Spam Dataset

1 Introduction

A data set collected at Hewlett-Packard Labs, that classifies 4601 **e-mails as spam or non-spam** (variable "class"). The spam dataset is one of the datasets used in **The Elements of Statistical Learning** by Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Besides the option to import it from **OpenML** it also comes as an example dataset in the packages **ElemStatLearn** and **kernlab**.

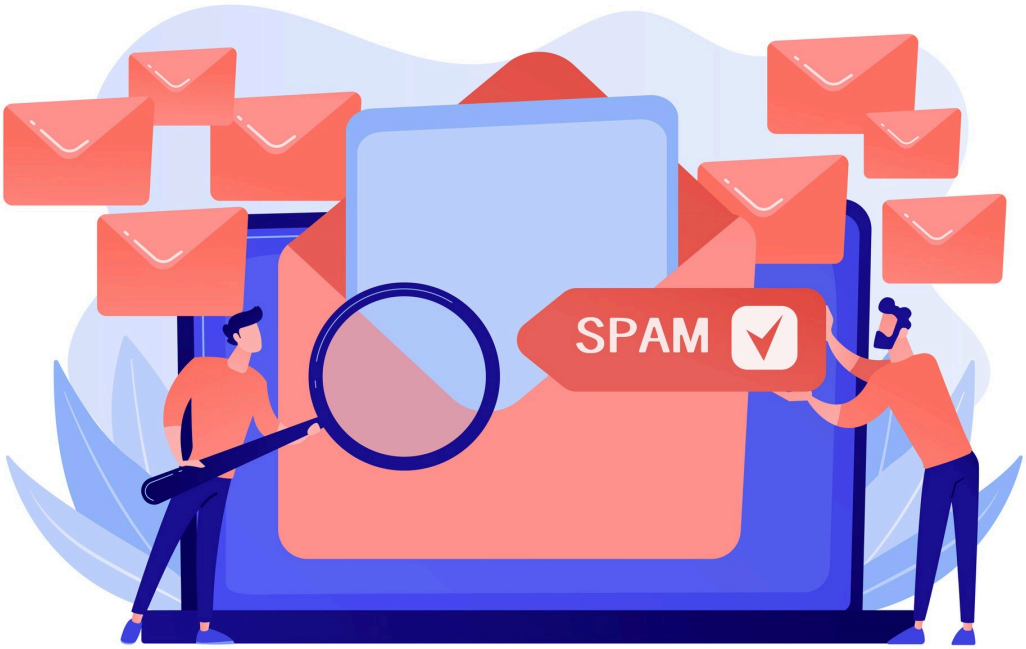


Figure 1: Source: vectorjuice (link)

Dataset basic information:

- `class` (target) : 0 = no spam, 1 = spam
- `word_freq_*`: 48 features corresponding to the relative frequency of a specific word in an e-mail
- `char_freq_*`: 6 features that measures the percentage of a sequence of specific characters occurs relative to the total number of characters
- `capital_run_length_average`: average length of uninterrupted sequences of capital letters
- `capital_run_length_longest`: length of the longest uninterrupted sequence of capital letters
- `capital_run_length_total`: total number of capital letters

We use OpenML (R-Package) to download the dataset in a machine-readable format and convert it into a `data.frame`:

```
# load the dataset from OpenML Library
d <- OpenML::getOMLDataSet(data.id = 44)

# convert the OpenML object to a tibble (enhanced data.frame)
spam <- d %>% dplyr::as_tibble()
skimmed_spam <- skimr::skim(spam)
print(spam)
```

```
## # A tibble: 4,601 x 58
##    word_freq_m~1 word_~2 word_~3 word_~4 word_~5 word_~6 word_~7 word_~8 word_~9
##            <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1          0       0.64    0.64       0    0.32    0       0       0       0
##  2          0.21    0.28    0.5        0    0.14    0.28    0.21    0.07    0
##  3          0.06    0       0.71       0    1.23    0.19    0.19    0.12    0.64
##  4          0       0       0          0    0.63    0       0.31    0.63    0.31
##  5          0       0       0          0    0.63    0       0.31    0.63    0.31
##  6          0       0       0          0    1.85    0       0       1.85    0
##  7          0       0       0          0    1.92    0       0       0       0
##  8          0       0       0          0    1.88    0       0       1.88    0
##  9          0.15    0       0.46       0    0.61    0       0.3     0       0.92
## 10          0.06    0.12    0.77       0    0.19    0.32    0.38    0       0.06
## # ... with 4,591 more rows, 49 more variables: word_freq_mail <dbl>,
## #   word_freq_receive <dbl>, word_freq_will <dbl>, word_freq_people <dbl>,
## #   word_freq_report <dbl>, word_freq_addresses <dbl>, word_freq_free <dbl>,
## #   word_freq_business <dbl>, word_freq_email <dbl>, word_freq_you <dbl>,
## #   word_freq_credit <dbl>, word_freq_your <dbl>, word_freq_font <dbl>,
## #   word_freq_000 <dbl>, word_freq_money <dbl>, word_freq_hp <dbl>,
## #   word_freq_hpl <dbl>, word_freq_george <dbl>, word_freq_650 <dbl>, ...
```

2 Exploratory Data Analysis (EDA)

In this part, we will walk through a few characteristics of spam dataset using library `skimr` and `DataExplorer`.

