Competition: Spam and Ham Classification

Dario Samuele Pishvai

2023-05-12

Introduction to the Dataset

The Dataset is composed by two Variables: “email” and “class”:

• “email” that contains the text of the emails

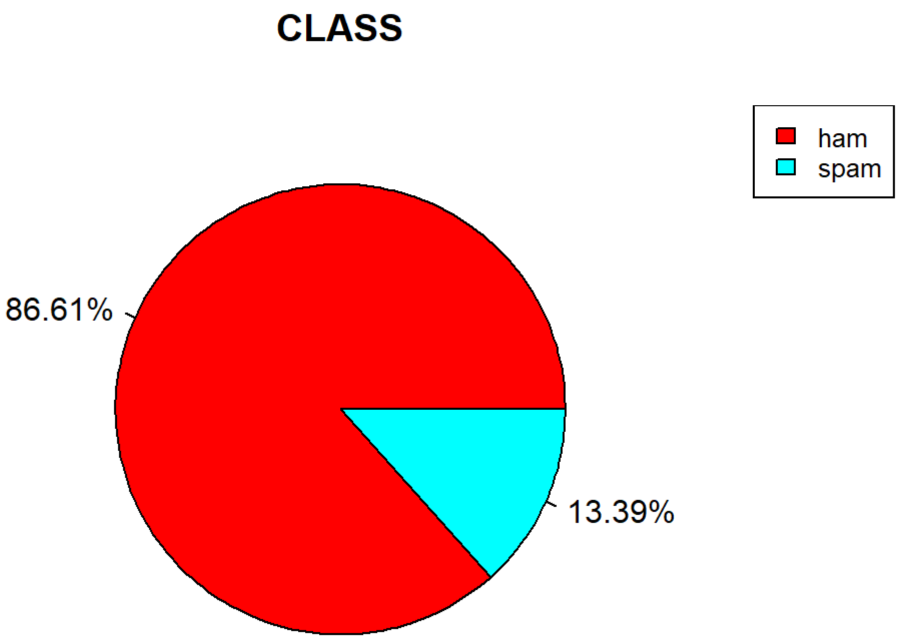
• “class” that assigns to each email a class: “spam” or “ham”.

The purpose of this analysis is to find the logistic regression model that best predicts the class of the email. How we can see from the pie plot, the Dataset is unbalanced.

Over the 4457 observations, we know that:

• 3860 observations that are classified as “ham”

• 597 observations that are classified as “spam”



Adding Variables

So, in order to better predict the “class” of the email, is possible looking for features that allow as to create and adding new variables, maybe usefull for ours predictive models. In my opinion, the best variables, that better help us to predict the class of the emails are:

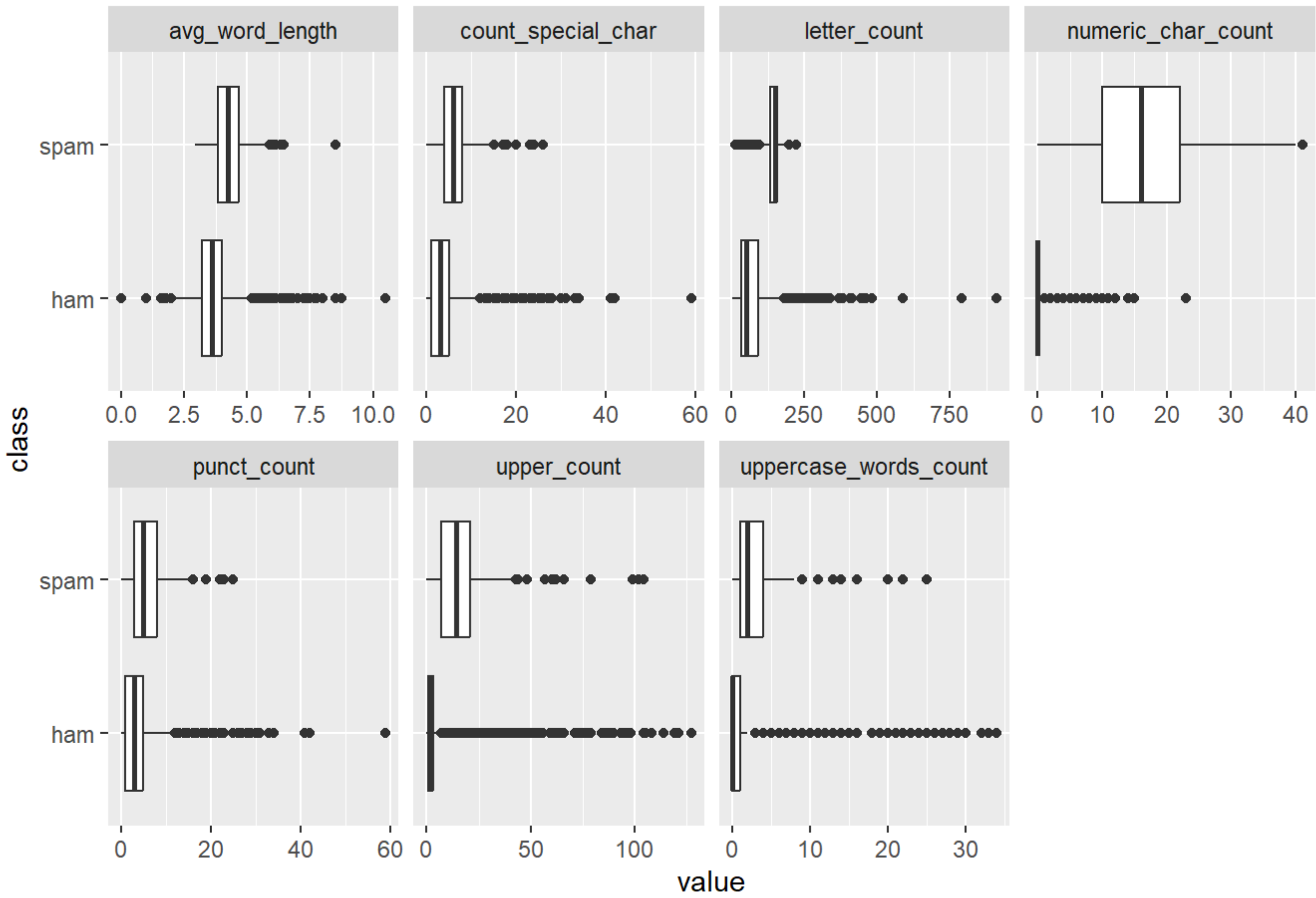
Show 25 **entries** **Search:**

	Variable	Class	Description
1	email	character	text of the emails
2	class	factor	Variable with two levels: spam and ham
3	keyword_present	factor	Variable which detects the precense of Keywords
4	n_digit_in_email	factor	Variable that detects if a number composed by 4 elements occur
5	cont_numb	factor	Variable that detects if a number is contained in the emails
6	special_char	factor	Variable that detects if one of this special characters [�$°§£€] occurs
7	upper_count	numeric	Variable that counts upper case
8	avg_word_length	numeric	Variable measures the average word length
9	numeric_char_count	numeric	Variable that counts the frequency of number for each email
10	letter_count	numeric	Variable to measure the number of letters that occurs
11	punct_count	numeric	Variable that counts the frequency of punctuation
12	count_special_char	numeric	Variable that counts the frequency of special characters
13	uppercase_words_count	numeric	Variable that counts the upper words

Showing 1 to 13 of 13 entries Previous 1 Next

Bivariate Analysis

BoxPlots



According to those boxplot the most usefull variables for prediction of “class” are:

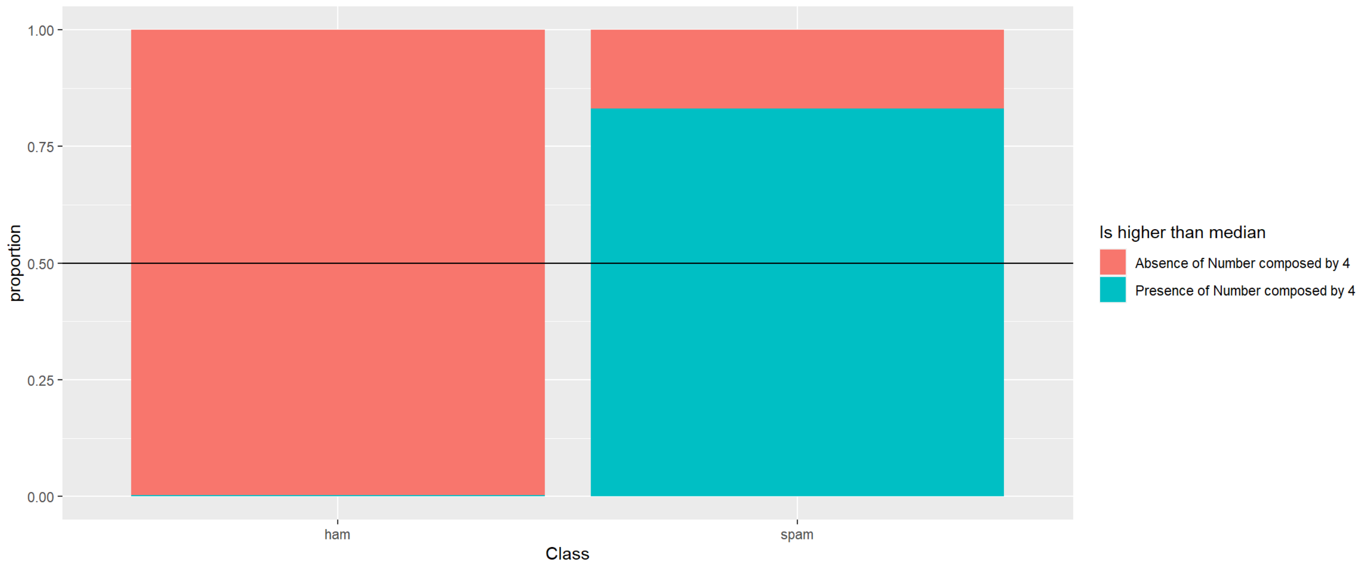
- “letter_count”
- “numeric_char_count”
- “upper_count”
- “uppercase_word_count”

While, in the other case seems that the medians are similar and there are many overlapping values between that groups.

Class X Keywords Presence

In this case the presence of Key word is more higher in the emails classified as “spam”

Class X N Digit in Emails



In this case the presence of number composed by 4 digit is more higher in the emails classified as “spam”

Class X Containing Numbers

In this case the presence of numbers is more higher in the emails classified as “spam”.

Splitting the Train Set

In order to better improve my model, Idecide to split the Train set in two sets:

- the “train” set, that now contains only the 80% of the originals observations
- the “train_validation” set, that contains the 20% of the originals observations

```
#splitting train
set.seed(8052023)
train_idx <- createDataPartition(train$class, times = 1, p = 0.8, list = FALSE)
train <- train[train_idx, ]
#train_validation set
train_validation <- train[-train_idx, ]
```

Logistic Regression

```
##
## Call:
## glm(formula = class ~ . - special_char + numeric_char_count:cont_numb -
##     numeric_char_count - cont_numb + avg_word_length:letter_count -
##     letter_count + count_special_char:special_char:punct_count +
##     punct_count:count_special_char - punct_count - count_special_char +
##     uppercase_words_count:upper_count - uppercase_words_count,
##     family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.6760  -0.1357  -0.0935  -0.0630   3.5092
##
## Coefficients:
##                                              Estimate Std. Error z value
## (Intercept)                                -11.895543   1.002759 -11.863
## upper_count                                  0.173654   0.022310   7.784
## keyword_present                              1.837193   0.349999   5.249
## avg_word_length                              0.721758   0.150010   4.811
## n_digit_in_email                             1.356308   0.609458   2.225
## cont_numb:numeric_char_count                 0.277518   0.030917   8.976
## avg_word_length:letter_count                 0.002759   0.000523   5.275
## punct_count:count_special_char              -0.022596   0.005459  -4.139
## upper_count:uppercase_words_count           -0.009849   0.001594  -6.177
## punct_count:count_special_char:special_char   0.008197   0.003441   2.382
##                                            Pr(>|z|)
## (Intercept)                                 < 2e-16 ***
## upper_count                                7.04e-15 ***
## keyword_present                            1.53e-07 ***
## avg_word_length                            1.50e-06 ***
## n_digit_in_email                             0.0261 *
## cont_numb:numeric_char_count                < 2e-16 ***
## avg_word_length:letter_count               1.32e-07 ***
## punct_count:count_special_char             3.49e-05 ***
## upper_count:uppercase_words_count          6.53e-10 ***
## punct_count:count_special_char:special_char   0.0172 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2810.02  on 3565  degrees of freedom
## Residual deviance:  447.97  on 3556  degrees of freedom
## AIC: 467.97
##
## Number of Fisher Scoring iterations: 8
```

```
## fitting null model for pseudo-r2
```

```
##  McFadden
## 0.8405822
```

Measure the variable importance using the function varImp()

```
##                                              Overall
## upper_count                                 7.783807
## keyword_present                             5.249140
## avg_word_length                             4.811387
## n_digit_in_email                            2.225433
## cont_numb:numeric_char_count                8.976351
## avg_word_length:letter_count                5.275364
## punct_count:count_special_char              4.139166
## upper_count:uppercase_words_count           6.177132
## punct_count:count_special_char:special_char 2.382048
```

Measure the multicollinearity using function vif()

```
##                                 upper_count
##                                    4.650647
##                             keyword_present
##                                    1.066763
##                             avg_word_length
##                                    1.080327
##                            n_digit_in_email
##                                    1.558754
##                cont_numb:numeric_char_count
##                                    1.699624
##                avg_word_length:letter_count
##                                    1.568890
##              punct_count:count_special_char
##                                   10.191461
##           upper_count:uppercase_words_count
##                                    4.266576
## punct_count:count_special_char:special_char
##                                    8.476860
```

Predictions using Train_validation

```
## Accuracy =  0.9859551
```

```
##       class.pred
##          0   1 Sum
##   ham  614   4 618
##   spam   6  88  94
##   Sum  620  92 712
```

```
## Specificity =  0.9935275
```

```
## Sensitivity =  0.9361702
```

Prediction on the Test Set

Now i can show the prediction:

```
## class.pred_test
##  ham spam
##  979  135
```