MACHINE LEARNING

MSC DATA SCIENCES AND BUSINESS ANALYTICS

CENTRALESUPÉLEC

ASSIGNMENT 2 –

MULTI-CLASS EMAIL CLASSIFICATION CHALLENGE

KAGGLE COMPETITION

Ruixu Chen, Camille Morand-Duval, Antoine Ohleyer, Ismail Zizi

Team ∇

20/11/2019

Introduction

Among the numerous emails received every day, only a few are relevant. Searching manually for emails which include important information is highly inefficient and can lead to missing crucial documents and findings. A multi-class email classifier would resolve such issues. The tool designed for this purpose uses the metadata of a given email to classify it into one of four categories: update, social, forum and promotional.

The machine learning model developed uses a training set provided by Kaggle. Multiple features have been extracted from the emails into a .csv file, namely the date, organisation of the sender, top level domain of the organisation, number of emails to which duplicates have been sent (cced), whether the email is a duplicate (bcced), the body type of the email, the number of images and the number of urls included, whether salutations are used, the number of characters in the subject and the body and finally the label associated with the email.

This report outlines the pipeline followed to reach a working model. It details the feature engineering involved in the design as well as the choice of the learning algorithm and the cross-validation techniques applied to evaluate the performance of the model for this multi-class classification task.

1. Feature engineering

1.1. Pre-processing

The data is pre-processed into an appropriate format. This transformation is applied to both the training and test sets. The processing includes cleaning the dates and times, converting the mail type classes to all lower case and splitting mail_type into two classes (for example: text / html becomes 'text' and 'html'). Further analysis shows that some inputs are incorrect, the initial features organisation and top-level domain values and mail type include respectively 5% NaN 0.2%. This issue is resolved by filling-in missing data i.e. adding 'org_missing' if the organisation name is missing. The training and test data are concatenated for feature engineering.

1.2. Exploratory data analysis

Choosing relevant features relies on understanding the data and using the trends found in an efficient way. To gain some knowledge of the data, some exploratory analyses were led. The distribution of the labels in the training set shows that about 40% of the emails are classified as updates and that about three quarters of the emails are classified as multipart / alternative. Furthermore, analysis of the categorical data validates the elimination of bcced as a critical feature as it is constantly equal to 0 in both the training and test sets.

Such insights are used to evaluate the potential differences between the training and test sets, leading to further feature selection. After the minmax normalisation of the training and test continuous data, it can be shown that the distribution of the number of images, characters in the subjects and body are quite similar whereas the number of urls and css differ. However, the differences are not considered high enough to eliminate specific features.

1.3. