Spam_report

Yulia_Volkova

12/01/2017

Task Description

In this report we will build a classifier to separate spam and non-spam emails data.

Data Pre-processing

```
setwd("~/Desktop/Task")
spam_data <- read.csv("Features (Volkova).csv - features_220616_no_MEs.csv.csv")
```

To find a solution of a given problem I used a dataset "Features (Volkova).csv - features_220616_no_MEs.csv.csv" that contains 50 variables and 15157 observations.

```
glimpse(spam_data)
```

```
## Observations: 15,157
## Variables: 50
## $ id                                                <fctr> ++0lEwqA|E...
## $ EmailServiceInSubjectFeatureType                  <lgl> FALSE, FALS...
## $ OfficeAttachmentFeatureType                       <int> 0, 0, 1, 0,...
## $ count_TopContactArtifactMI_Content                <int> 0, 0, 0, 0,...
## $ count_EmoticonMI_AbstractMessage                  <int> 0, 0, 0, 0,...
## $ EmailForwardingFeatureType                        <lgl> FALSE, FALS...
## $ EmailShortResponseToSubjectQuestionFeatureType    <lgl> FALSE, FALS...
## $ DayOfWeekFeatureType                              <int> 2, 2, 3, 6,...
## $ count_SentenceNoFinalPunctuationMI_AbstractMessage <int> 1, 1, 0, 0,...
## $ INFORMALGreetingMI_AbstractMessage                <lgl> TRUE, TRUE,...
## $ EmailEmptyBodyFeatureType                         <lgl> FALSE, FALS...
## $ markSequence                                      <int> 0, 0, 0, 0,...
## $ EmailInternetSlangFeatureType                     <int> 0, 0, 0, 0,...
## $ TimeOfWeekFeatureType                             <int> 139249, 139...
## $ EmailResponseWithAttachmentFeatureType            <lgl> FALSE, FALS...
## $ EmailResponseFromMobileFeatureType                <lgl> FALSE, FALS...
## $ RawDataTypeFeature                                <fctr> EMAIL, EMA...
## $ EmailSentenceCountFeatureType                     <int> 5, 5, 0, 13...
## $ EmailEmptySubjectResponseFeatureType              <lgl> FALSE, FALS...
## $ INFORMALClosingMI_AbstractMessage                 <lgl> TRUE, TRUE,...
## $ EmailEmptySubjectFeatureType                      <lgl> FALSE, FALS...
## $ hasNotArtifact_ServiceMessageMI                   <lgl> TRUE, TRUE,...
## $ hasNotArtifact_GreetingMI                         <lgl> FALSE, FALS...
## $ EmailBodyUrlReferenceFeatureType                  <lgl> FALSE, FALS...
## $ count_InterjectionMI_AbstractMessage              <int> 0, 0, 0, 0,...
## $ PhotoInAttachmentFeatureType                      <dbl> 0.0, 0.0, 0...
## $ EmailEntertainmentUrlFeatureType                  <int> 0, 0, 0, 0,...
## $ count_LowerCaseFirstWordMI_AbstractMessage        <int> 1, 1, 0, 0,...
## $ EmailLengthFeatureType                            <fctr> 235, 235, ...
```

```
## $ count_SecretiveBehaviourMI_AbstractMessage         <int> 0, 0, 0, 0,...
## $ nCount_UrgencyMI                                   <dbl> 0, 0, 0, 0,...
## $ FORMALGreetingMI_AbstractMessage                   <lgl> FALSE, FALS...
## $ EmailAttachmentWithoutBodyFeatureType              <lgl> FALSE, FALS...
## $ EmailNumberOfRecipientsFeatureType                 <int> 1, 1, 16, 1...
## $ EmailForeignLanguageFeatureType                    <lgl> FALSE, FALS...
## $ hasNotArtifact_ClosingMI                           <lgl> FALSE, FALS...
## $ VolumeFeatureType                                  <int> 1, 1, 1, 1,...
## $ count_LowerCaseSentenceMI_AbstractMessage          <int> 0, 0, 0, 0,...
## $ exclamationMark                                    <int> 0, 0, 0, 0,...
## $ ellipsisFrequency                                  <int> 1, 1, 0, 0,...
## $ nCount_SecretiveBehaviourMI                        <dbl> 0, 0, 0, 0,...
## $ TimeOfDayFeatureType                               <int> 52849, 5284...
## $ EmailSenderIdFeatureType                           <int> 219, 219, 5...
## $ nCount_ProfanityMI                                 <dbl> 0.00000000,...
## $ FORMALClosingMI_AbstractMessage                    <lgl> FALSE, FALS...
## $ InviteInAttachmentFeatureType                      <int> 0, 0, 0, 0,...
## $ count_UrgencyMI_AbstractMessage                    <int> 0, 0, 0, 0,...
## $ count_ProfanityMI_AbstractMessage                  <int> 0, 0, 0, 0,...
## $ EmailSubjectCapitalizationFeatureType              <lgl> NA, NA, NA,...
## $ noise_flag                                         <int> 0, 0, 0, 1,...
```

Mutate variables

The variables description shows that there is an integer variable that was written as factor, so it was translated into integer type. Also there were logical variables that needed to be translated into factor type for convenience and finally "days of the week" variable was translated into an ordered factor and the target variable was made a factor as well. Moreover, the "id" variable was excluded from the data as it is not useful for the model. It was then checked that there are no variables that always take a constant value.

```
spam_data <- mutate(spam_data,
                    EmailLengthFeatureType = as.numeric(EmailLengthFeatureType),
                    noise_flag = factor(noise_flag, labels = c("Non-spam",
"Spam")),
                    DayOfWeekFeatureType = factor(DayOfWeekFeatureType, ordered
= T))


spam_data[sapply(spam_data, is.logical)] <- spam_data[sapply(spam_data,
is.logical)] %>%
  mutate_all(.funs = factor)

spam_data <- select(spam_data, -id)

all(sapply(spam_data, function(x) length(unique(x))) > 1)

## [1] TRUE
```

Dealing with NA

The distribution of the missing values was analysed and it was found that there is a variable that almost fully consists of NA's. This varibale was excluded from the dataset. Moreover, almost every variable had two NA's and they too were excluded.

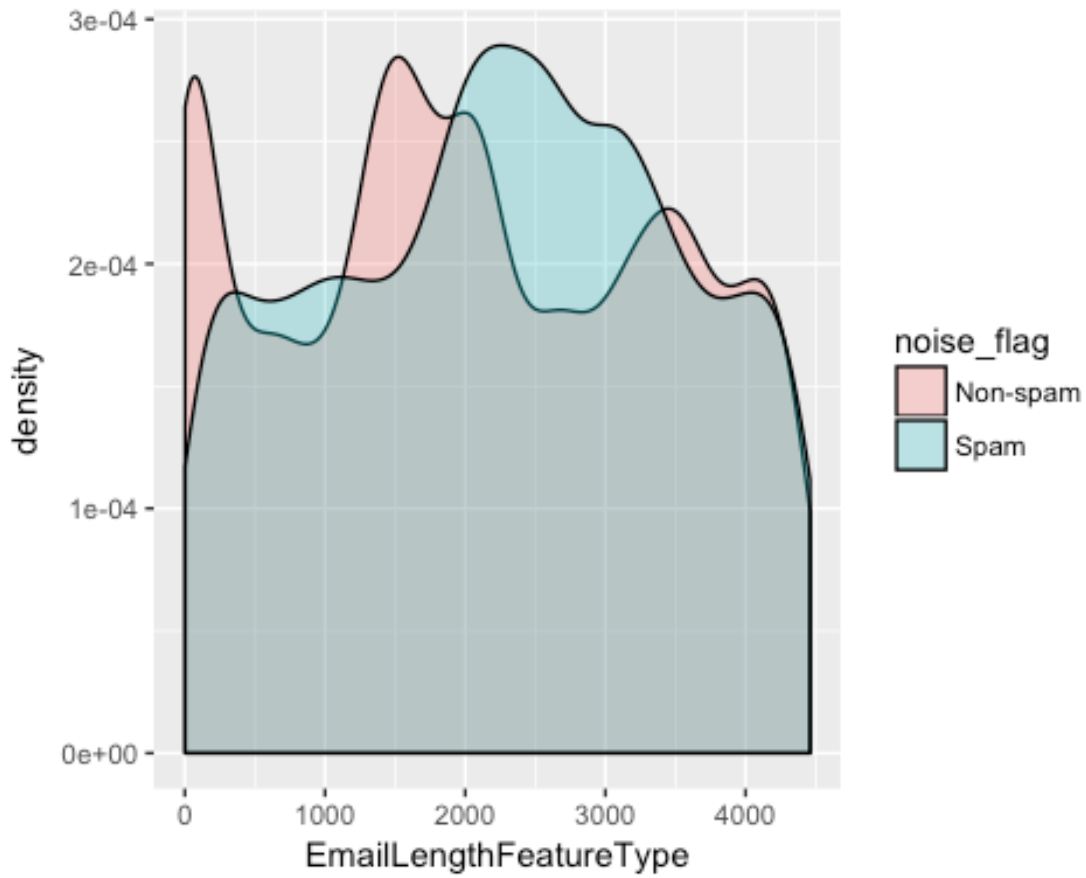
```
na_count <- summarise_all(spam_data, funs(sum(is.na(.))))
na_count$EmailSubjectCapitalizationFeatureType # the number of NA's in the
variable

## [1] 14144

spam_data <- select(spam_data, -EmailSubjectCapitalizationFeatureType)
spam_data <- na.omit(spam_data)
```

Hypothesis Testing

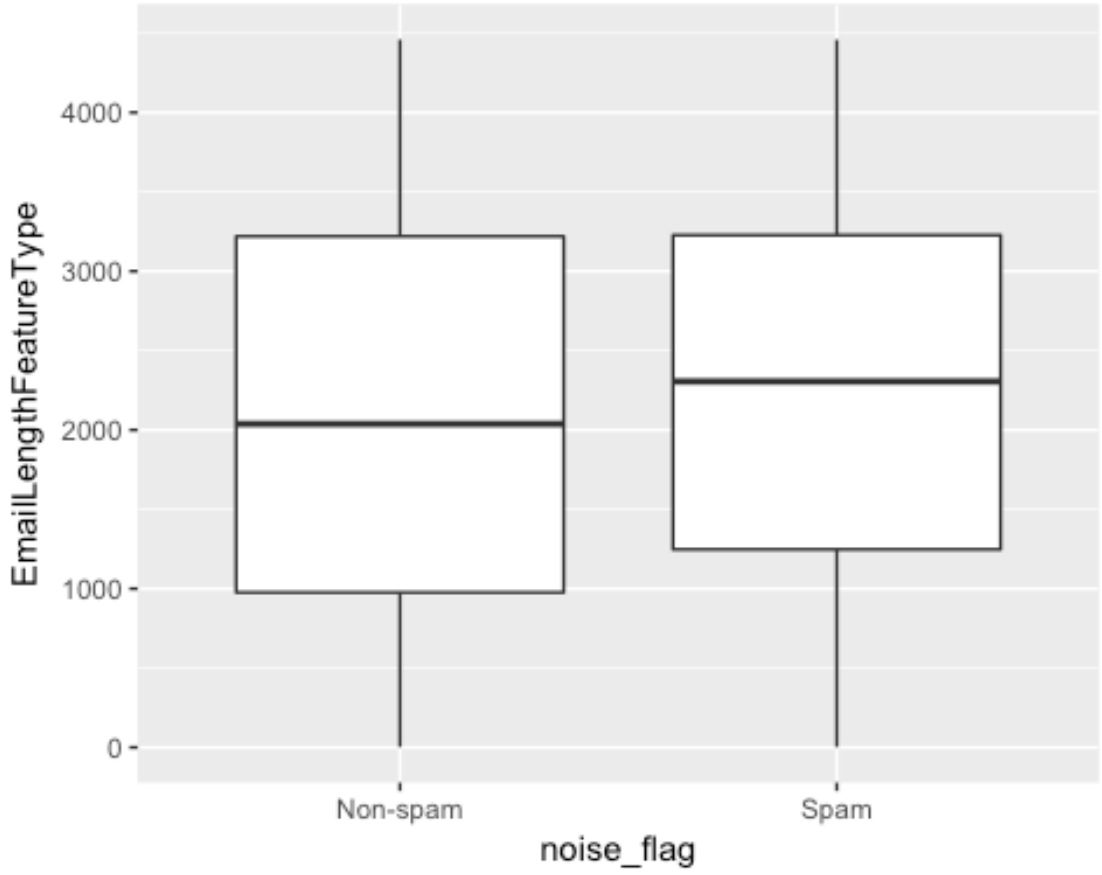
Hyp 1

Here several hypothesis were tested in order to see which variables allow us to separate out spam and non-spam data. Firstly, the emails with and without spam were compared by email length.

```
ggplot(spam_data, aes(x = EmailLengthFeatureType,
                      fill = noise_flag)) +
  geom_density(alpha = 0.3)
```



```
ggplot(spam_data, aes(x = noise_flag, y = EmailLengthFeatureType)) +
  geom_boxplot()
```



From the plotted data it can be seen that spam letters have bigger length. The non-parametric test was used for statistical testing of this hypothesis. (This is because the distribution is clearly not normal).