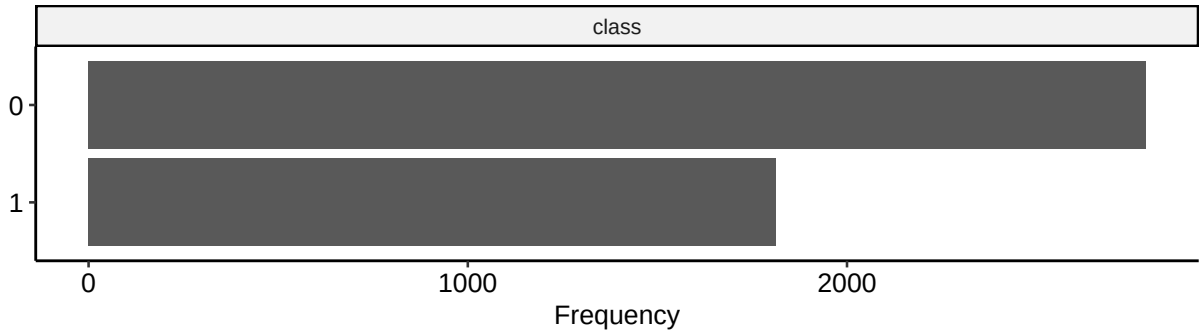
2.1 Factor variables

General statistics about factor variables from spam dataset:

```
skimr::partition(skimmed_spam)$factor %>%
        knitr::kable(format = 'latex', booktabs = TRUE) %>%
        kableExtra::kable_styling(latex_options = 'HOLD_position')
```

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
class	0	1	FALSE	2	0: 2788, 1: 1813

```
DataExplorer::plot_bar(spam, ggtheme = ggpubr::theme_pubr(base_size = 10))
```



The dataset has 1 factor variable, i.e. `class`. The variable does not have missing values. From the statistics and the discrete distribution of `class`, there are 1813 emails in this dataset classified as spam, which account for more than 39% of the total number of emails.

2.2 Numerical variables

First, let's check if the numerical variables have any missing values:

```
spam_numerical <- spam %>% select(where(is.numeric))
# Number of numerical features
ncol(spam_numerical)
```

```
## [1] 57
```

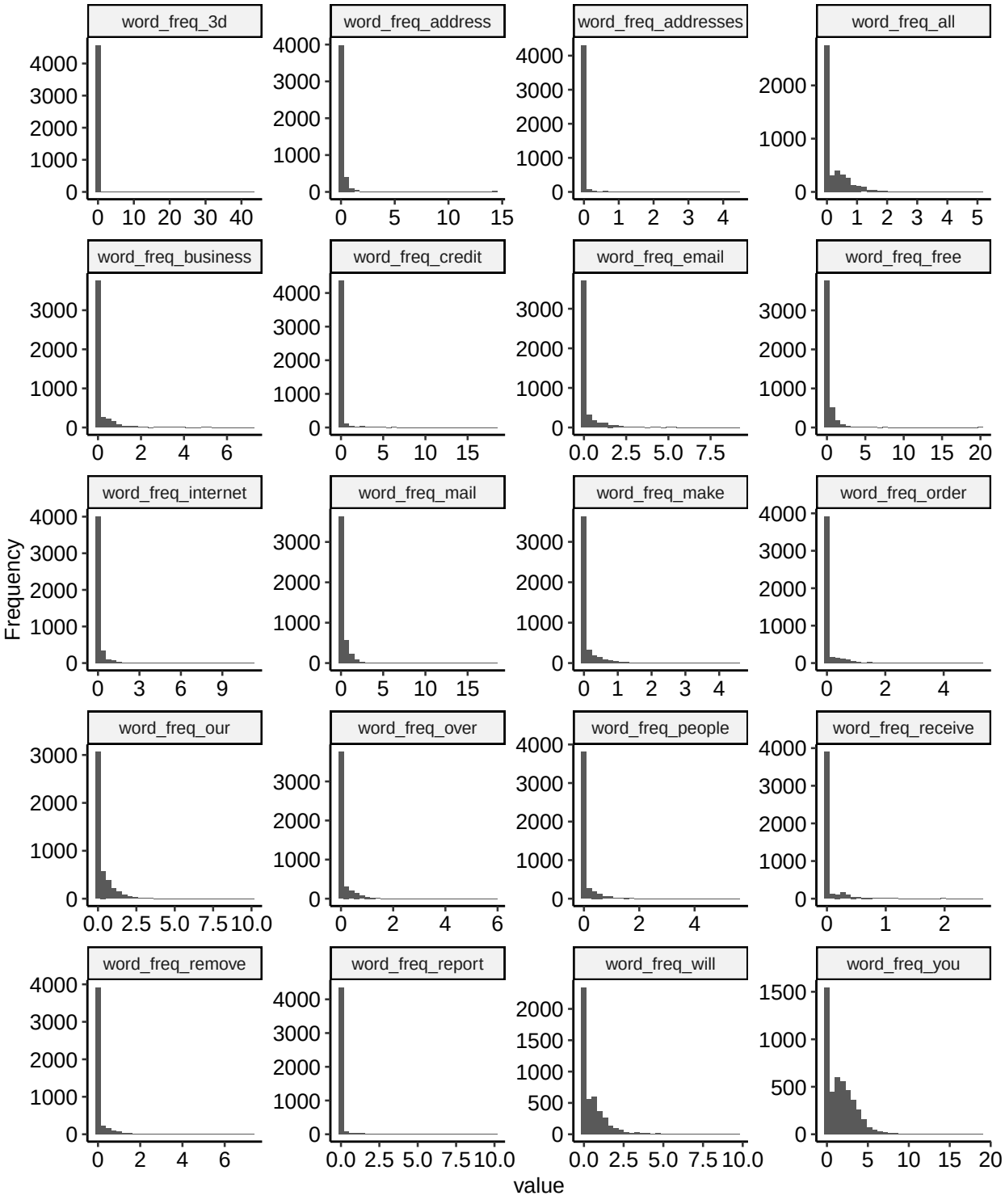
```
# List any numerical features having more than one NA value
names(which(colSums(is.na(spam_numerical))>0))
```

```
## character(0)
```

