Sonar Dataset

1 Introduction

This is the data set used by Gorman and Sejnowski in their study ("Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets" in Neural Networks, Vol. 1, pp. 75-89.) of the classification of sonar signals using a neural network. The task is to train a network to discriminate between sonar signals bounced off a metal cylinder and those bounced off a roughly cylindrical rock.

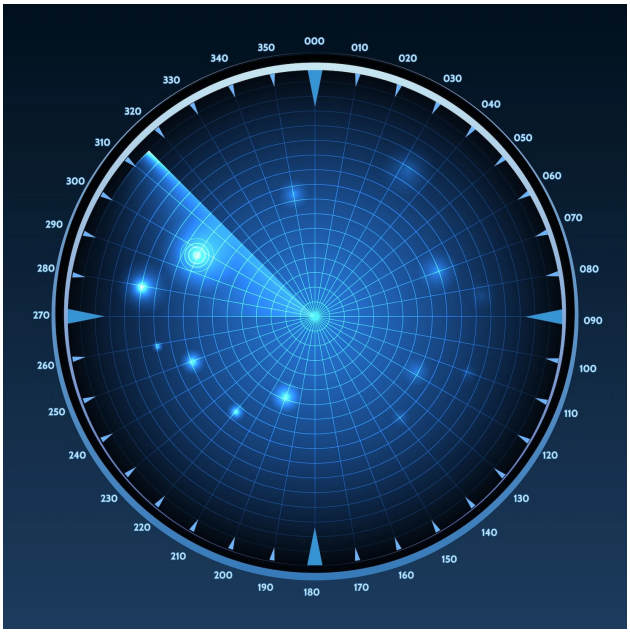


Figure 1: Source: macrovector (link)

Dataset basic information:

- **`class`** (target): "Rock" / "Mine" (metal cylinder)
- `attribute_[1-60]`: 60 variables, each variable represents the energy within a particular frequency band, integrated over a certain period of time. The integration aperture for higher frequencies occur later in time, since these frequencies are transmitted later during the chirp.

The numbers in the labels are in increasing order of aspect angle, but they do not encode the angle directly.

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 = 40)
```

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

```
## # A tibble: 208 x 61
##    attribute_1 attribu~1 attri~2 attri~3 attri~4 attri~5 attri~6 attri~7 attri~8
##          <dbl>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1      0.02      0.0371  0.0428  0.0207  0.0954  0.0986  0.154   0.160   0.311
##  2      0.0453    0.0523  0.0843  0.0689  0.118   0.258   0.216   0.348   0.334
##  3      0.0262    0.0582  0.110   0.108   0.0974  0.228   0.243   0.377   0.560
##  4      0.01      0.0171  0.0623  0.0205  0.0205  0.0368  0.110   0.128   0.0598
##  5      0.0762    0.0666  0.0481  0.0394  0.059   0.0649  0.121   0.247   0.356
##  6      0.0286    0.0453  0.0277  0.0174  0.0384  0.099   0.120   0.183   0.210
##  7      0.0317    0.0956  0.132   0.141   0.167   0.171   0.0731  0.140   0.208
##  8      0.0519    0.0548  0.0842  0.0319  0.116   0.0922  0.103   0.0613  0.146
##  9      0.0223    0.0375  0.0484  0.0475  0.0647  0.0591  0.0753  0.0098  0.0684
## 10      0.0164    0.0173  0.0347  0.007   0.0187  0.0671  0.106   0.0697  0.0962
## # ... with 198 more rows, 52 more variables: attribute_10 <dbl>,
## #   attribute_11 <dbl>, attribute_12 <dbl>, attribute_13 <dbl>,
## #   attribute_14 <dbl>, attribute_15 <dbl>, attribute_16 <dbl>,
## #   attribute_17 <dbl>, attribute_18 <dbl>, attribute_19 <dbl>,
## #   attribute_20 <dbl>, attribute_21 <dbl>, attribute_22 <dbl>,
## #   attribute_23 <dbl>, attribute_24 <dbl>, attribute_25 <dbl>,
## #   attribute_26 <dbl>, attribute_27 <dbl>, attribute_28 <dbl>, ...
```

2 Exploratory Data Analysis (EDA)

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

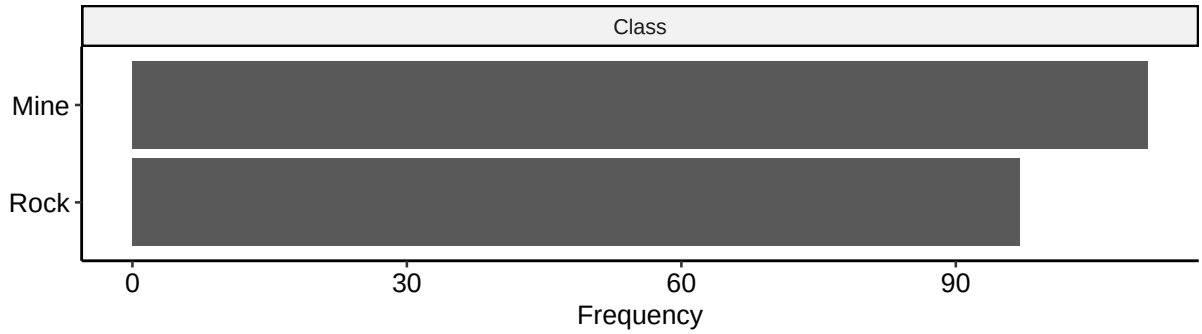
2.1 Factor variables

General statistics about factor variables from sonar dataset:

