Data Mining project - part I

Katarzyna Macioszek, Ada Majchrzak

November 29, 2023

1 Introduction

In this project, we will use data mining methods to perform analysis of *Spambase* dataset. The considered dataset was created from two collections of e-mails, the first being a set of non-spam personal and professional e-mails, and the second one coming from individuals who had filed spam. The e-mails were analyzed in terms of frequency of appearance of 54 selected words, numbers and characters, and also in terms of the number of capital letters present. Those statistics were then collected into table and each record was labeled as spam or non-spam. Our goal is to first extract interesting and insightful data characteristics by Exploratory Data Analysis procedures (plots, summary statistics), and then to perform a classification task, distinguishing spam from non-spam. For the second part, we will compare the performance of several methods, namely Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), K-Nearest Neighbours (KNN), and Random Forest. We will also evaluate those methods on different sets of features. The benefit from this kind of study could be obtaining a model effective in terms of spam filtering. We know that spam e-mails are a very common problem in modern world, and although every mailbox nowadays has built-in filter for that kind of content, non of them are completely accurate, so it is certainly a good idea to keep studying this topic.

2 Exploratory Data Analysis

Dataset description

The dataset consist of 4601 records, none of which contain any missing values. There are 57 numerical features and one categorical, *type*, being a target class. The target class takes two values – spam and nonspam. As we can see from Figure 1, the classes are quite well balanced. Around 60% of the records fall into nonspam category, and about 40% of them are spam. As for the numerical features, they can be divided into two groups – 54 features containing information about the frequency of selected words, numbers and special characters, expressed as percentage (further called Frequency features), and 3 features containing information about the capital letters in the e-mail (further called Capital features):

- capitalAve – average length of an uninterrupted sequence of capital letters,
- capitalLong – length of longest uninterrupted sequence of capital letters,
- capitalTotal – total number of capital letters.

Basic summary statistics for the numerical features can be found in Table 1. The maximum frequency found in the dataset equals 42.8, and it is observed for *num3d* variable. In general, the values for Frequency features are really low – in most of the cases, even the 3^{rd} quartile is very close to 0, or even equal to 0, which suggest that the identification of potential outliers might prove challenging. There are only two variables of that type for which the 3^{rd} quartile exceeds 1 – *you* (2.6) and *your* (1.3). Of course, this data stucture results in quite small standard deviations for the Frequency features. Regarding Capital features, here the range of values is significantly broader. The minimum for all three of those is 1, while the maximum is 1102.5 for *capitalAve*, 9989.0 for *capitalLong*, and 15841.0 for *capitalTotal*. As expected, also the standard deviations for this set of features are much higher compared to the previous one. Those differences need to be kept in mind and accounted for during the classfication tasks by appropriate data transformations.

Potential outliers

As we already mentioned, the distribution observed for the Frequency features is quite peculiar, with all three quartiles equal to zero in most cases. For that reason, finding the potential outliers migth be problematic – for sure the classical approach with $1.5IQR$ would not be a good idea here, as it could missclassify most of the nonzero observations as outliers. Moreover, given the characteristics of our data, even the unusually high frequencies don't necessarily need to be treated as outliers causing biased results in data analysis or

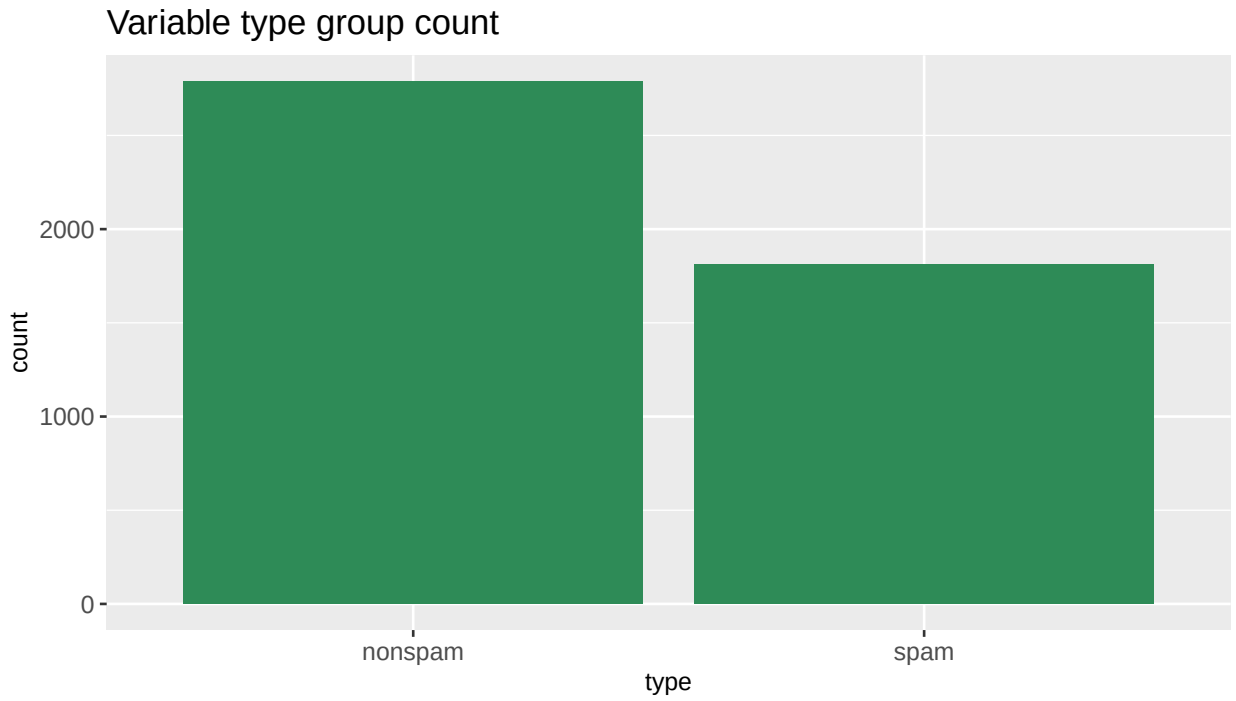

Figure 1: Target class distribution

classification task. On the contrary, we expect that they can be quite informative when trying to distinguish between spam and nonspam e-mails, as some of the words or characters will not be used much in private messages, but repeated frequently in spam, and vice versa. Considering all that, we will not try to identify and remove the potential outliers, but we will use boxplots to see how various transformations will influence data distribution and the number of detected potential oultiers. Of course, as we have 54 Frequency features, we will not show the boxplots for all of them. Instead, we picked three variables with the widest range of values - *george*, *num3d* and *charExclamation*. As for the data transformations, we will compare zscore standardization and $\log(x + 0.1)$ transformation.

Figure 2 shows the boxplots for *george* variable. We can see that for the untransformed data there are multiple outliers in the *nonspam* class, few outliers for *spam*, and the distribution for both classes is strongly concentrated around zero. For zscore it looks almost identical, but in the case of log transformation, we can see that the distribution spreads out a little – at least we can distinguish quartiles for the *nonspam* class, and the number of detected outliers decreases. The changes are even more visible when we consider only nonzero records. Here, again the untransformed data and zscore are very similar, and the log transformation in this case displays wider boxes with only two outliers visible for the *spam* class. A similar situation can be observed for the *num3d* (Figure 3) variable. Although the differences between the raw and transformed data are not that clear in case of all records, it becomes very obvious when we just consider the nonzero values. Zscore and untransformed data are again very similar, while for the log transformation we have very wide boxes and no outliers. Same observations can be drawn from Figure 4, showing the boxplots for *charExclamation* variable, although this time, we were not able to achieve a distribution without any outliers, even in the case of log transformation with nonzero records only. Still, this method clearly outperformed the zscore, resulting in much wider boxes and longer whiskers, with less values detected as outliers.

As for the Capital features, their distributions also characterize with having very low 3^{rd} quartile compared to maximum observed value. However, this time the 1^{st}, 2^{nd} and 3^{rd} quartile differ from one another, and the minimum value for all three variables is 1, so we don't need to consider the nonzero case while looking for potential outliers. Still, the distributions are very much concentrated around some relatively small value, with numerous outliers spread out above that value, so we'll consider exactly the same transformations for this set of features.

Let us first take a look at the *capitalAve* variable boxplots, shown on Figure 5. As in the previous cases, the log transformation stands out with wide boxes, but this time the number of outliers does not seem to be significantly decreased. For the *capitalLong* (Figure 6) and *capitalTotal* (Figure 7), the results of log transformation are even better, reducing the number of potential outliers compared to zscore and untransformed data.

To sum up, as the $log(x + 0.1)$ data transformation seems to work better in most cases than the zscore standardization in case of reshaping the feature distribution, we expect that the classifiers will perform better on such transformed data. However, we are going to validate it later in this report by comparing the performance of classification models using both methods.

	Mean	Std.Dev	Min	Max	Q1	Median	Q3
address	0.21	1.29	0.00	14.28	0.00	0.00	0.00
addresses	0.05	0.26	0.00	4.41	0.00	0.00	0.00
all	0.28	0.50	0.00	5.10	0.00	0.00	0.42
business	0.14	0.44	0.00	7.14	0.00	0.00	0.00
capitalAve	5.19	31.73	1.00	1102.50	1.59	2.28	3.71
capitalLong	52.17	194.89	1.00	9989.00	6.00	15.00	43.00
capitalTotal	283.29	606.35	1.00	15841.00	35.00	95.00	266.00
charDollar	0.08	0.25	0.00	6.00	0.00	0.00	0.05
charExclamation	0.27	0.82	0.00	32.48	0.00	0.00	0.32
charHash	0.04	0.43	0.00	19.83	0.00	0.00	0.00
charRoundbracket	0.14	0.27	0.00	9.75	0.00	0.06	0.19
charSemicolon	0.04	0.24	0.00	4.38	0.00	0.00	0.00
charSquarebracket	0.02	0.11	0.00	4.08	0.00	0.00	0.00
conference	0.03	0.29	0.00	10.00	0.00	0.00	0.00
credit	0.09	0.51	0.00	18.18	0.00	0.00	0.00
cs	0.04	0.36	0.00	7.14	0.00	0.00	0.00
data	0.10	0.56	0.00	18.18	0.00	0.00	0.00
direct	0.06	0.35	0.00	4.76	0.00	0.00	0.00
edu	0.18	0.91	0.00	22.05	0.00	0.00	0.00
email	0.18	0.53	0.00	9.09	0.00	0.00	0.00
font	0.12	1.03	0.00	17.10	0.00	0.00	0.00
free	0.25	0.83	0.00	20.00	0.00	0.00	0.10
george	0.77	3.37	0.00	33.33	0.00	0.00	0.00
hp	0.55	1.67	0.00	20.83	0.00	0.00	0.00
hpl	0.27	0.89	0.00	16.66	0.00	0.00	0.00
internet	0.11	0.40	0.00	11.11	0.00	0.00	0.00
lab	0.10	0.59	0.00	14.28	0.00	0.00	0.00
labs	0.10	0.46	0.00	5.88	0.00	0.00	0.00
mail	0.24	0.64	0.00	18.18	0.00	0.00	0.16
make	0.10	0.31	0.00	4.54	0.00	0.00	0.00
meeting	0.13	0.77	0.00	14.28	0.00	0.00	0.00
money	0.09	0.44	0.00	12.50	0.00	0.00	0.00
num000	0.10	0.35	0.00	5.45	0.00	0.00	0.00
num1999	0.14	0.42	0.00	6.89	0.00	0.00	0.00
num3d	0.07	1.40	0.00	42.81	0.00	0.00	0.00
num415	0.05	0.33	0.00	4.76	0.00	0.00	0.00
num650	0.12	0.54	0.00	9.09	0.00	0.00	0.00
num85	0.11	0.53	0.00	20.00	0.00	0.00	0.00
num857	0.05	0.33	0.00	4.76	0.00	0.00	0.00
order	0.09	0.28	0.00	5.26	0.00	0.00	0.00
original	0.05	0.22	0.00	3.57	0.00	0.00	0.00
our	0.31	0.67	0.00	10.00	0.00	0.00	0.38
over	0.10	0.27	0.00	5.88	0.00	0.00	0.00
parts	0.01	0.22	0.00	8.33	0.00	0.00	0.00
people	0.09	0.30	0.00	5.55	0.00	0.00	0.00
pm	0.08	0.43	0.00	11.11	0.00	0.00	0.00
project	0.08	0.62	0.00	20.00	0.00	0.00	0.00
re	0.30	1.01	0.00	21.42	0.00	0.00	0.11
receive	0.06	0.20	0.00	2.61	0.00	0.00	0.00
remove	0.11	0.39	0.00	7.27	0.00	0.00	0.00
report	0.06	0.34	0.00	10.00	0.00	0.00	0.00
table	0.01	0.08	0.00	2.17	0.00	0.00	0.00
technology	0.10	0.40	0.00	7.69	0.00	0.00	0.00
telnet	0.06	0.40	0.00	12.50	0.00	0.00	0.00
will	0.54	0.86	0.00	9.67	0.00	0.10	0.80
you	1.66	1.78	0.00	18.75	0.00	1.31	2.64
your	0.81	1.20	0.00	11.11	0.00	0.22	1.27