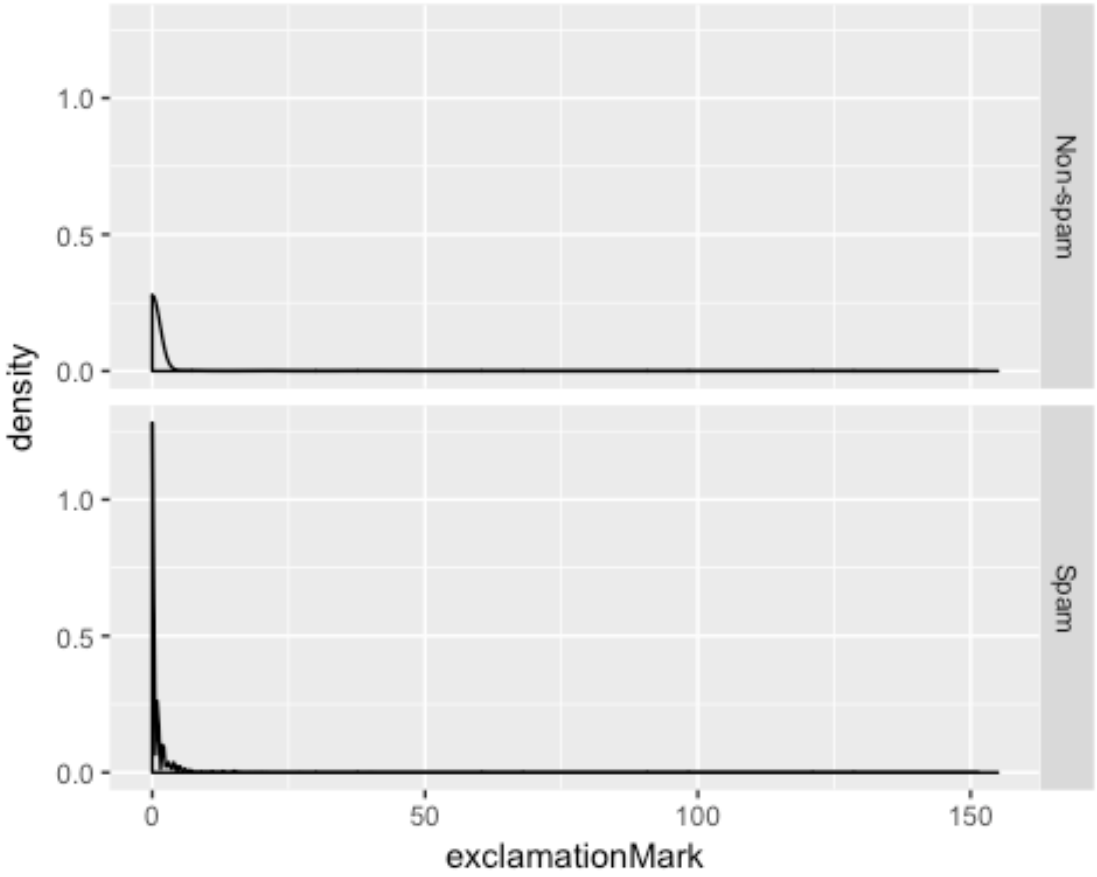
```
spam_data <- mutate(spam_data,
                    EmailLengthFeatureType = (EmailLengthFeatureType -
min(EmailLengthFeatureType)) /
                      (range(EmailLengthFeatureType)[2] -
range(EmailLengthFeatureType)[1]))
fit <- wilcox.test(EmailLengthFeatureType ~ noise_flag, spam_data)
fit

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  EmailLengthFeatureType by noise_flag
## W = 22157000, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0
```

The statistical results also confirm the hypothesis (p = 7.573243510^{-19}).

Hyp 2

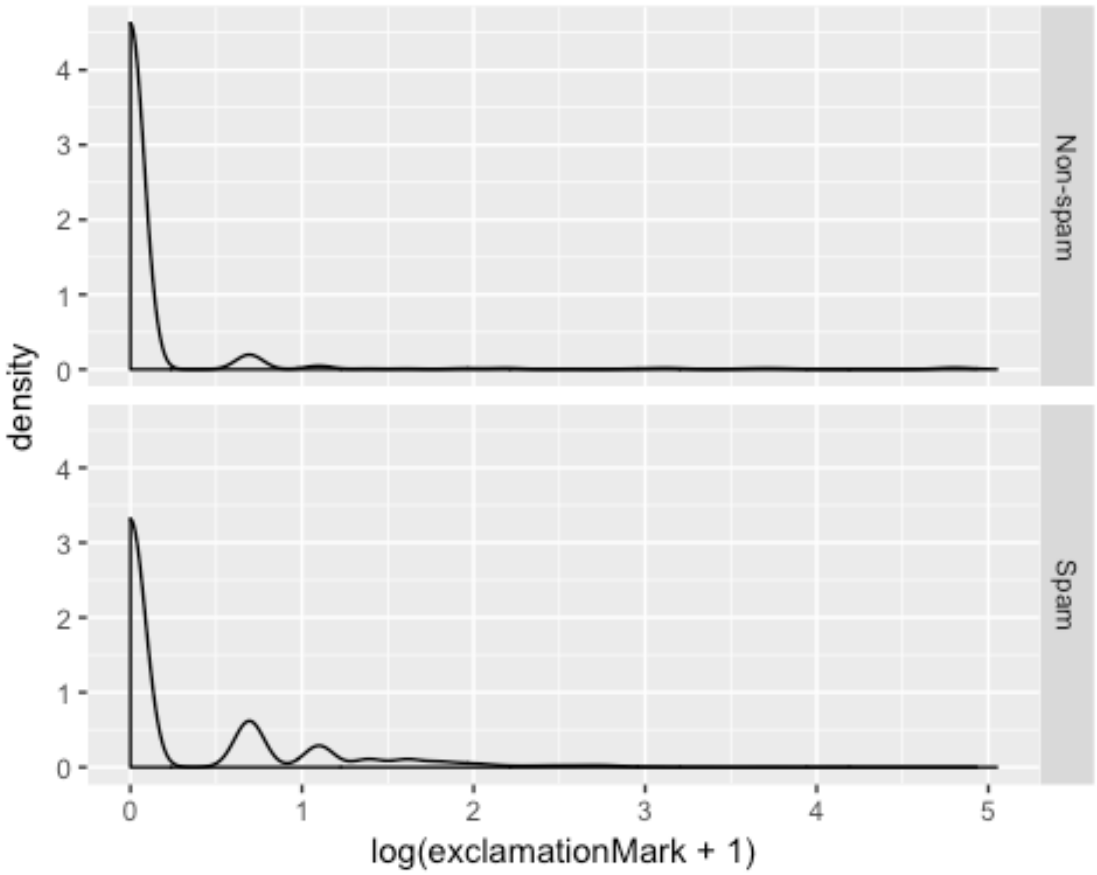
Secondly, the emails with and without spam were compared by the number of exclamation marks contained.

```
ggplot(spam_data, aes(x = exclamationMark)) +
  geom_density() +
  facet_grid(noise_flag ~ .)
```