```
skimr::partition(skimmed_sonar)$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	Min: 111, Roc: 97

```
sonar_factor <- sonar %>% select(where(is.factor))
DataExplorer::plot_bar(sonar_factor, ggtheme = ggpubr::theme_pubr(base_size = 10))
```



There is only one factor variable in this dataset, and it is also the target variable: `class`. The factor variable does not have missing values and have a balanced distribution between two of its factors: `Mine` accounts for roughly 53.4% and `Rock` accounts for roughly 46.6% of total data points.

2.2 Numerical Variables

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

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

```
## [1] 60
```

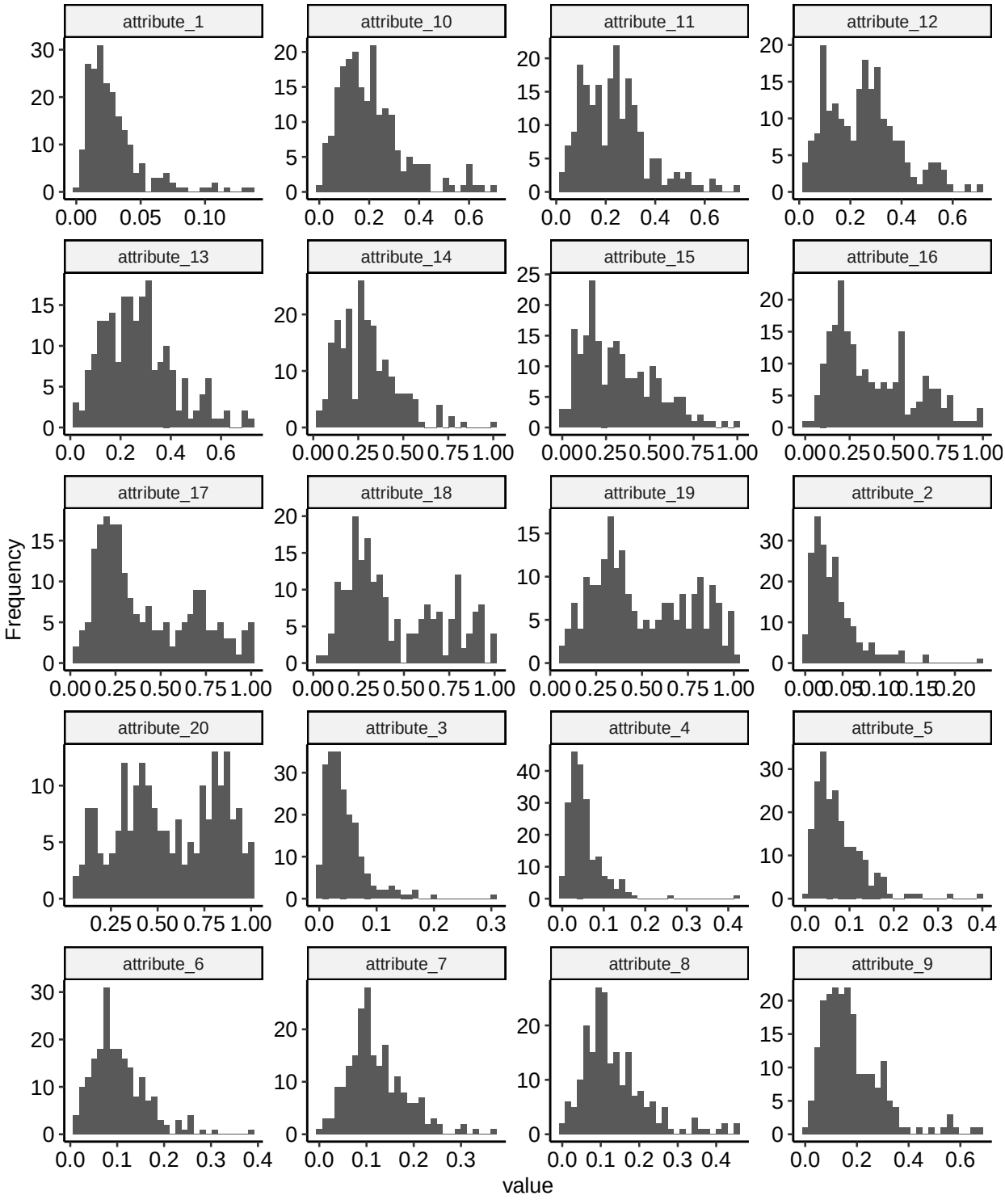
```
# List any numerical features having more than one NA value
names(which(colSums(is.na(sonar_numerical))>0))
```

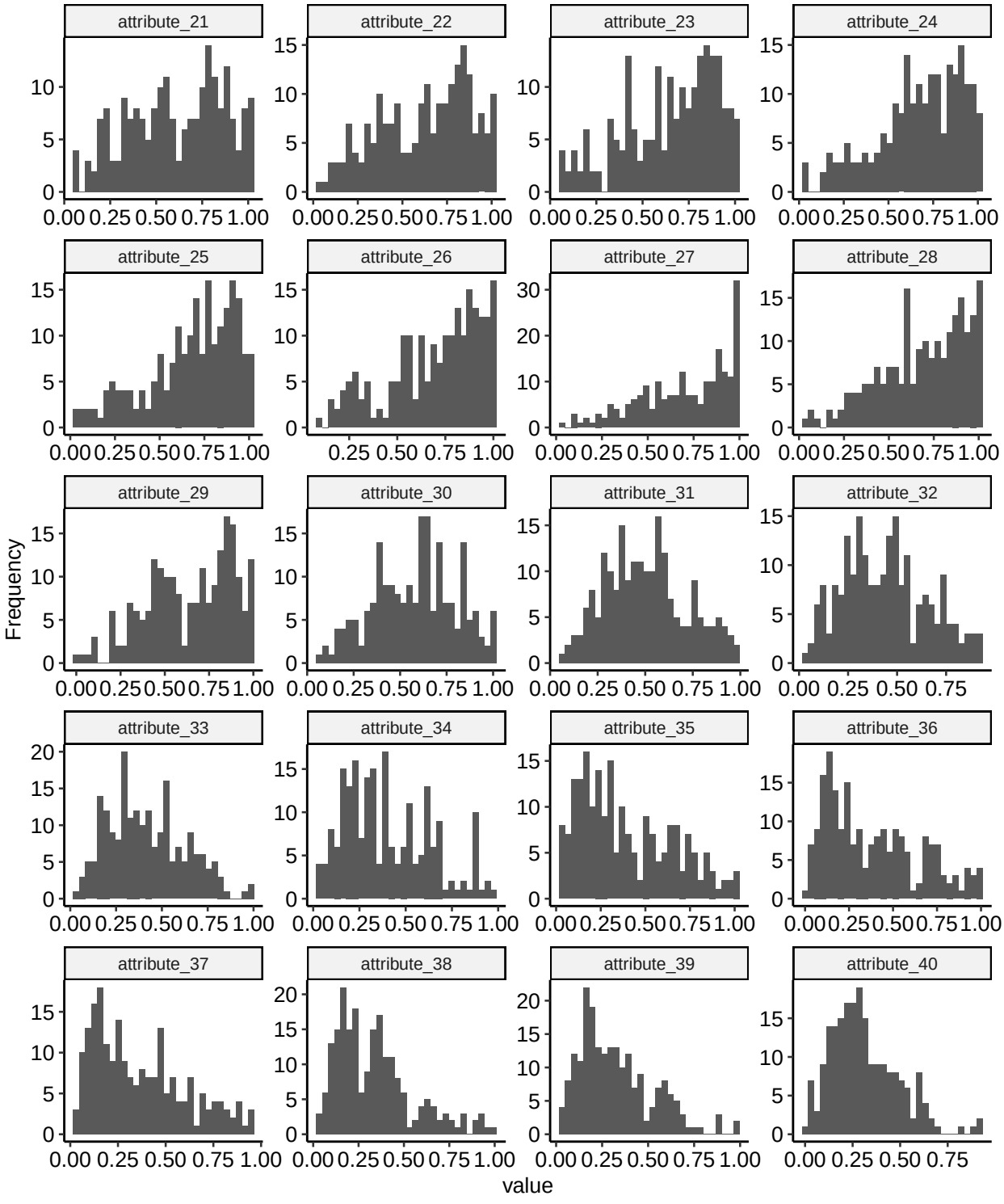
```
## character(0)
```

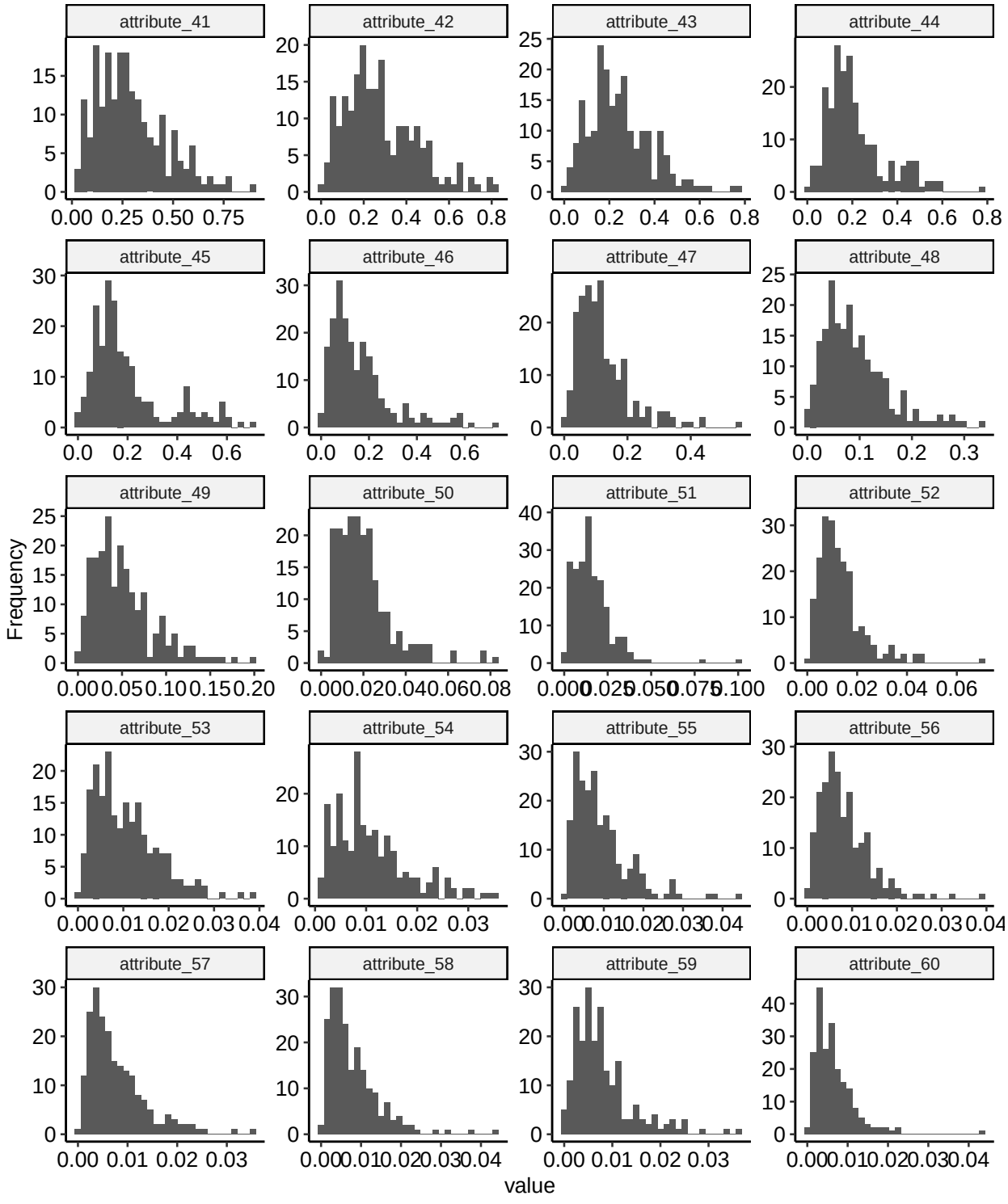
As can be seen, there is no missing value in 60 numerical features.