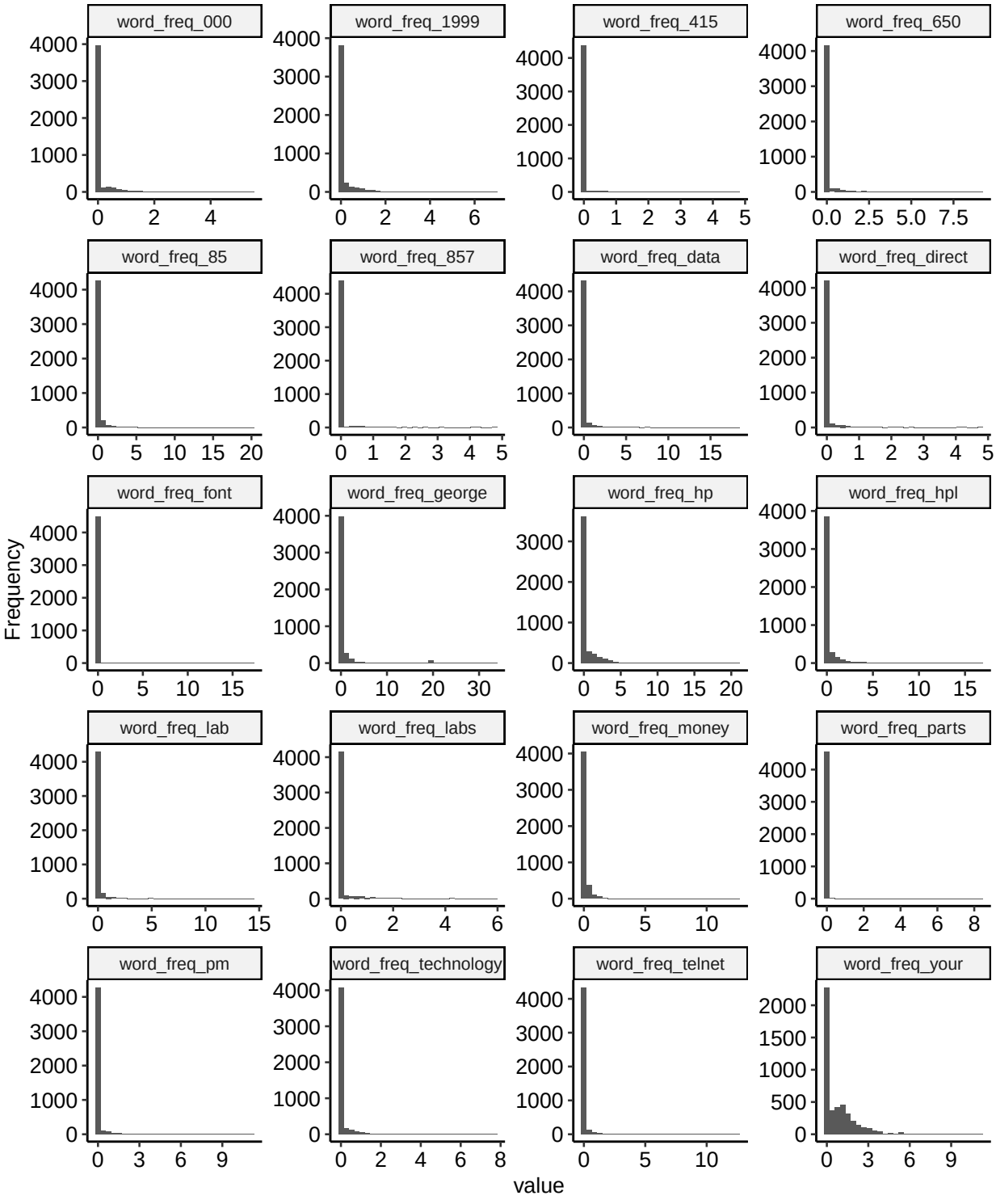
As can be seen, there is no missing value in 57 numerical features.

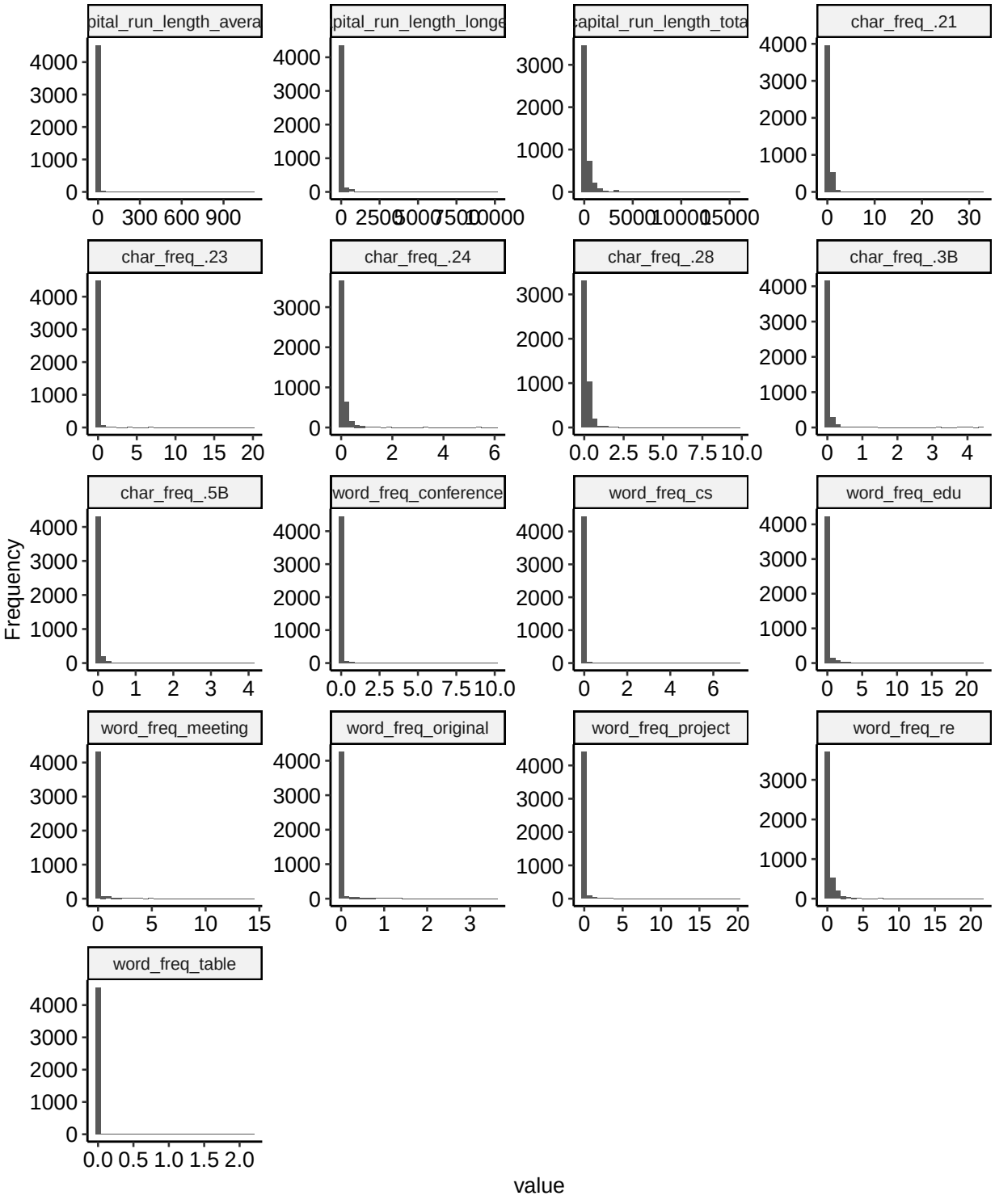
Next, we plot histograms for the numerical features to get to know better their distributions.

```
DataExplorer::plot_histogram(
        spam_numerical,
        ggtheme = ggpubr::theme_pubr(base_size = 10),
        nrow = 5
)
```