The plot shows strong asymmetry and outliers. This makes statistical testing difficult. Log of the qualitative variable was taken to solve this problem.

```
ggplot(spam_data, aes(x = log(exclamationMark + 1))) +
  geom_density() +
  facet_grid(noise_flag ~ .)
```



Even though there is still some asymmetry the influence of outliers is diminished. Taking in consideration the large size of the sample non-parametric testing can be applied.

```
fit2 <- wilcox.test(exclamationMark ~ noise_flag, spam_data)
fit2

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  exclamationMark by noise_flag
## W = 19388000, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0
```

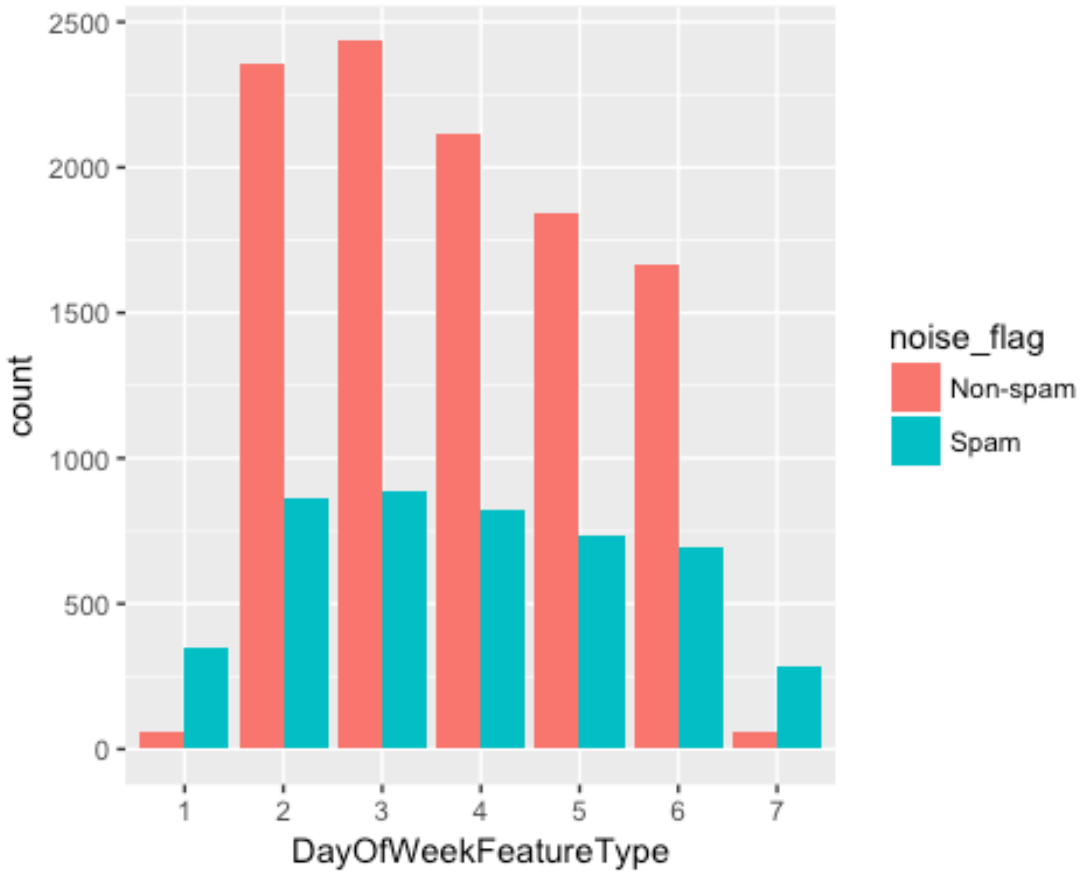
The statistical results also confirm the hypothesis (p = 6.93228910^{-243}).

Hyp 3

Finally, the dependence of the day of the week with spam/non-spam was tested.

```
table(spam_data$noise_flag, spam_data$DayOfWeekFeatureType)

##
##                1    2    3    4    5    6    7
##   Non-spam    62 2353 2433 2116 1845 1663   56
##   Spam       348  859  885  820  735  692  288
```

```
fit3 <- chisq.test(table(spam_data$noise_flag, spam_data$DayOfWeekFeatureType))
fit3

##
##  Pearson's Chi-squared test
##
## data:  table(spam_data$noise_flag, spam_data$DayOfWeekFeatureType)
## X-squared = 1090.7, df = 6, p-value < 2.2e-16

ggplot(spam_data, aes(x = DayOfWeekFeatureType, fill = noise_flag)) +
  geom_bar(position = "dodge")
```



On the plot it is seen that the distribution of normal letters and spam is dependent on the day of the week when the letter is sent. Spam letters are more frequently sent on weekends. It can be assumed that less of the normal letters are sent on the weekends whereas the distribution of the amount of spam letteres remains more or less constant throughtout the week.

To test this hypothesis the Chi-squared test was used. The results also confirm the hypothesis (p = 2.103972310^{-232}).

Summary

1. The exploratory analysis showed that the data has variables of different types, factor and numeric. The variable made of missing values was deleted from the data, the most obvious predictors such as "day of the week", "email length" and "exclamation marks

number" were proved to have significant influence on whether the email is a spam. The hypothesises were confirmed by statistical tests.

2. It was possible to single out several significant predictors. However, such an approach can hardly answer the question to which of the features are the most significant for the classifier solution. To clarify the problem we need to build a classifier and obtain a range for significance of the features.

3. The majority of count variables have a clear asymmetry and this allows to take the log of the variable to get more robust results.

- In addition the qualitative variables are presented in different scales and so we can scale them to 0-1 transformation:

$$z_i = \frac{x_i - min(x_i)}{Range(X)}$$

Classifier

Teach and Test Classifier

To solve the problem of binary classification the RandomForest algorithm was applied. To prevent overfitting the data was divided up into training and testing subsets.

```
set.seed(42)
split <- runif(nrow(spam_data)) > 0.2
train <- spam_data[split, ]
test <-  spam_data[!split,]

rf <- randomForest(noise_flag ~., train)

predictions <- predict(rf, test)
mean(test$noise_flag == predictions)

## [1] 0.9672131
```

The accuracy of the classifier on the testing subset equals 0.97. This is a relatively good result for the problem.

Variables Importance

In addition the given classifier outlined 10 most significant variables that allow to classify data as spam and non-spam:

```
var_importance <- as.data.frame(importance(rf))
var_importance$var_name <- row.names(var_importance)
var_importance <- arrange(var_importance, -MeanDecreaseGini)

slice(var_importance, 1:10)

##    MeanDecreaseGini                                var_name
## 1        1738.02664              EmailSenderIdFeatureType
## 2         719.16756          EmailSentenceCountFeatureType
## 3         627.01871     EmailNumberOfRecipientsFeatureType
```

```
## 4         185.35762                  EmailLengthFeatureType
## 5         183.63759 EmailResponseWithAttachmentFeatureType
## 6         176.05891                   TimeOfDayFeatureType
## 7         134.38689                  TimeOfWeekFeatureType
## 8         128.97002                        exclamationMark
## 9         126.90316      INFORMALClosingMI_AbstractMessage
## 10         85.83807                       ellipsisFrequency
```

Finally, lets deduce the metrics of classifier quality such as precision and recall.

```
tp <- sum(ifelse(test$noise_flag == "Spam" & predictions == "Spam", 1, 0))
fn <- sum(ifelse(test$noise_flag == "Spam" & predictions == "Non-spam", 1, 0))
fp <- sum(ifelse(test$noise_flag == "Non-spam" & predictions == "Spam", 1, 0))

precision <-  tp / (tp + fp)
recal <- tp / (tp + fn)
```

Precision equals 0.96, recall equals 0.93.

Conclusion

In this report a classifier to separate out spam and non-spam emails was built. Firstly, the data was pre-processed, so that the analysis could be performed. Then some statistical tests were carried out to see the influence of particular variables on the spam and non-spam classification of the emails. It was shown that hypothesis testing does not allow to deduce the most significant variables with regards to spam classification. Therefore, a RandomForest classifier was built. The overfitting problem was solved by partitioning the data into train and test subsets. RandomForest allowed for classification of a relatively high accuracy, 0.97. This means that in almost 97 out of 100 cases the classifier will correctly predict if the given email belongs to a class of spam emails.