Table 1: Summary statistics for numerical features

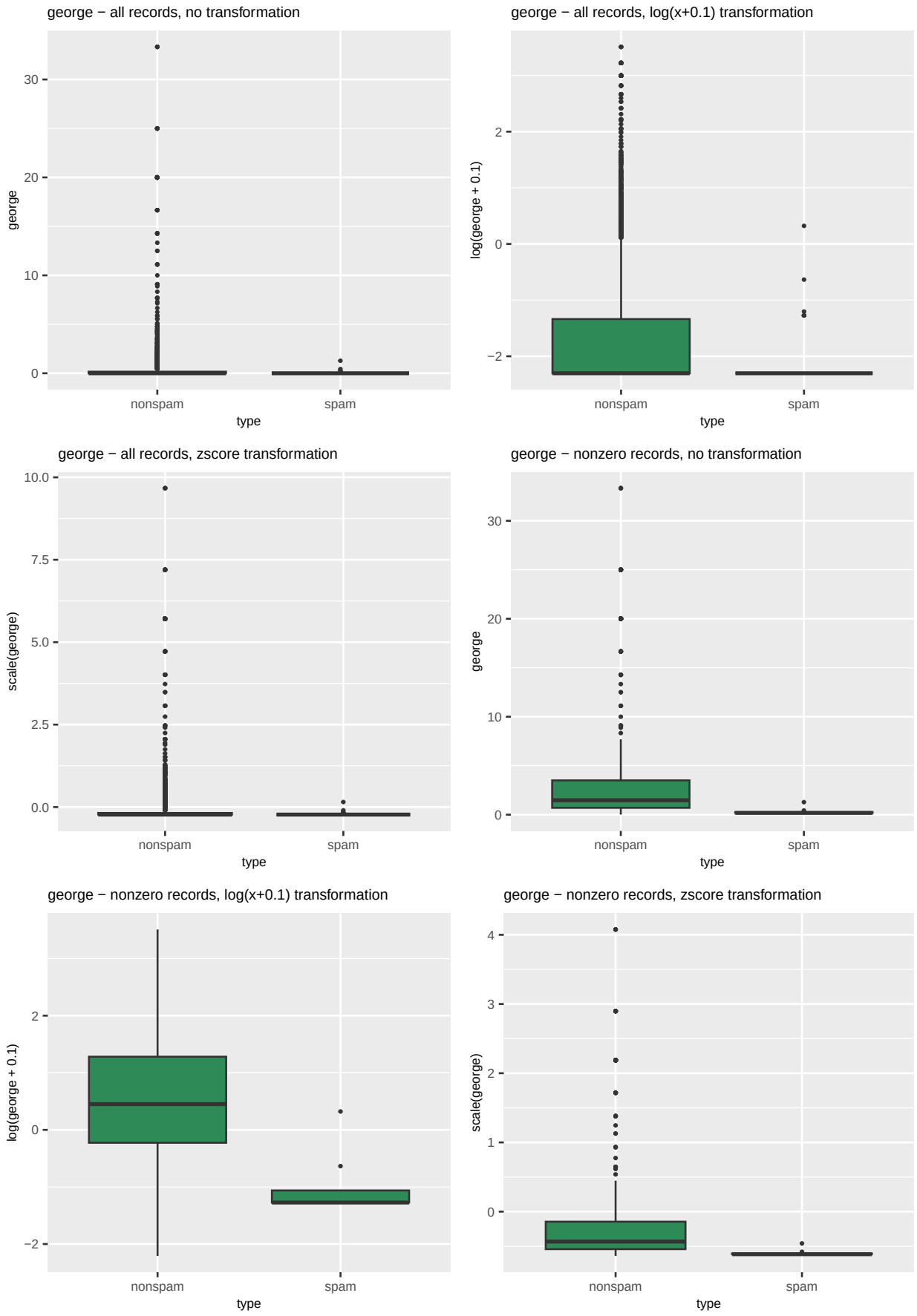

Figure 2: Boxplots for george variable

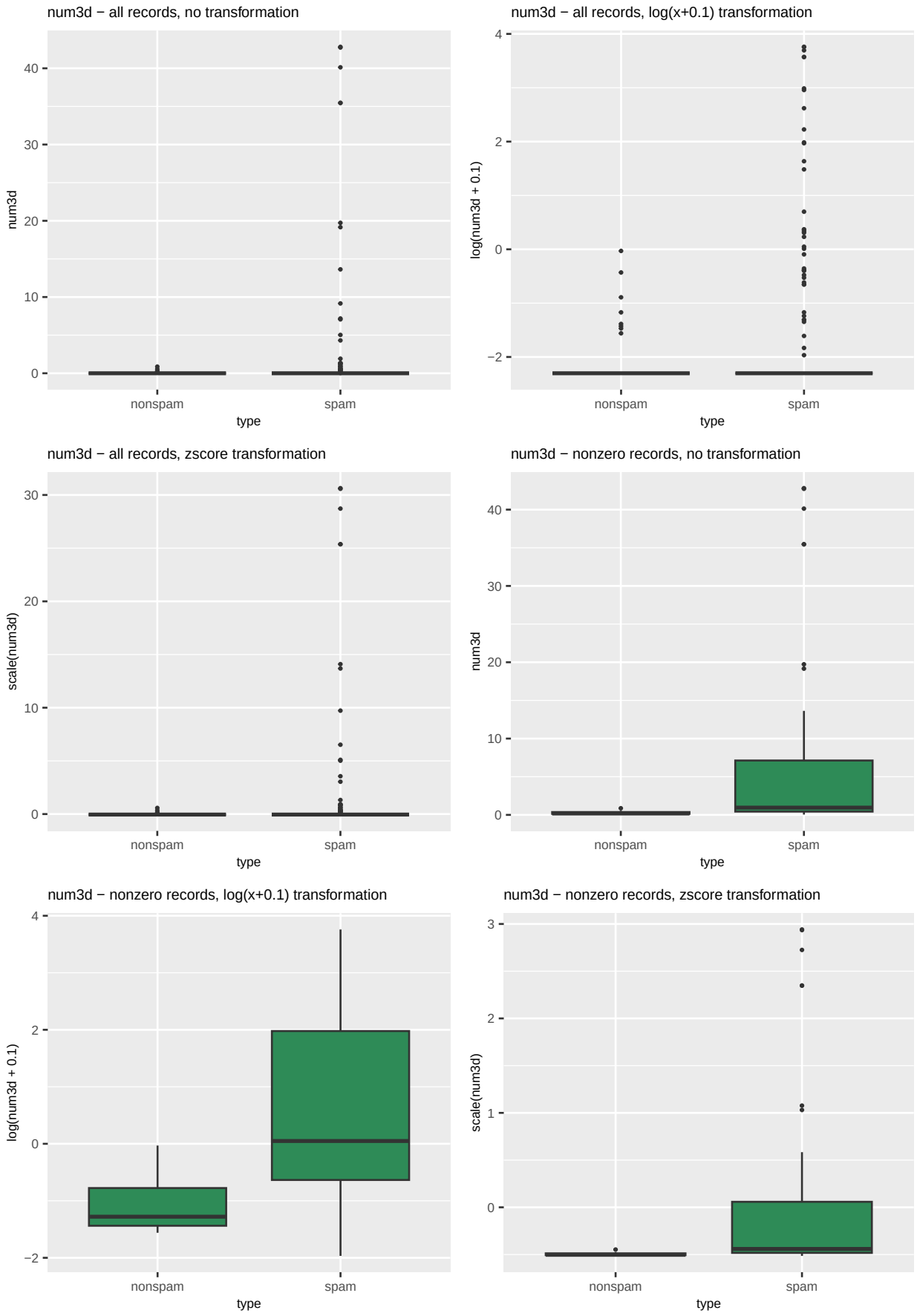

Figure 3: Boxplots for num3d variable

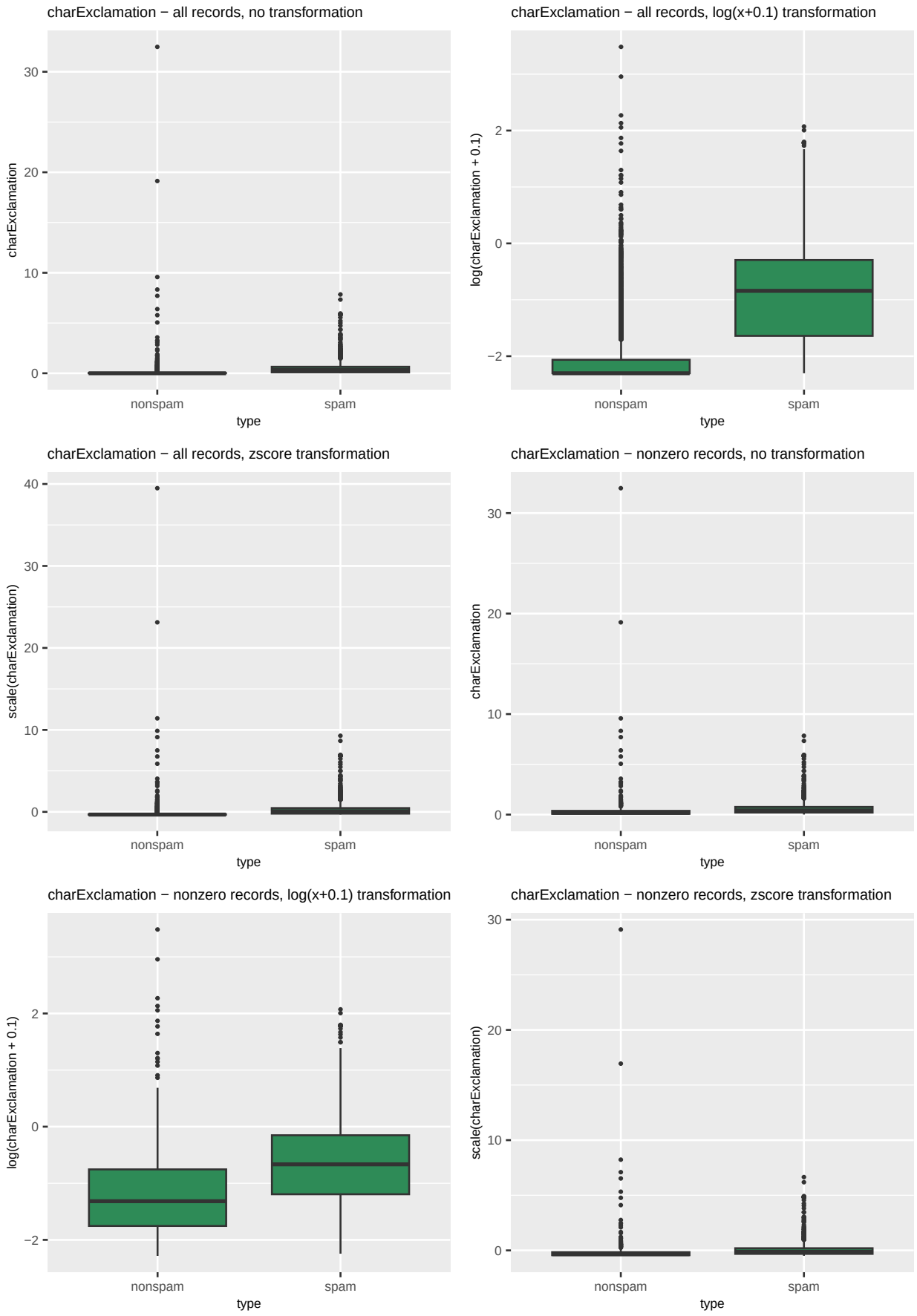

Figure 4: Boxplots for charExclamation variable

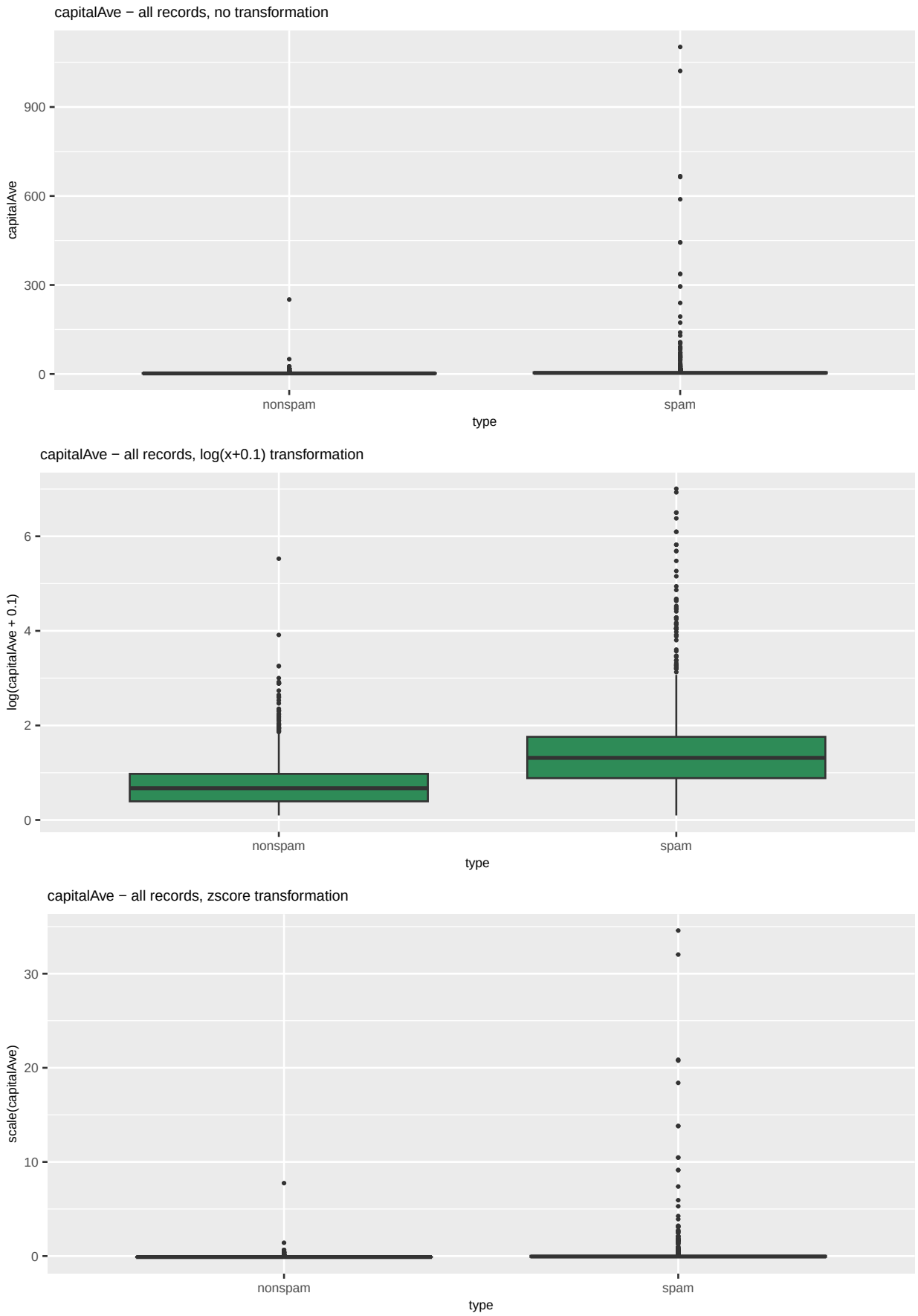

Figure 5: Boxplots for capitalAve variable

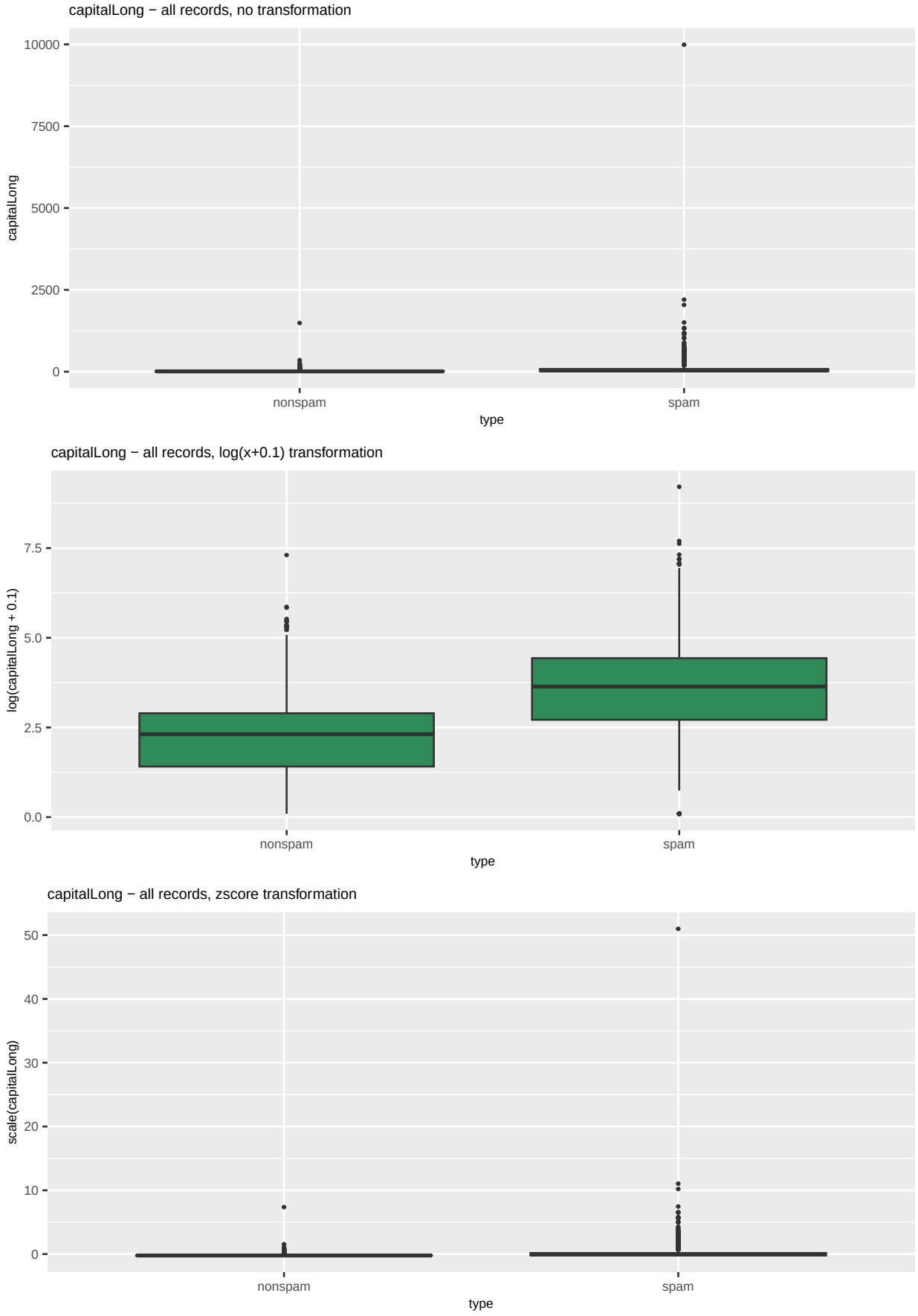

Figure 6: Boxplots for capitalLong variable

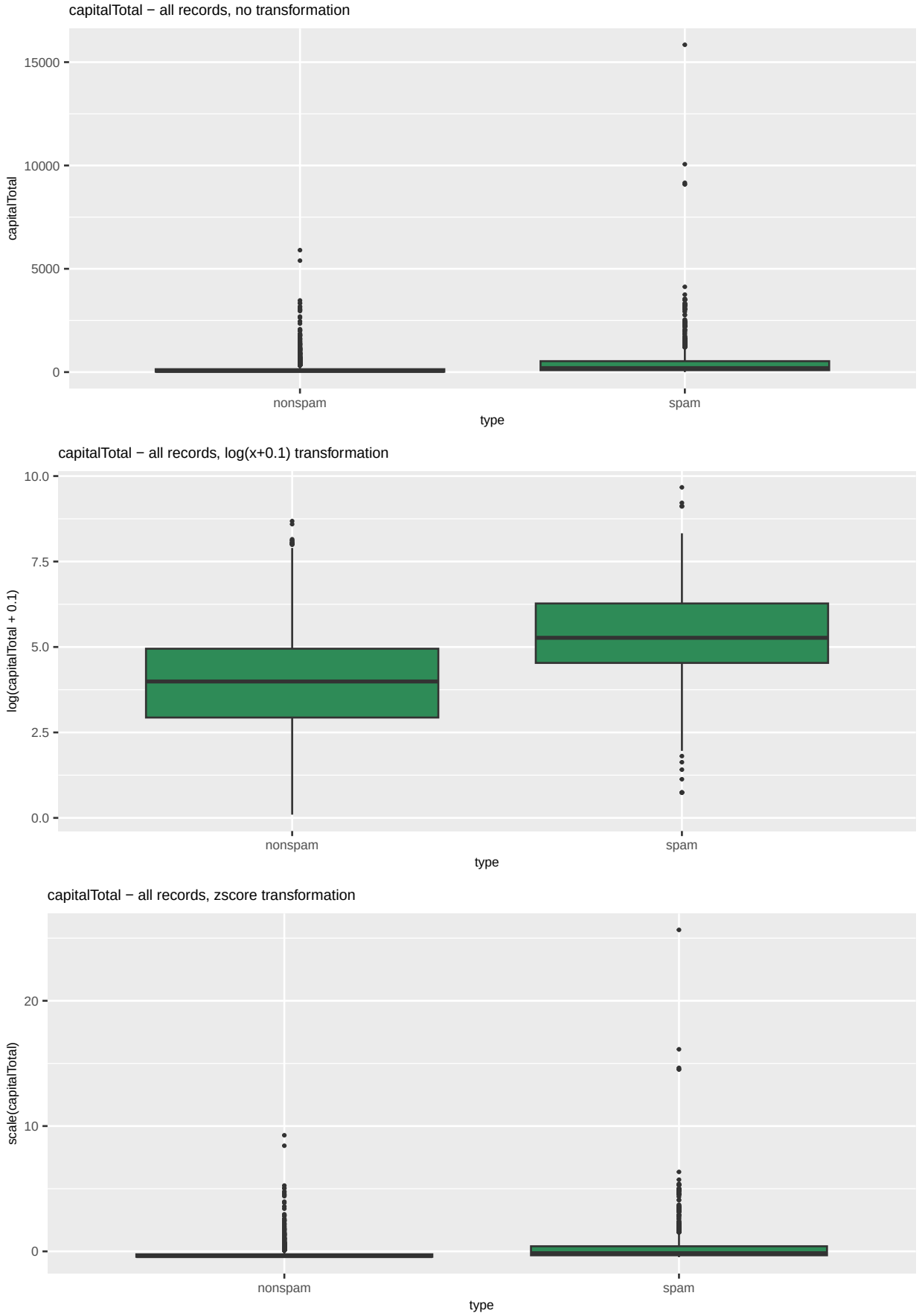

Figure 7: Boxplots for capitalTotal variable

Feature correlation

The spambase data set has as much as 57 features, so elimination of some of those may have a significant impact on efficiency of classification algorithms. The most common approach here is to look at feature correlations. This way we can remove some variables that are highly correlated with others, as they do not provide much more information in the classification problem. It is also faster and more convenient to use only one of such variables. First let us take a look at correlation matrix, that was presented on Figure 8 as it can show some interesting details about the data set.