word_freq_3d
word_freq_address
word_freq_addresses
word_freq_all
word_freq_business
word_freq_credit
word_freq_email
word_freq_free
word_freq_internet
word_freq_mail
word_freq_make
word_freq_order
word_freq_our
word_freq_over
word_freq_people
word_freq_receive
word_freq_remove
word_freq_report
word_freq_will
word_freq_you
Frequency
value

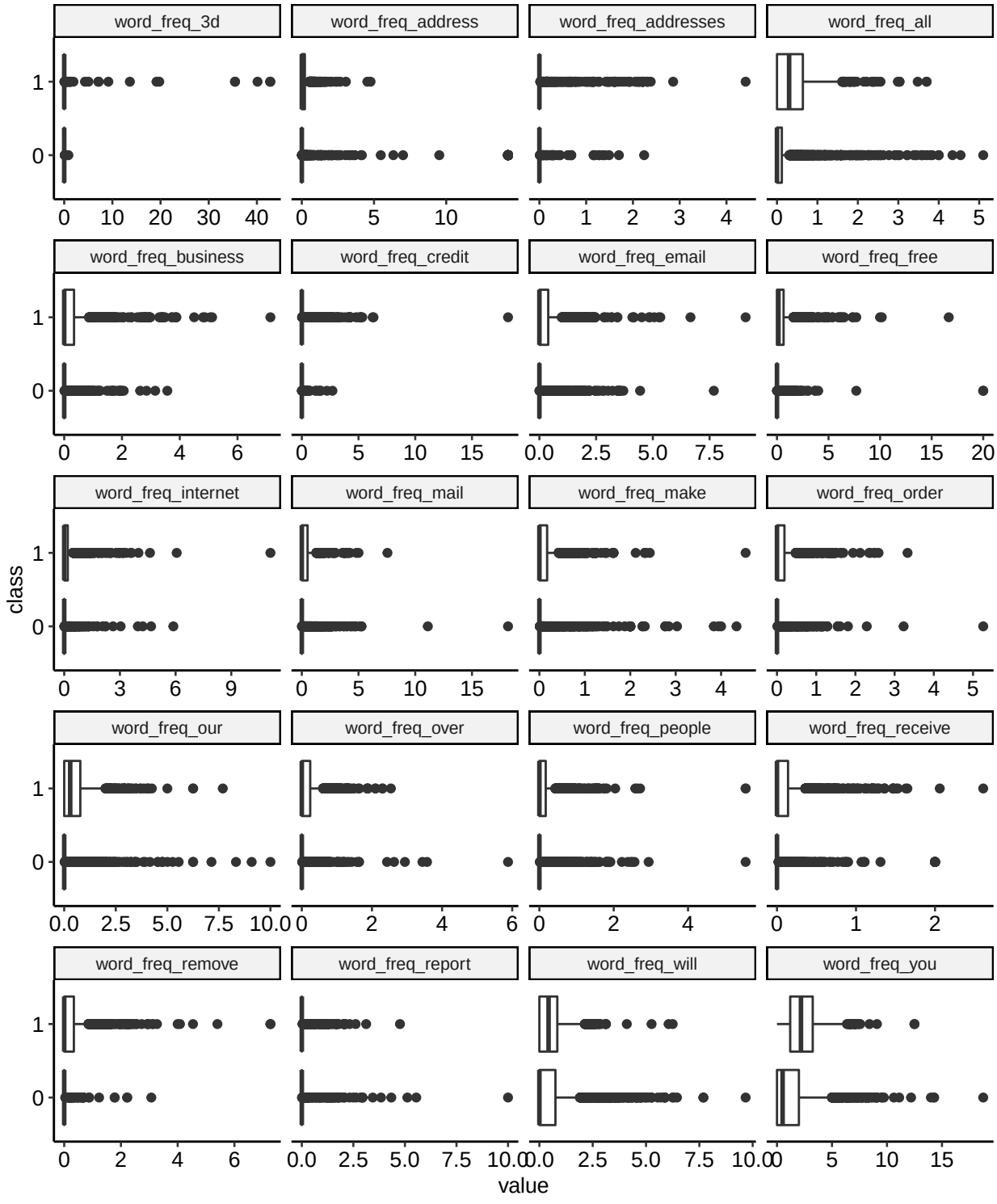
Page 1


word_freq_000
word_freq_1999
word_freq_415
word_freq_650
word_freq_85
word_freq_857
word_freq_data
word_freq_direct
word_freq_font
word_freq_george
word_freq_hp
word_freq_hpl
word_freq_lab
word_freq_labs
word_freq_money
word_freq_parts
word_freq_pm
word_freq_technology
word_freq_telnet
word_freq_your
Frequency
value