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

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


attribute_1
attribute_10
attribute_11
attribute_12
attribute_13
attribute_14
attribute_15
attribute_16
attribute_17
attribute_18
attribute_19
attribute_2
attribute_20
attribute_3
attribute_4
attribute_5
attribute_6
attribute_7
attribute_8
attribute_9
Frequency
value

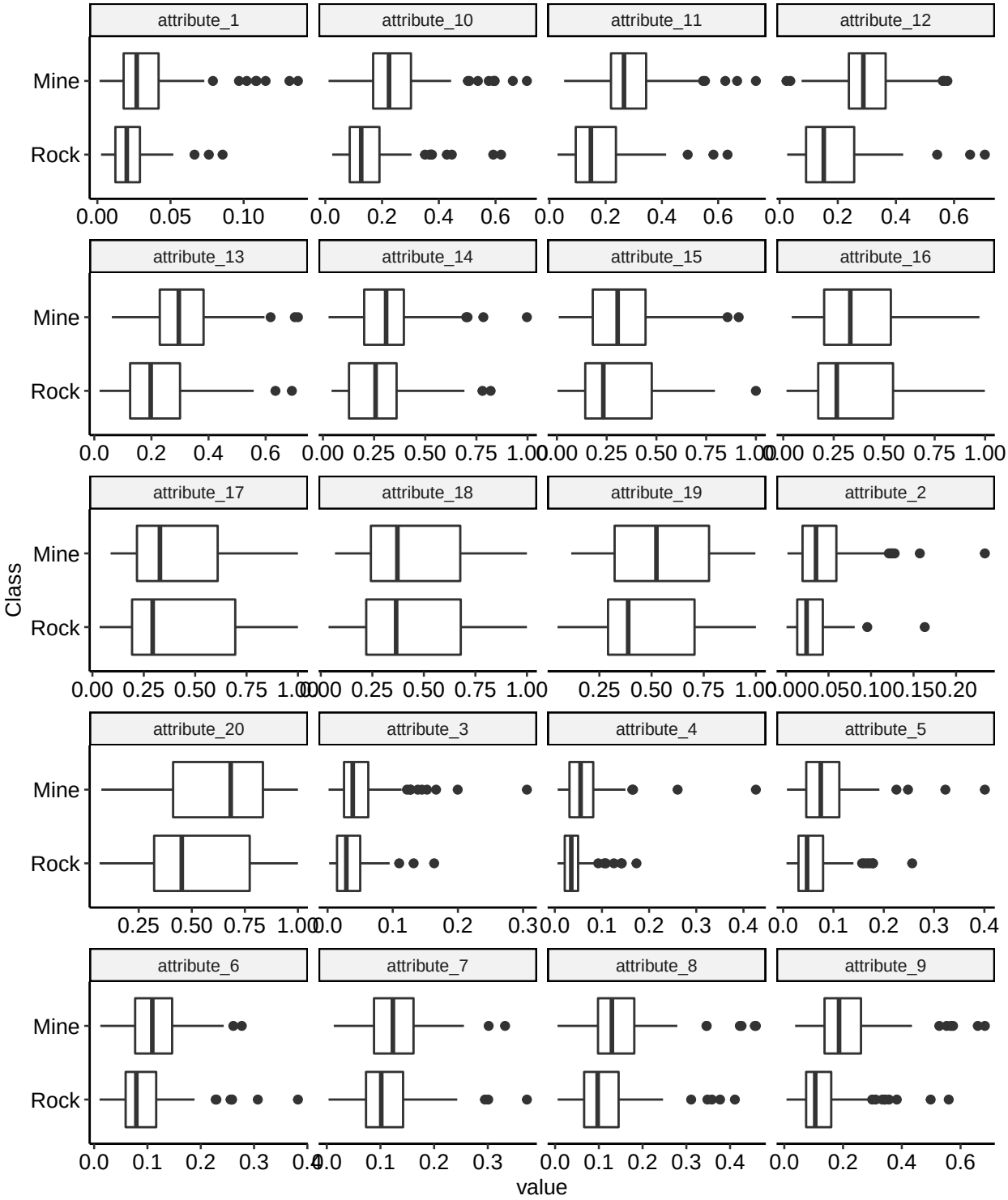

attribute_21
attribute_22
attribute_23
attribute_24
attribute_25
attribute_26
attribute_27
attribute_28
attribute_29
attribute_30
attribute_31
attribute_32
attribute_33
attribute_34
attribute_35
attribute_36
attribute_37
attribute_38
attribute_39
attribute_40
Frequency
value