From the correlation matrix we see that there are a few features highly correlated with one another. However generally the explanatory variables don't seem to be highly correlated, so we expect that not many variables can be skipped in the later analysis. To check that further we use function `findCorrelation` with cutoff equal to 0.8.

```
## Compare row 32  and column  34 with corr  0.996
##   Means:  0.143 vs 0.059 so flagging column 32
## Compare row 34  and column  40 with corr  0.845
##   Means:  0.127 vs 0.057 so flagging column 34
## All correlations <= 0.8
```

From the output we can see that with adopted cutoff we can eliminate features 'num415' and 'num857', so now we are left with 55 variables. This is not the only advantage that correlation matrix can give us, as we can also find variables that have the highest correlation with e-mail type. However earlier presented correlation matrix is not easy to read. Let us take out the features that have highest positive or negative correlation with type.

	correlation_with_spam
your	0.38
num000	0.33
remove	0.33
charDollar	0.32
you	0.27
free	0.26
business	0.26
hp	0.26
capitalTotal	0.25
our	0.24

Table 2: Features most correlated with target class variable

Results presented above are also visible on the matrix. It's important to point out that those features may have a great impact on classification of e-mails, which we will try to investigate later on.

Average frequency

Other important data feature is the average value. In the classification problem the average value among all variables is not as crucial as the difference in average word or character frequency between two investigated types. Let us take out variables, that demonstrate the highest difference in average value.

	type	george	you	your	hp	free	hpl	charExclamation	our	re	edu
1	spam	0.00	2.26	1.38	0.02	0.52	0.01	0.51	0.51	0.13	0.01
2	nonspam	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29

Table 3: Average frequency of variables across e-mail types

Here also we expect that, as for features highly correlated with *type*, those also will have crucial impact on spam classification.