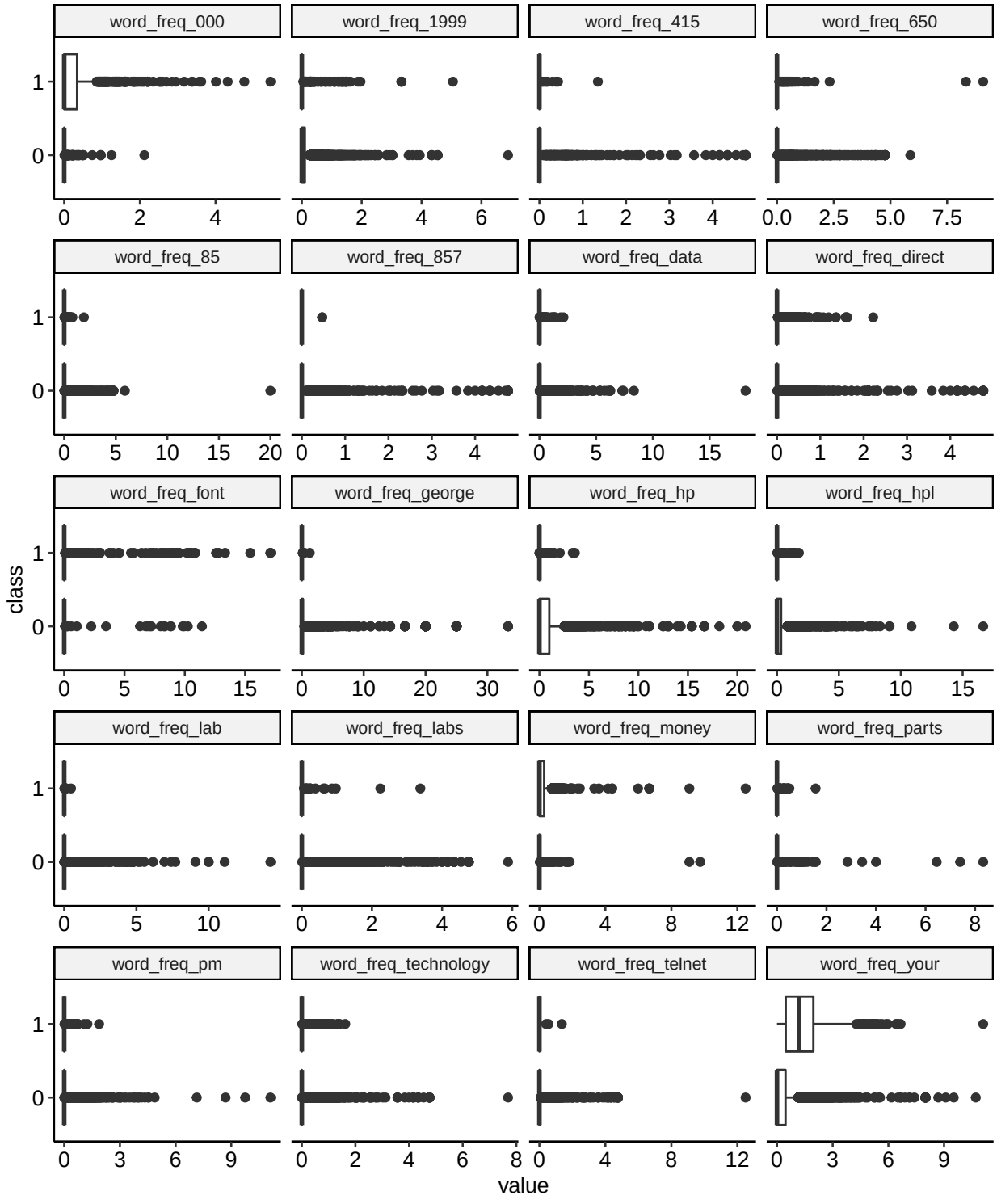


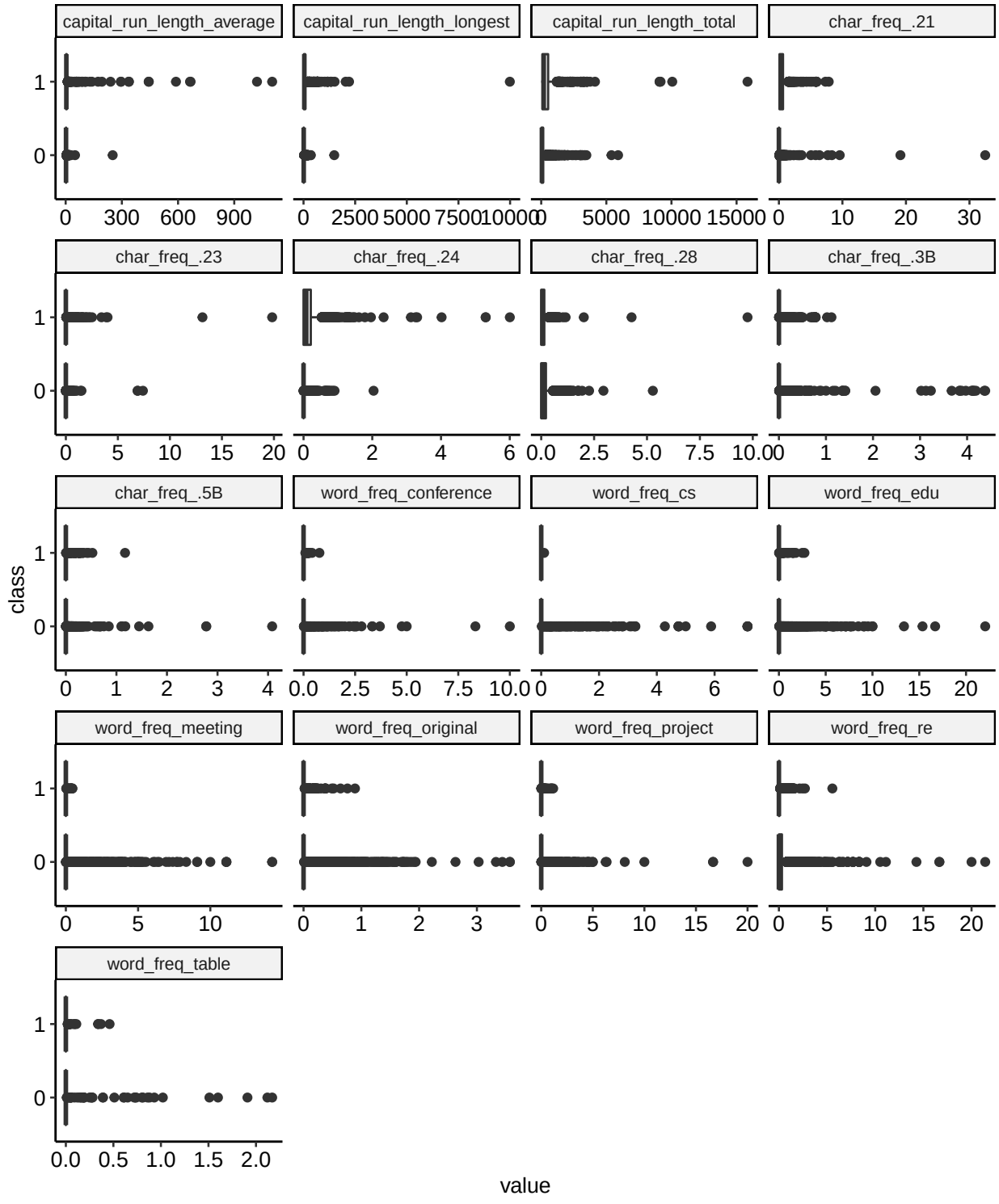
According to these histograms, it is clear that the distributions of most numerical variables are highly skewed. We can also plot the boxplots for these numerical variables and separate by class (spam/non-spam) to discover more information:

```
DataExplorer::plot_boxplot(
        spam,
        by = "class",
        ggtheme = ggpubr::theme_pubr(base_size = 10),
        nrow = 5
)
```


word_freq_3d
word_freq_address
word_freq_addresses
word_freq_all
word_freq_business
word_freq_credit
word_freq_email
word_freq_free
word_freq_internet
word_freq_mail
word_freq_make
word_freq_order
word_freq_our
word_freq_over
word_freq_people
word_freq_receive
word_freq_remove
word_freq_report
word_freq_will
word_freq_you
class
value

Page 1

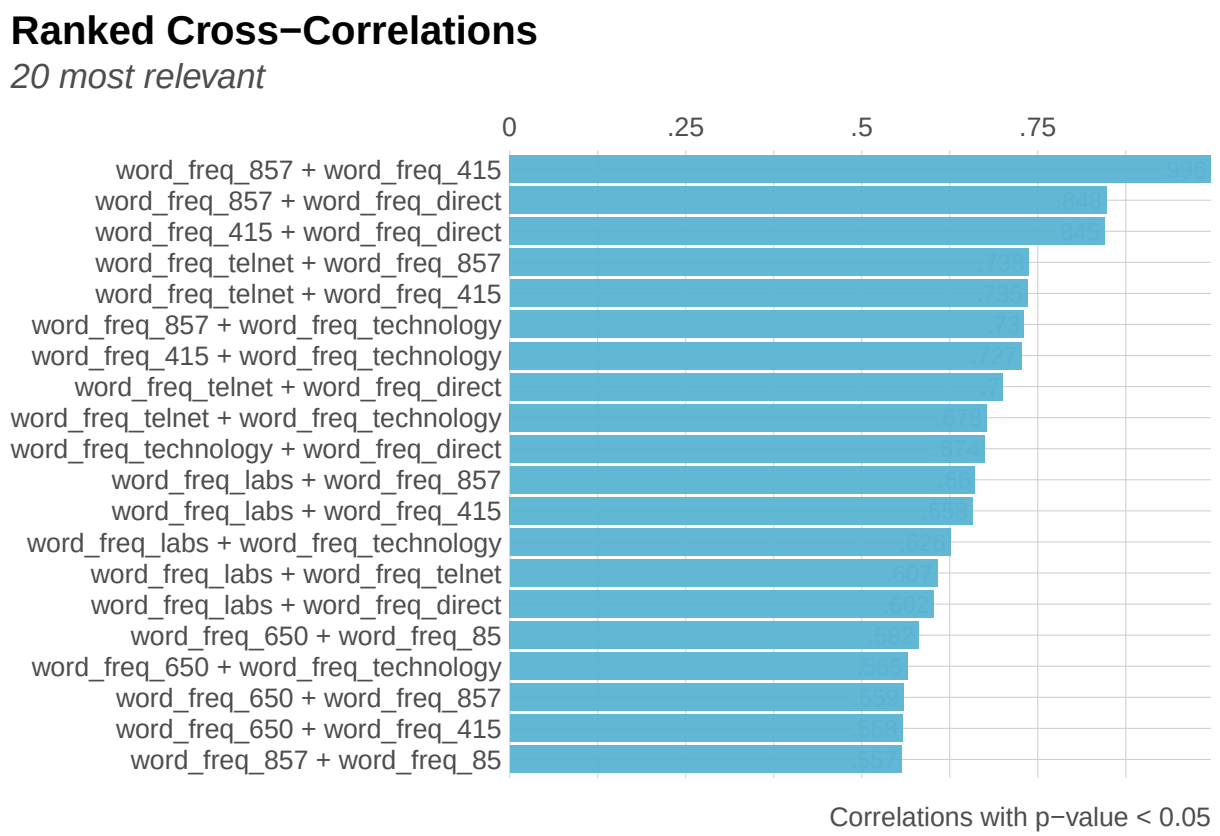

word_freq_000
word_freq_1999
word_freq_415
word_freq_650
word_freq_85
word_freq_857
word_freq_data
word_freq_direct
word_freq_font
word_freq_george
word_freq_hp
word_freq_hpl
word_freq_lab
word_freq_labs
word_freq_money
word_freq_parts
word_freq_pm
word_freq_technology
word_freq_telnet
word_freq_your
class
value



From these boxplots, it can be seen that the numerical features have a lot of outliers. Another interesting point is that the word count for `you` and `your` can potentially be an indicator for specifying spam emails as spam emails tend to have more occurrences of these words.

To understand more the linear relationship between the pairs of numerical variables, we create a correlation ranking of top 20 with the highest magnitude of correlation:

```
corr_cross(spam_numerical,
  max_pvalue = 0.05, # display only significant correlations (at 5% level)
  top = 20 # display top 20 couples of variables (by correlation coefficient)
)
```



From this ranking, we can infer a few things. First, the words/numbers `857`, `415`, `direct`, `telnet`, `technology` are highly correlated. This suggests these words might belong to an underlying group. Moreover, the high level of correlation between `857` and `415` may also suggest the presence of collinearity.

Next, we begin with the data preprocessing notes.

3 Data preprocessing notes

In this section, we present a few notes that can be beneficial for preprocessing the data.

3.1 Data quality assessment

From the EDA, we can see that this dataset is clean with no missing data, mismatched data types, the measurement is consistent between features, which is simply the count.

3.2 Data cleaning

The dataset has a lot of outliers in the numerical features. However, handling outliers needs to be taken with care. Do those outliers exist because of some errors in measurements? Or do they just represent natural

variations in the true population? In the case of this dataset, numerical features have highly right skewed distribution, which can be the cause of outliers.

3.3 Data transformation

As the numerical features are highly right skewed, it may be helpful for some models to perform `log` transformation to mitigate the skewness and reduce the outliers. Here are some examples after applying log scale:

```
spam_numerical_log <- spam_numerical %>%
        mutate_all(~(log(.) %>% as.vector)) %>%
        rename_with(~paste0(.x, "_log"))
DataExplorer::plot_histogram(
        spam_numerical_log[,1:16],
        ggtheme = ggpubr::theme_pubr(base_size = 10)
)
```