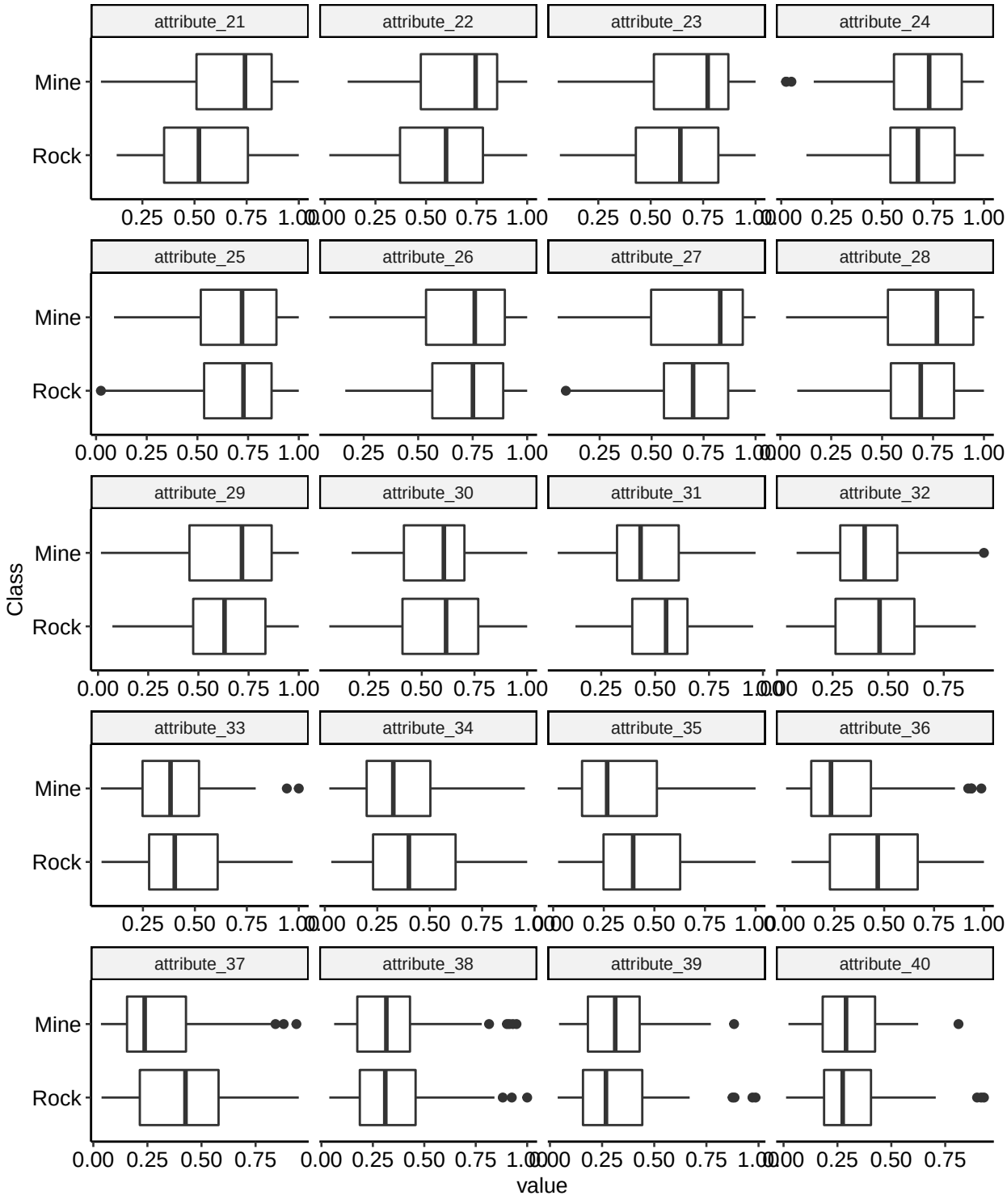
As can be seen from the histograms, there are a lot of features with highly skewed distribution. We can also plot the boxplots for these numerical variables and separate by `Class` to discover more information:

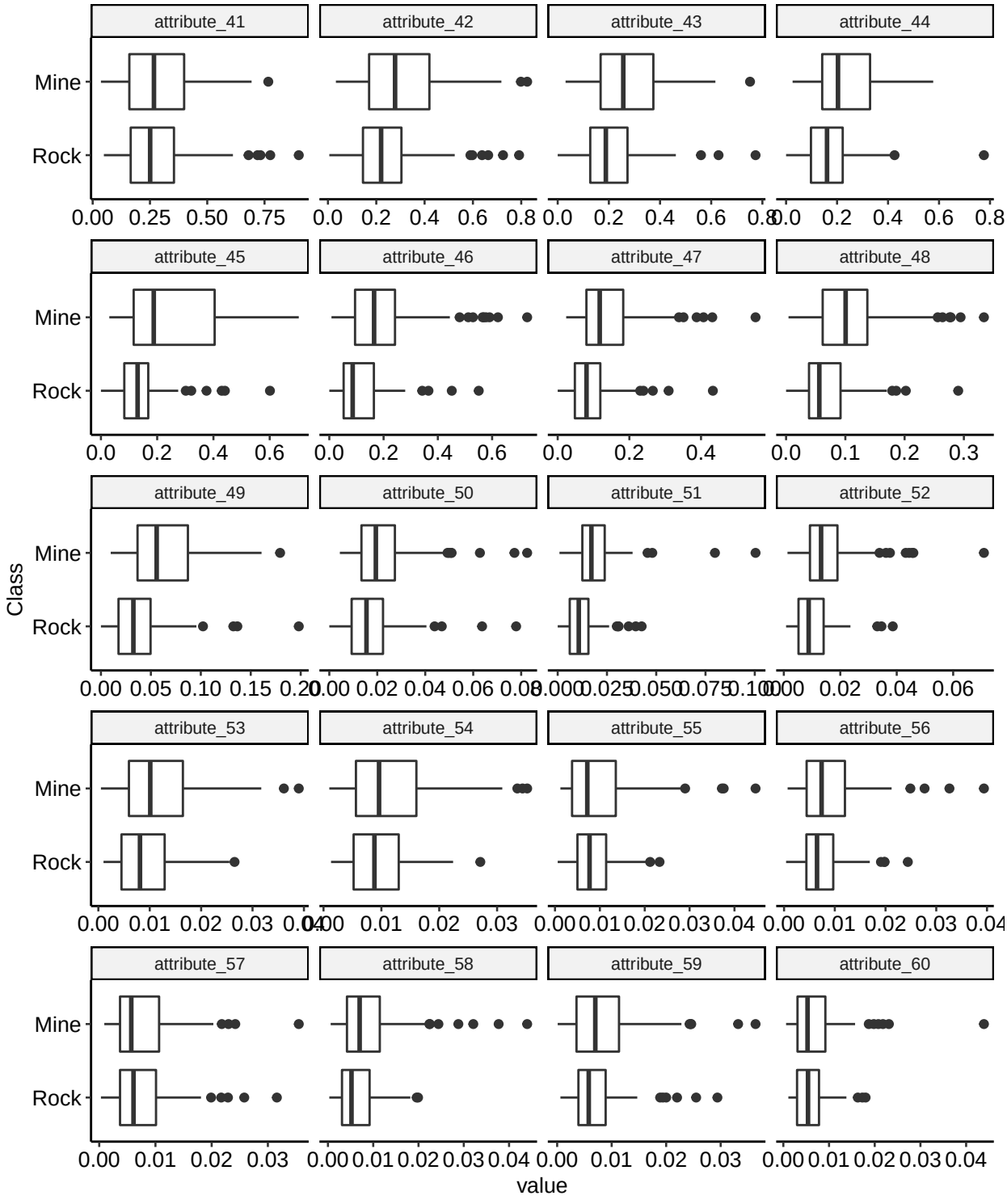
```
DataExplorer::plot_boxplot(
        sonar,
        by = "Class",
```

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



Page 1


attribute_21
attribute_22
attribute_23
attribute_24
attribute_25
attribute_26
attribute_27
attribute_28
attribute_29
attribute_30
attribute_31
attribute_32
attribute_33
attribute_34
attribute_35
attribute_36
attribute_37
attribute_38
attribute_39
attribute_40
Mine
Rock
Class
0.00 0.25 0.50 0.75 1.00
value

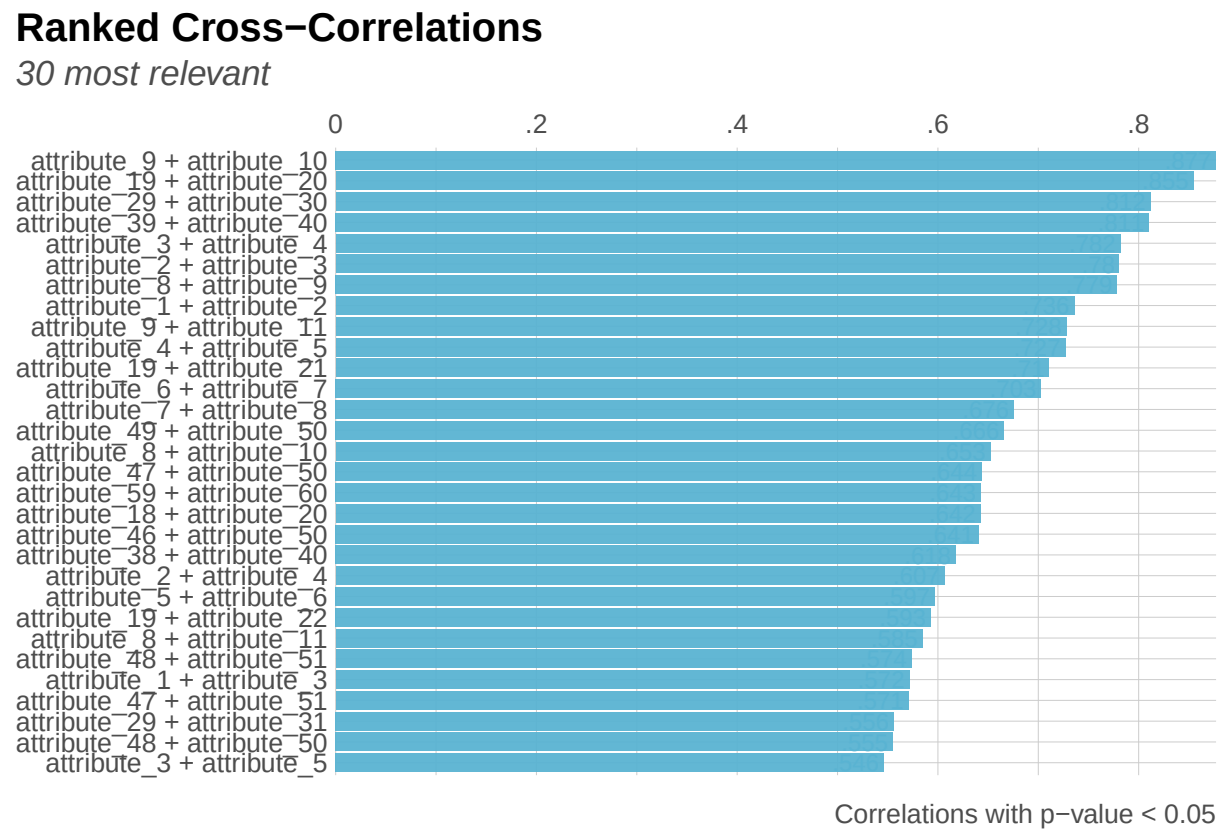


According to these boxplots, there are features that have stronger separation between the two classes and can be used as indicators for separating them. For example, with features `attribute_10`, `attribute_11`, `attribute_12`, lower values of these features may indicate the data point to be from class `Rock` and vice versa.

To understand more the linear relationship between the pairs of numerical variables, we create a correlation

ranking of top 30 with the highest magnitude of correlation:

```
corr_cross(sonar_numerical,
  max_pvalue = 0.05, # display only significant correlations (at 5% level)
  top = 30 # display top 30 couples of variables (by correlation coefficient)
)
```



From the cross correlation graph, we can see that there are pairs of numerical variables that have strong positive correlation (>0.8): `attribute_9` - `attribute_10`, `attribute_19` - `attribute_20`, `attribute_29` - `attribute_30`, `attribute_39` - `attribute_40`. There are also other 16 pairs having positive correlation > 0.6. These signals may 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 the energy within a particular frequency band, integrated over a certain period of time.

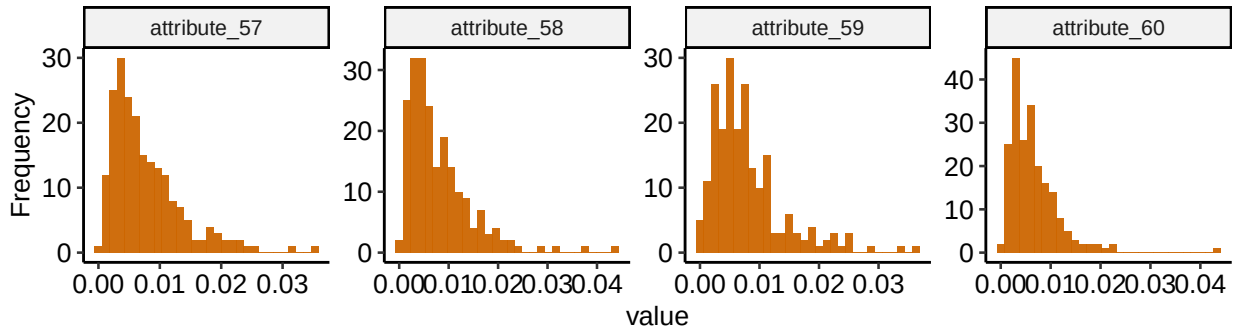
3.2 Data cleaning

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, some 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 can be helpful for some models to apply `log` transformation to mitigate the skewness and reduce the outliers. Here are some examples after applying log scale:

```
# Here are the features before log transforming
DataExplorer::plot_histogram(
        sonar_numerical[,57:60],
        ggtheme = ggpubr::theme_pubr(base_size = 10)
)
```




```
# Here are the features after log transforming
sonar_numerical_log <- sonar_numerical %>%
        mutate_all(~(log(.) %>% as.vector)) %>%
        rename_with(~paste0(.x, "_log"))
DataExplorer::plot_histogram(
        sonar_numerical_log[,57:60],
        ggtheme = ggpubr::theme_pubr(base_size = 10)
)
```