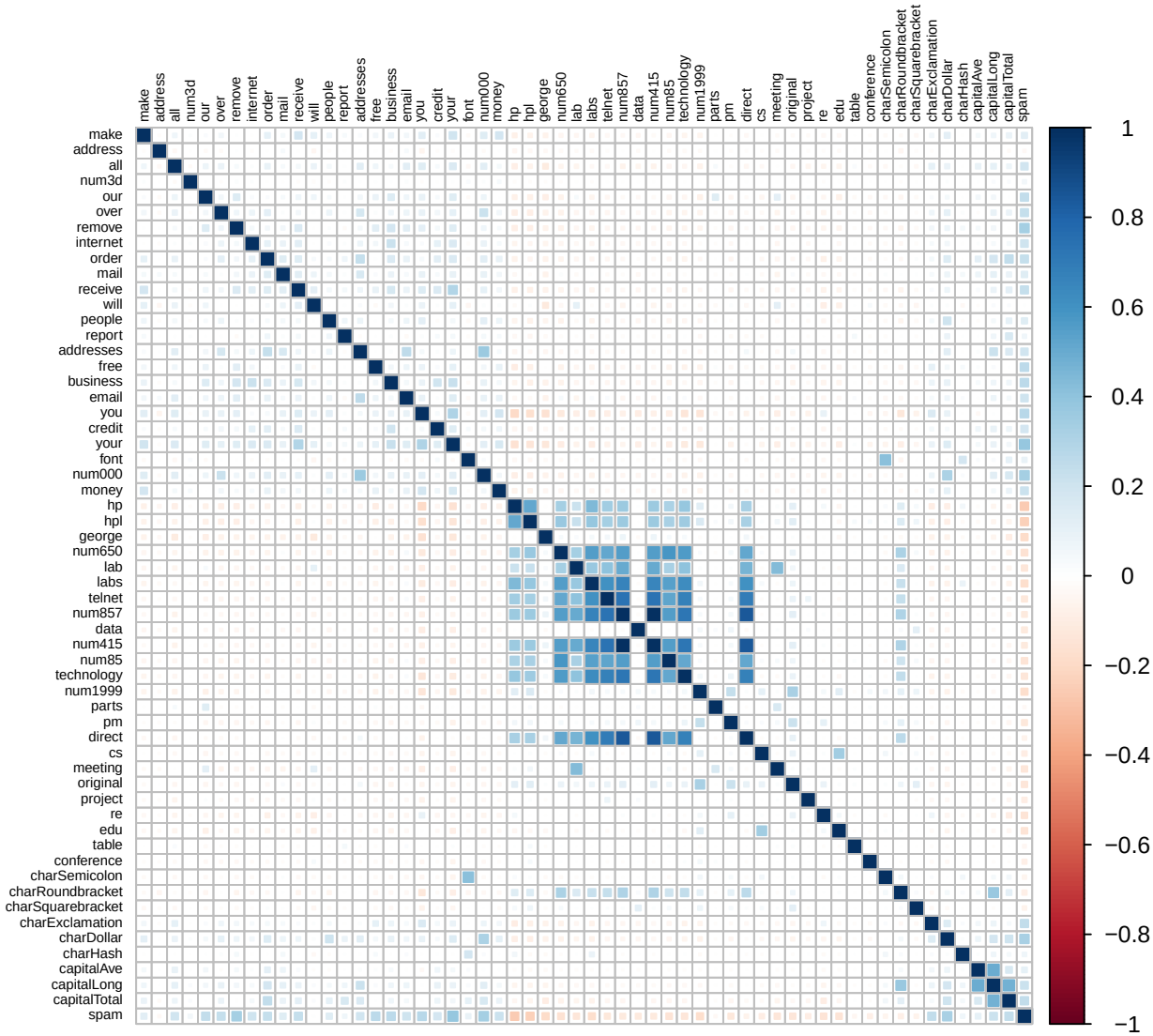

Figure 8: Correlation matrix of spam dataset

3 Classification

For the classification task, we will consider the complete dataset and several subsets of features and assess the accuracy for all of them. First, we want to check how well Frequency features and Capital features perform when used separately. Then, we will examine if feature reduction to the features highly correlated with *type* or those with highest difference in average frequency between classes will lead to satisfactory results. Apart from comparing different feature subsets, we will also inspect how the zscore scaling and $log(x + 0.1)$ transformation will affect classification outcome. Additionally, for the models that have the hyperparameters (KNN, Random Forest), we will tune them using the 5-fold cross validation algorithm.

KNN

Before approaching classification task with KNN, first we wanted to compute the average classification error for two types of data transformations, and for $k \in \{1, 2, 3, \ldots, 10\}$. To do that, we performed 100 independent runs of train-test data split, training, prediction and classification error calculation for each value of k and each transformed dataset. The results are displayed on Figure 9. We can see that the shape of both lines is similar, with maximum at $k = 2$, but the minimum for zscore is observed at $k = 1$, and for $log(x + 0.1)$ at $k = 5$, which means that, on average, $k = 2$ always performs the worst, but the best performing value of k depends on the data variant. Moreover, the error for log transformation is much smaller than for the zscore for all values of k, which suggests that we should rather use the log-transformed data for our classification task. Now we can check how KNN will perform with different sets of features, each time using 5-fold cross validation algorithm to tune the optimal value of k.

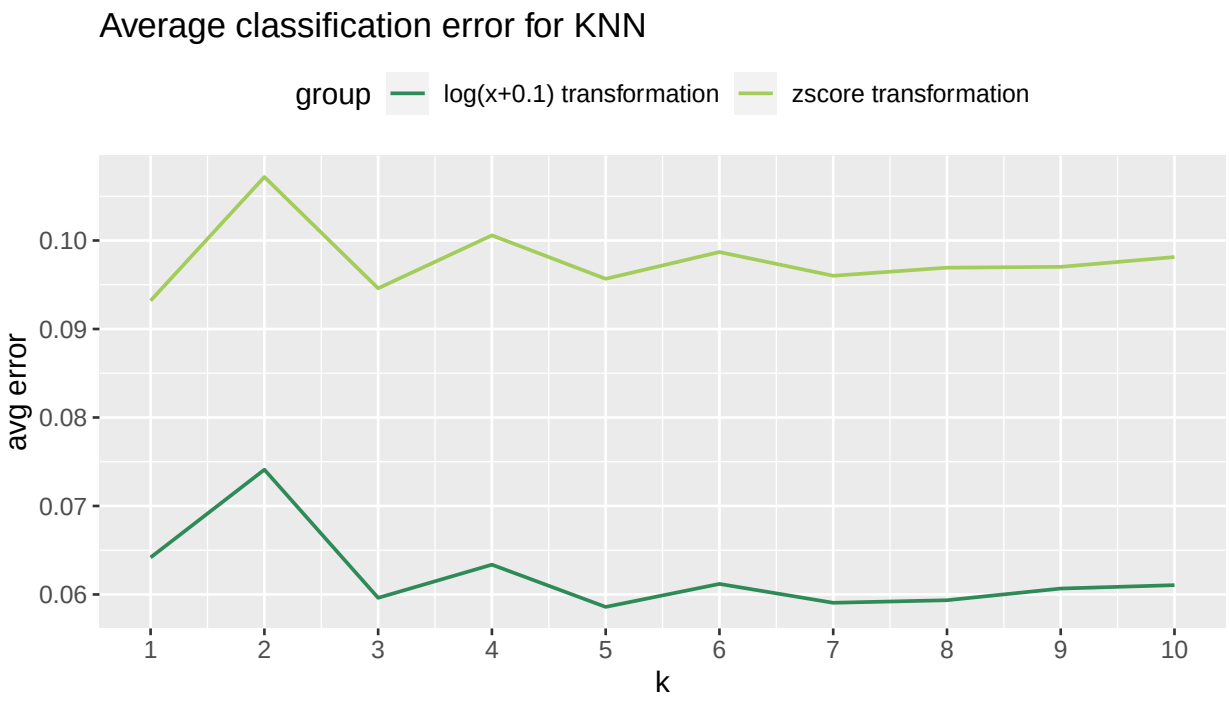

Figure 9: Average classification error for KNN

All features

Table 4 shows the confusion matrix for all features KNN classification. In this case, the algorithm picked $k = 3$ as the optimal hyperparameter. We see that most observations were classified correctly, resulting in 0.9391 accuracy, 0.9472 sensitivity and 0.9265 specificity, so the model performed slightly worse for the *nonspam* class (we treat *spam* as positive and *nonspam* as negative). Of course, the reason for that might be the slight class disproportion, as the model had less observations of *nonspam* to learn from.

	Reference	
Prediction	nonspam	spam
nonspam	664	33
spam	37	416

Table 4: Confusion matrix - KNN, all features

Frequency features

For Frequency features, we observe a worse performance, with 0.9278 accuracy, 0.9288 sensitivity and 0.9263 specificity. Again, the results are slightly better for the *spam* class. Here, the optimal k is 5.

	Reference	
Prediction	nonspam	spam
nonspam	665	32
spam	51	402

Table 5: Confusion matrix - KNN, Frequency features

Capital features

In the case of Capital features, the optimal value of k is 1, and the KNN model performance goes down significantly compared to the previous two sets of features. The observed accuracy is only 0.8009, while the sensitivity is 0.8391, definitely outperforming the specificity which is just 0.7435. We observe similar trend as in two cases described above, but this time much more obvious.

	Reference	
Prediction	nonspam	spam
nonspam	579	118
spam	111	342

Table 6: Confusion matrix - KNN, Capital features

Features highly correlated with *type*

Although it would seem that choosing the features highly correlated with target class variable would lead to very good classification results, this is not the case for our data. The confusion matrix looks very similar to that shown for the Capital features, and what follows, also the statistics are very close. This time we observe accuracy of 0.8052, sensitivity 0.8355 and specificity 0.7573. Optimal value of k is 1.

	Reference	
Prediction	nonspam	spam
nonspam	589	108
spam	116	337

Table 7: Confusion matrix - KNN, Features highly correlated with type

Features with highest average difference in frequency between types

The last set of features shows improvement over the previous two. Here, the accuracy is 0.8904, sensitivity 0.9050 and specificity 0.8674. It is definitely not as good as in the case where we included all features or those related to frequency, but the difference is not as significant as for the Capital features or the features highly correlated with *type*. The trend of better performance for the *nonspam* class is preserved, and the 5-fold CV algorithm picked $k = 8$.

Random Forest

In case of Random Forest, we will follow the same procedure as for KNN – first, we are going to look at the average classification error plot for both transformed datasets, this time with *mtry* parameter (number of features considered in each tree) on X-axis. The plot is shown on Figure 10. This time it's not so easy to tell which data transformation performed better. The lines intersect several times and none of them is clearly above the other. However, the $\log(x + 0.1)$ transformation resulted in smaller error around the optimal value of *mtry* ($\lfloor p \rfloor = 7$, p – number of features). For that reason, and to keep consistent with the classifiers discussed earlier, we will use the log-transformed data for the classification task with Random Forest.

All features

In case of all features Random Forest classification, the 5-fold CV algorithm picked 11 as the optimal value of *mtry*. The accuracy is 0.9513, which exceeds that of a KNN model, but not significantly (approximately 2% difference). The improvement is visible also with sensitivity (0.9520) and specificity (0.9501). Same as KNN, the model performed better for the *nonspam* class.

	Reference	
Prediction	nonspam	spam
nonspam	638	59
spam	67	386

Table 8: Confusion matrix - KNN, Features with highest average difference in frequency between types

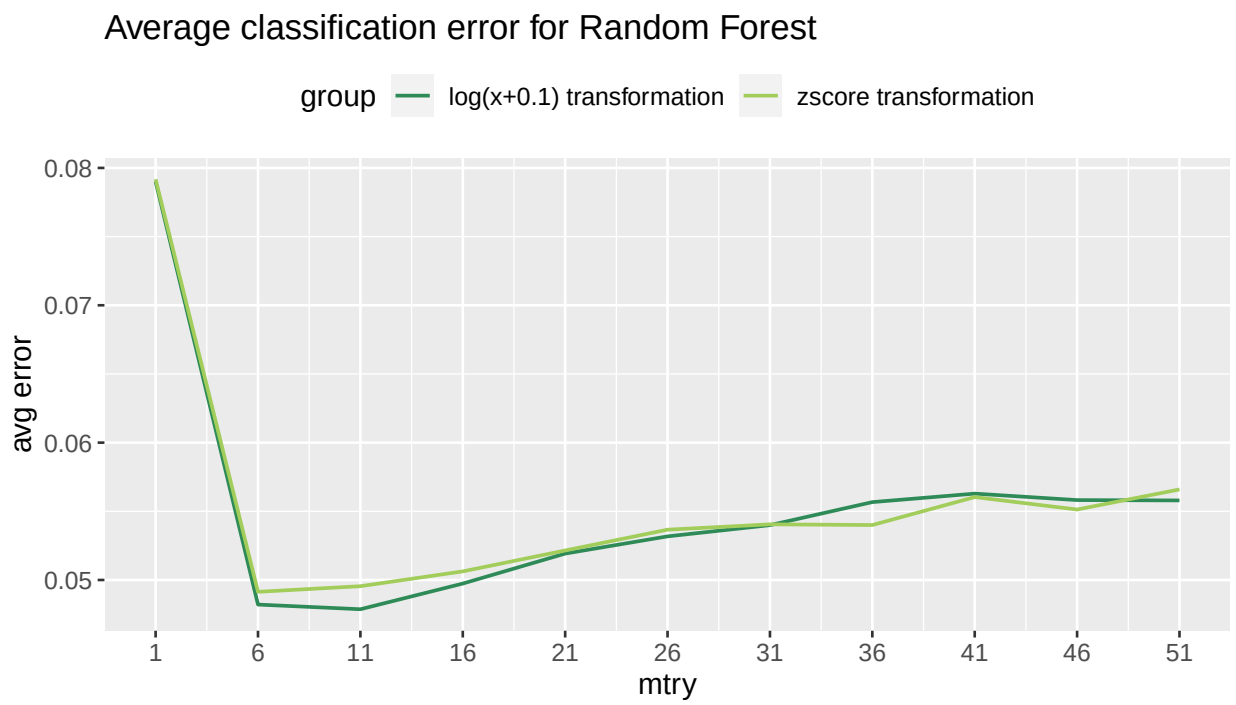

Figure 10: Average classification error for Random Forest

Frequency features

After reducing the dataset to Frequency features only, we observe a very small decrease in accuracy (0.9504). What's interesting, the sensitivity has improved slightly, reaching the value of 0.9533, while the specificity is worse than for all features example, being 0.9459. The optimal value of *mtry* is once again 11.

Capital features

For the Capital features, the optimal value of *mtry* turned out to be 1. We observe a significant decrease of the model performance in this case. The accuracy is 0.8243, the sensitivity is 0.8377 and the specificity is 0.8010. Those results are better than for the KNN run on the same set of features, but for sure we would not call them satisfying.

Features highly correlated with *type*

As opposed to KNN, in case of Random Forest we don't observe a significant decrease of model performance after selecting the features highly correlated with *type*. The accuracy here is 0.9287, the sensitivity is 0.9349, and the specificity is 0.9187. Those results are slightly worse than for all features or Frequency features, but still quite good. The optimal value of *mtry* is 3 this time.

