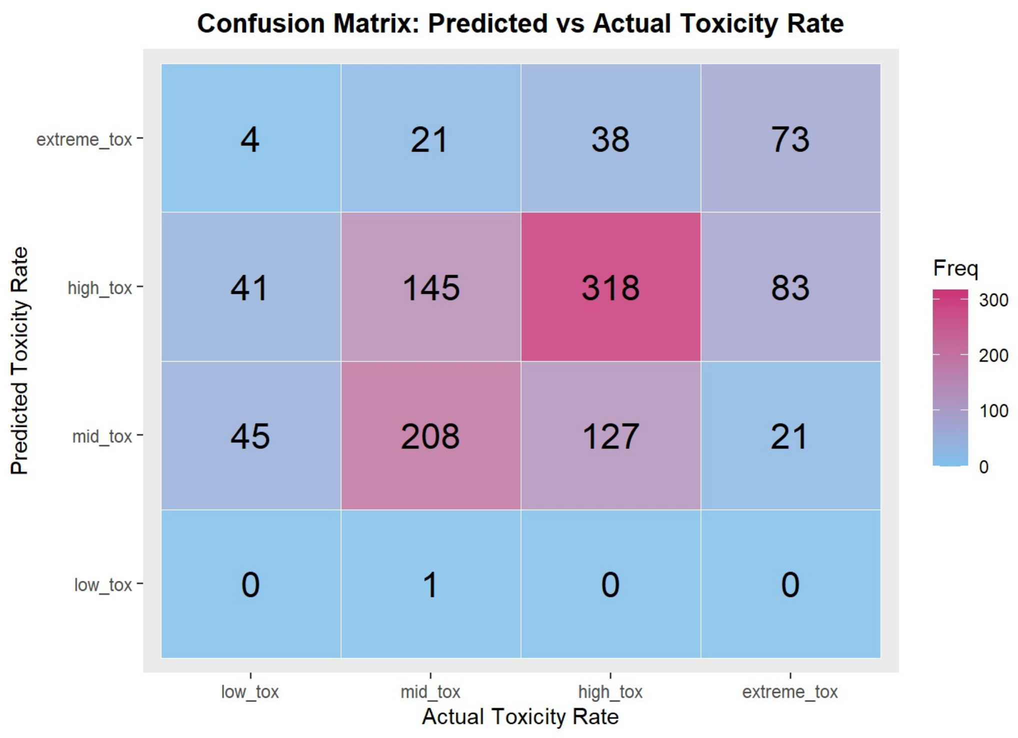
```
## Call:
## multinom(formula = tox_group ~ tox_type + response + phrase_types +
##     lex_counts + Pos, data = tox_clean)
##
## Coefficients:
##             (Intercept) tox_typeharassment tox_typehate_speech: gender
## mid_tox       -1.294949         -0.3519179                  -0.3182684
## high_tox      -1.040655          0.6304975                  -0.2185389
## extreme_tox   -3.312234          3.1137608                  -0.6096141
##             tox_typehate_speech: lgbtq* tox_typehate_speech: nationality
## mid_tox                     -0.08068318                        0.3658757
## high_tox                    -0.66159108                        0.4558601
## extreme_tox                -12.93653498                        1.4336384
##             tox_typehate_speech: race tox_typeprofanity tox_typethreat
## mid_tox                     12.029271          1.512943     -0.5374518
## high_tox                    -6.980013          1.318927     -0.9919894
## extreme_tox                 12.109128          1.730404      0.9591841
##             responseperson responsepost: animate phrase_typesindirect
## mid_tox          0.3314952             0.4133356            0.4441833
## high_tox        -0.2976340             0.5738404           -0.6836966
## extreme_tox      0.1663567             0.7132810           -0.9008600
##             lex_counts   PosADV       PosS      PosV
## mid_tox       1.378872 1.031065  0.4440796 0.4675973
## high_tox      1.557314 1.843307  0.4169531 1.2519908
## extreme_tox   2.150668 1.529598 -0.6493933 0.9670155
##
## Std. Errors:
##             (Intercept) tox_typeharassment tox_typehate_speech: gender
## mid_tox       0.4981587          0.8653613                   0.3018243
## high_tox      0.4928338          0.7953483                   0.3041358
## extreme_tox   0.5854297          0.8061736                   0.4018301
##             tox_typehate_speech: lgbtq* tox_typehate_speech: nationality
## mid_tox                    1.145798e+00                        0.5934825
## high_tox                   1.278024e+00                        0.5836730
## extreme_tox                2.106732e-06                        0.6148709
##             tox_typehate_speech: race tox_typeprofanity tox_typethreat
## mid_tox                  4.671916e-01          1.060658      0.6656609
## high_tox                 3.421437e-09          1.049576      0.6395148
## extreme_tox              4.671907e-01          1.099097      0.6531274
##             responseperson responsepost: animate phrase_typesindirect
## mid_tox          0.3117255             0.3428276            0.2525692
## high_tox         0.3136781             0.3376870            0.2563995
## extreme_tox      0.4032442             0.4078546            0.3202768
##             lex_counts    PosADV      PosS      PosV
## mid_tox      0.2650090 0.8159221 0.3141892 0.3881505
## high_tox     0.2645698 0.7824161 0.3163518 0.3808741
## extreme_tox  0.2799455 0.8522639 0.3858703 0.4202079
##
## Residual Deviance: 2347.903
## AIC: 2437.903
```

As we can see both of our metrics have significantly improved, almost threefold (Residual Deviance: 2347.903, AIC: 2437.903).

```
predicted_classes1 <- predict(multi_model_clean, newdata = tox_clean)
probs1 <- predict(multi_model_clean, newdata = tox_clean, "probs")

conf_matrix1 <- as.data.frame(table(predicted_classes1, tox_clean$tox_group))
```

```
ggplot(conf_matrix1, aes(x = Var2, y = predicted_classes1, fill = Freq)) +
  geom_tile(color = "white", alpha = 0.8) +
  geom_text(aes(label = Freq), size = 6) +
  labs(title = "Confusion Matrix: Predicted vs Actual Toxicity Rate",
       x = "Actual Toxicity Rate",
       y = "Predicted Toxicity Rate") +
  scale_fill_gradient(low = "skyblue2", high = "violetred3") +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        panel.grid = element_blank())
```



The following figure demonstrates the confusion matrix between actual and predicted scores of ratings. We can see that the class with lox toxicity isn't predicted at all, which could be explained by the insufficient amount of comments of this type. Classes with middle and high toxic rate are predicted the best, though they are often confused, but it seems that the difference between classes is not so significant.

Multimodel 3

Here we would like to create a model that would only depend on the amount of toxic words and part of speech in order to see whether part of speech could influence the toxicity rate.

```
multi_model_pos <- multinom(tox_group ~ lex_counts + Pos, data = tox_clean)
```

```
## # weights:  24 (15 variable)
## initial  value 1559.581156
## iter  10 value 1323.368417
## iter  20 value 1280.092571
## final  value 1280.092080
## converged
```

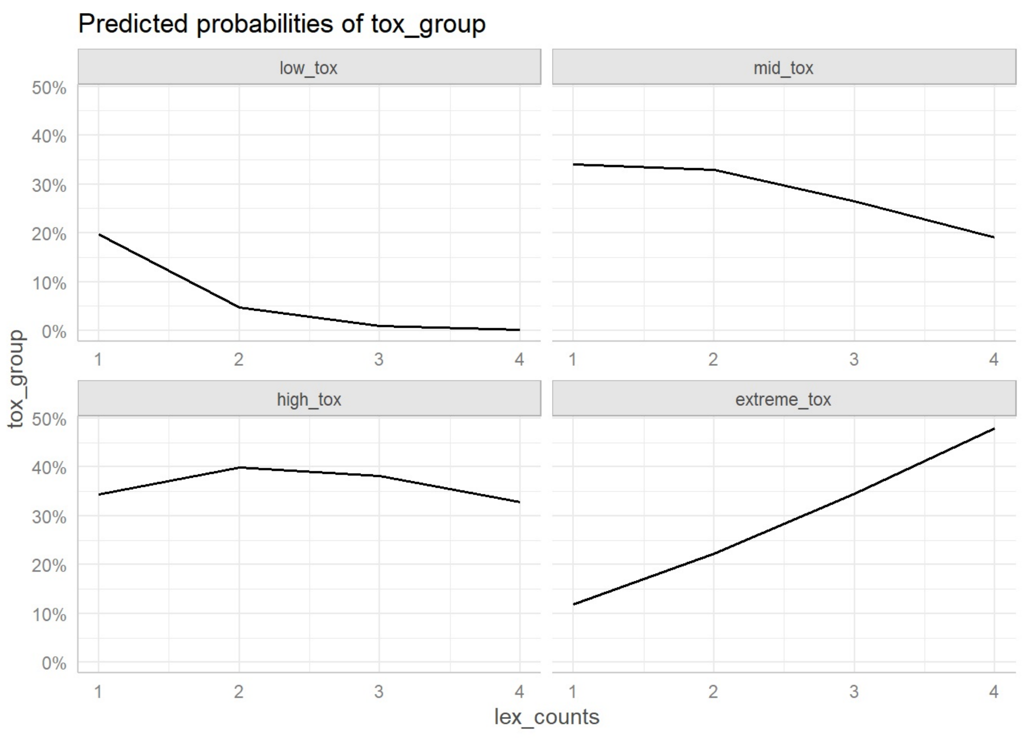
```
summary(multi_model_pos)
```

```
## Call:
## multinom(formula = tox_group ~ lex_counts + Pos, data = tox_clean)
##
## Coefficients:
##             (Intercept) lex_counts    PosADV       PosS      PosV
## mid_tox      -0.8305589   1.377280 0.9448715  0.3890716 0.5075278
## high_tox     -0.9982209   1.555532 1.9372210  0.1647219 1.2010707
## extreme_tox  -2.5368291   2.035323 1.5174658 -0.9917606 1.2151938
##
## Std. Errors:
##             (Intercept) lex_counts    PosADV      PosS      PosV
## mid_tox       0.4307768  0.2584729 0.8089552 0.3010938 0.3739705
## high_tox      0.4287972  0.2574775 0.7749580 0.3005077 0.3630741
## extreme_tox   0.4767294  0.2693982 0.8320760 0.3562953 0.3908605
##
## Residual Deviance: 2560.184
## AIC: 2590.184
```

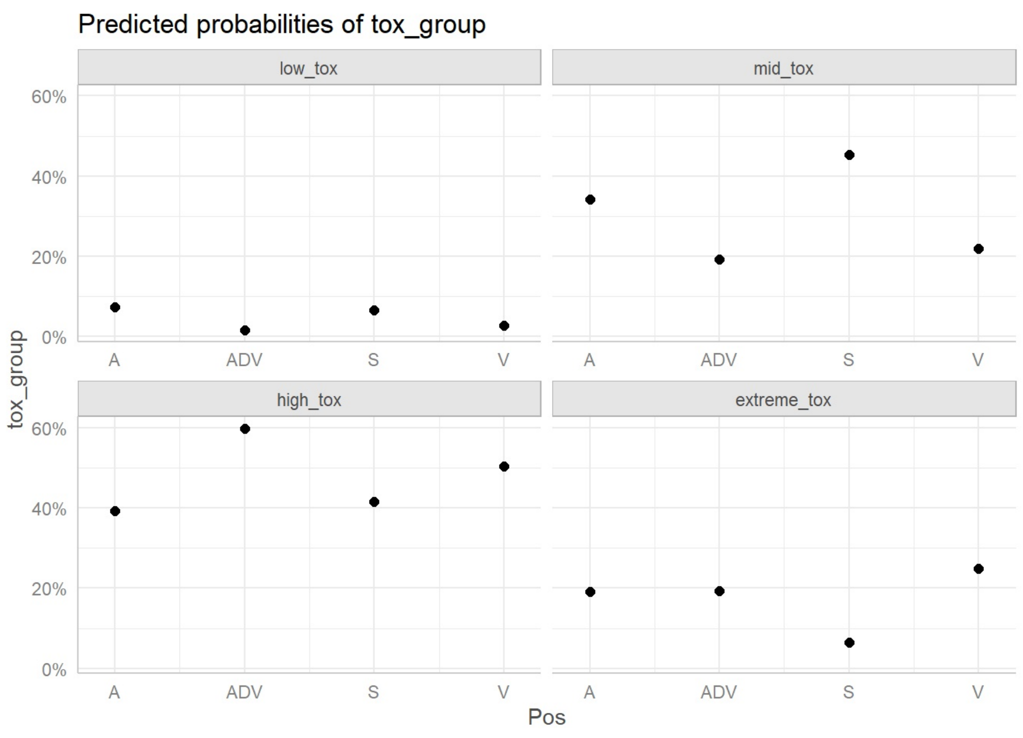
The scores of the model resemble the scores of the model trained on the “clean” dataset (Residual Deviance: 2560.184, AIC: 2590.184).

```
gg_effects <- ggpredict(multi_model_pos, terms = names(coef(multi_model_pos))[-1])
plot(gg_effects)
```

```
## $lex_counts
```



```
##
## $Pos
```

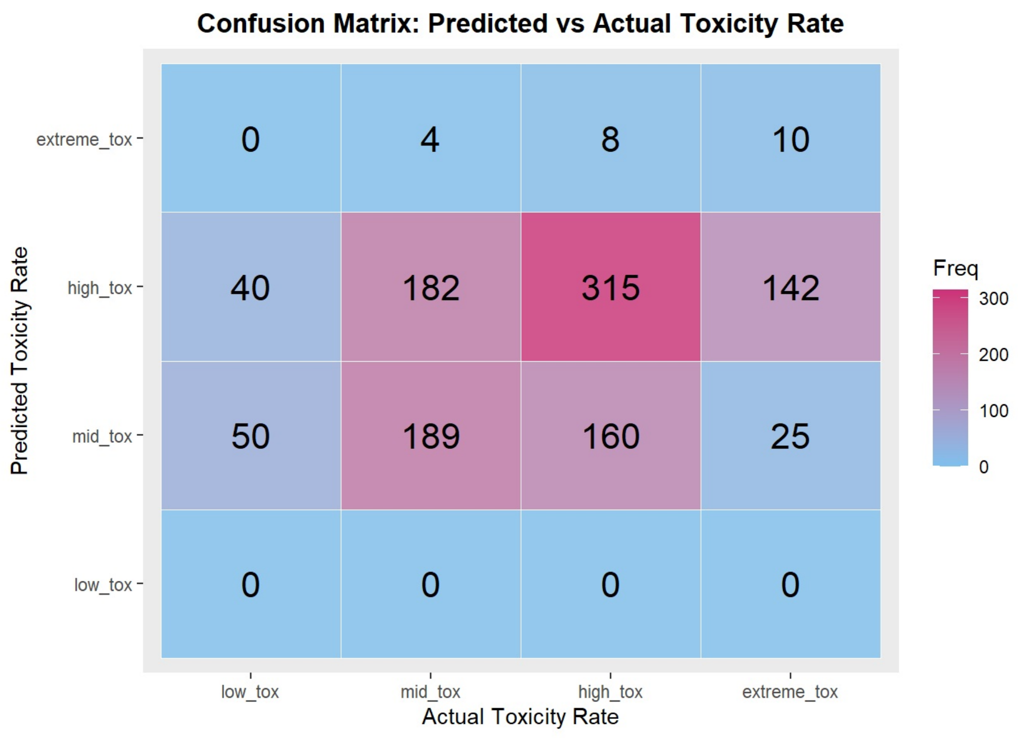


Looking closely at the effects of the model, we could notice that only for comments with extreme toxicity the amount of words makes differences and influences the result. For low and medium toxicity the tendency is reversed, less words have more positive impact on the model than more words. Speaking about parts of speech, it is interesting to note that nouns have more effect on the middle toxicity rating scores, while adverbs are more likely to be the sign of high toxicity group.

```
predicted_classes_pos <- predict(multi_model_pos, newdata = tox_clean)
probs_pos <- predict(multi_model_pos, newdata = tox_clean, "probs")

conf_matrix_pos <- as.data.frame(table(predicted_classes_pos, tox_clean$tox_group))
```

```
ggplot(conf_matrix_pos, aes(x = Var2, y = predicted_classes_pos, fill = Freq)) +
  geom_tile(color = "white", alpha = 0.8) +
  geom_text(aes(label = Freq), size = 6) +
  labs(title = "Confusion Matrix: Predicted vs Actual Toxicity Rate",
       x = "Actual Toxicity Rate",
       y = "Predicted Toxicity Rate") +
  scale_fill_gradient(low = "skyblue2", high = "violetred3") +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        panel.grid = element_blank())
```



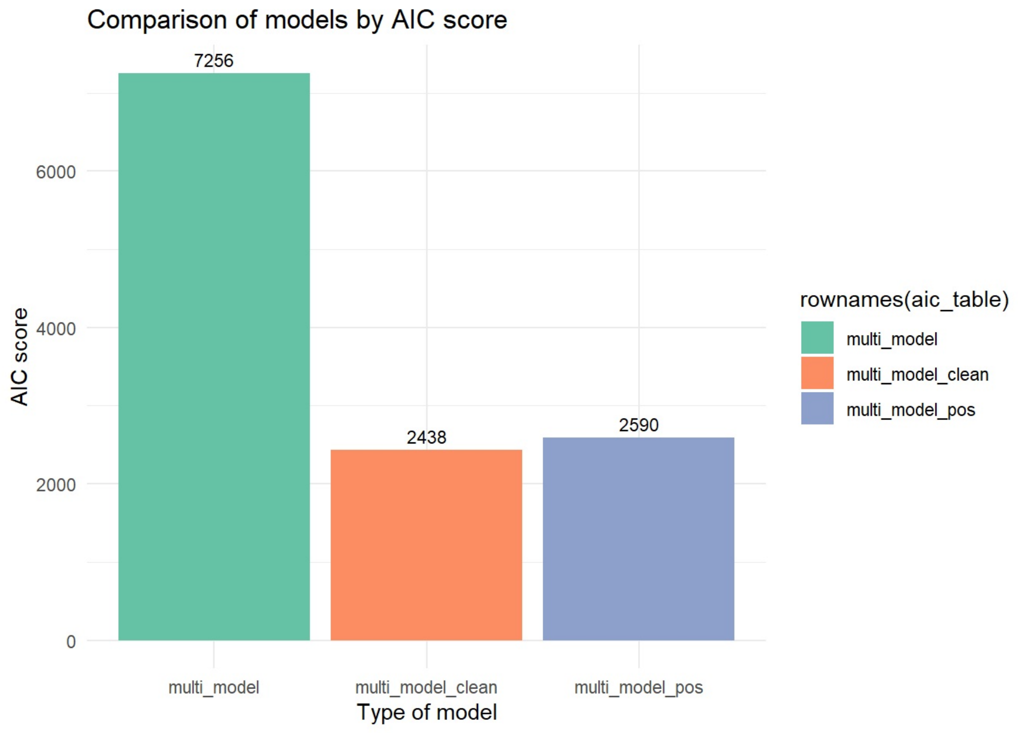
The figure above depicts the confusion matrix for actual and predicted scores. As we see, this time not only low,but also extreme toxicity are badly predicted. The middle and high toxicity classes are often confused.

```
aic_table <- AIC(multi_model, multi_model_clean, multi_model_pos)
```

```
## Warning in AIC.default(multi_model, multi_model_clean, multi_model_pos): не все
## модели были подогнаны под размер данных
```

```
aic_table$delta_AIC <- aic_table$AIC - min(aic_table$AIC)

ggplot(aic_table,aes(x = rownames(aic_table), y = AIC, fill = rownames(aic_table))) +
  geom_col() +
  labs(x = "Model", y = "AIC") +
  ggtitle('Comparison of models by AIC score')+
  xlab('Type of model')+
  geom_text(aes(label = round(AIC)), vjust = -0.5, size = 3) +
  ylab('AIC score')+
  scale_fill_brewer(palette = "Set2")+
  theme_minimal()
```



The graphic depicts a comparison of three different models based on their AIC (Akaike Information Criterion) scores. AIC is a measure of model quality, where lower values indicate better models (better fit with fewer parameters. In this figure we can see that **multinominal model clean** trained on the engineered data gives out the best performance with the scire of 2438. The success could be explained by the feature engineering, data filtering and grouping of target variable.

Related works

The task of finding and filtering toxicity is not unique and new, it often appears in IT competitions on Kaggle and other different sources. For example, for Russian language the dataset of Russian Toxic comments exist [https://www.kaggle.com/datasets/alexandersemiletov/toxic-russian-comments (https://www.kaggle.com/datasets/alexandersemiletov/toxic-russian-comments)]. The annotation in this corpus is easier and more primitive in some way, for example, there are only 3 types of toxicity such as: insult, threat and obscenity, which could be insufficient for the tasks of linguistic expertise.

The other existing corpus which resembles ours in terms of the annotation is Jigsaw Unintended Bias in Toxicity Classification [https://www.kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification/data (https://www.kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification/data)]. This dataset is collected for English language and also has the ratings classification, but instead of types of toxicity it focuses on the identities that act as the target for hate and negativity.

Both of this datasets show that it seems plausible to use our dataset for model training and improving the qualities of the existing filter algorithms. How this work could be done is explained in the article Detoxifying Language Models with a Toxic Corpus [https://aclanthology.org/2022.ltedi-1.6.pdf (https://aclanthology.org/2022.ltedi-1.6.pdf)]. By incorporating toxic data during fine-tuning, the authors demonstrate significant improvements in model safety, measured through reduced harmful outputs and lower bias metrics. Key results show that models trained with this approach achieve better detoxification performance compared to baseline methods, while maintaining linguistic quality. The study highlights the dual utility of toxic corpora—both as a diagnostic tool and a corrective resource—to enhance responsible AI deployment.

Discussion

While we were analysing our data, we found how abnormally the distribution was. Not all the types were presented in the equal matter, which could have hardly influenced the results. Even though we tried to use engineered data, its still could have disturbed the results of chi-square test and t-test. The Welchs’s t-test would be an option to use here. Another reason for concern would be the type of the model, as we were predicting rating scores the ordinal linear model could have been a better fit here if the data distribution was better. I could only predict that an enlarged dataset would outperform and decrease the error values in model training.

To conclude, our analysis provides invaluable insights in the specific of the usage of the toxic language in Russian and shows the possibilities and perspectives of the toxic corpus for further tasks of classification and toxicity filtration in machine learning and fine-tuning.