Features with highest average difference in frequency between types

For the last set of features, the 5-fold CV algorithm picked 4 as the optimal value of *mtry*. The model performance is slightly worse than in case of the previous set, but definitely better than for the Capital features, giving accuracy of 0.9139. What's worth noticing, the trend favoring the *nonspam* class that we observed for all other cases, is reversed here. The sensitivity is 0.9096, while the specificity equals 0.9214.

	Reference	
Prediction	nonspam	spam
nonspam	675	22
spam	34	419

Table 9: Confusion matrix - Random Forest, all features

	Reference	
Prediction	nonspam	spam
nonspam	673	24
spam	33	420

Table 10: Confusion matrix - Random Forest, Frequency features

Discriminant analysis

All features

Another useful classification algorithm is discriminant analysis. In this section we will use linear and quadratic discriminant analysis to determine e-mail type on data transformed as earlier, so we compare scaled data and data transformed with logarithm. At first let us use these methods on the whole data set and examine the accuracy. To do it we use the confusion matrix, first for scaled data we have Table 14 and Table 15.

The accuracy for described methods is quite satisfying, for LDA it's 0.8904 and for QDA - 0.847. So in this case linear method works significantly better. There's also a disproportion in confusion matrices for those methods. For LDA there are more spam messages predicted as nonspam. Opposite situation occurs for QDA, which works in favor of spam e-mails. Those features are nicely represented with sensitivity and specificity, that for LDA are respectively 0.9098, 0.8802 and for QDA: 0.7336, 0.9677.

We perform the same algorithms on the data with logarithmic transformation and obtain Tables 16 and 17.

We see that this transformation helped our algorithm to obtain better accuracy: 0.9348 for LDA and 0.8574 for QDA. From that we conclude that other feature subsets will be tested for data after logarithmic transformation.

In the next part we try to build predictions using only some subsets of features. We start with using either only variables representing frequencies and those describing capital letters statistics.

Frequency features

We again provide the confusion matrices that describes the classification methods: Table 18 add Table 19.

Above results exhibit accuracy 0.9278 for LDA and 0.853 for QDA. We also observe similar disproportion in characteristics as for the previous case with 0.9363 sensitivity and 0.9229 specificity for LDA and respectively 0.7474 and 0.9583 for QDA. As we see elimination of features describing capital letters didn't have an enormous impact on discriminant analysis results.

Capital features

This time we take into account only three features containing information about capital letters and get Tables 20 and 21.

Here we obtain much worse accuracy: 0.7304 for LDA and 0.7148 - QDA. What's interesting is that this time the disproportion between wrong classification of different discriminant analysis algorithms is not that high and both methods are more likely to incorrectly classify spam as nonspam. This results in sensitivity 0.7061 and specificity 0.7410 for LDA and 0.7254, 0.7148 respectively for QDA.

Features highly correlated with type

LDA and QDA results are presented in Table 22 and Table 23.

Despite having much less explanatory variables than when taking all frequency features we obtain similar accuracy: 0.8939 - LDA and 0.88 - QDA. It's also important to note that QDA accuracy is better in this case than for any other feature subset tested before. This variable selection also omits the disproportions between two analyzed discriminant analysis methods. Such property results in similar statistics for both of them. Namely 0.9211 sensitivity and 0.8798 specificity for LDA, 0.8851 and 0.8772 respectively for QDA.

	Reference	
Prediction	nonspam	spam
nonspam	614	83
spam	119	334

Table 11: Confusion matrix - Random Forest, Capital features

	Reference	
Prediction	nonspam	spam
nonspam	661	36
spam	46	407

Table 12: Confusion matrix - Random Forest, Features highly correlated with type

features with highest average difference in frequency between types

Last feature selection that we test is for variables exhibiting highest difference in average frequency between types. Confusion matrices are shown in Table 24 ad Table 25.

In this case accuracy of LDA is still comparable to other feature subsets, namely 0.8887, however for QDA we get 0.793. It means that this feature subset didn't have much impact on linear algorithm, but for quadratic one it displays much worse accuracy. Also for QDA the disproportion between sensitivity and specificity is significant: 0.6646 and 0.9618 respectively. For LDA the disproportion is almost invisible: 0.877 - sensitivity and 0.8957 - specificity.

4 Conclusions

The analysis focused on spam e-mail classification, covering Exploratory Data Analysis, feature selection, and the evaluation of classification algorithms. Notable findings include the impact of various data transformations and the significance of choosing different feature subsets for classification models. All the algorithms were quite responsive to variable selection. On general, the worst performing subset was Capital features, while the best performance was achieved by using all explanatory variables and Frequency features. What's interesting, reducing the number of columns from 57 even to 10, usually didn't result in a significant drop in accuracy.

When it comes to algorithmic performance, KNN's sensitivity to data transformation was evident, with log transformation consistently outperforming z-score scaling for all values of k. On the contrary, for the Random Forest the difference between the two data transformations was not that clear, however the log-transformation was chosen as the better one as it displayed smaller classification error for the optimal value of *mtry* parameter. Random Forest's performance persistently exceeded that of KNN, LDA and QDA, achieving higher accuracy, sensitivity, and specificity for all feature subsets.

Discriminant analysis, contrary to KNN and Random Forest, exhibit high disproportions between specificity and sensitivity. For LDA, better sensitivity is observed in most cases, while for QDA the specificity was usually higher, especially in the case of all features. Again, the comparison of scaled and log-transformed data highlighted the effectiveness of log transformation in enhancing classification accuracy. Additionally, the QDA falls behind all other considered algorithms.

Above conclusions highlight the importance of strategic feature selection and data transformation for optimal classification results. For the future research, one could consider the deep learning approach for spam classification, such as various types of neural networks. On top of that, one could use the cost matrix to deploy the cost-sensitive learning approach. Of course, the challenge here could be determining which e-mail type misclassification should be assigned with higher cost. Additionally, the cluster analysis could be used to separate the data, which will be done in the next part of the project.

	Reference	
Prediction	nonspam	spam
nonspam	664	33
spam	66	387

Table 13: Confusion matrix - Random Forest, Features with highest average difference in frequency between types

	Reference	
Prediction	nonspam	spam
nonspam	661	36
spam	90	363

Table 14: Confusion Matrix - LDA, all features

	Reference	
Prediction	nonspam	spam
nonspam	539	158
spam	18	435

Table 15: Confusion Matrix - QDA, all features

	Reference	
Prediction	nonspam	spam
nonspam	675	22
spam	53	400

Table 16: Confusion Matrix - LDA, all features

	Reference	
Prediction	nonspam	spam
nonspam	558	139
spam	25	428

Table 17: Confusion Matrix - QDA, all features

	Reference	
Prediction	nonspam	spam
nonspam	670	27
spam	56	397

Table 18: Confusion Matrix - LDA, frequency features

	Reference	
Prediction	nonspam	spam
nonspam	552	145
spam	24	429

Table 19: Confusion Matrix - QDA, frequency features

	Reference	
Prediction	nonspam	spam
nonspam	595	102
spam	208	245

Table 20: Confusion Matrix - LDA, capital features

	Reference	
Prediction	nonspam	spam
nonspam	619	78
spam	247	206

Table 21: Confusion Matrix - QDA, capital features

	Reference	
Prediction	nonspam	spam
nonspam	666	31
spam	91	362

Table 22: Confusion Matrix - LDA, features correlated with type

	Reference	
Prediction	nonspam	spam
nonspam	650	47
spam	91	362

Table 23: Confusion Matrix - QDA, features correlated with type

	Reference	
Prediction	nonspam	spam
nonspam	644	53
spam	75	378

Table 24: Confusion Matrix - LDA, features with highest average difference in

	Reference	
Prediction	nonspam	spam
nonspam	478	219
spam	19	434

Table 25: Confusion Matrix - QDA, features with highest average difference in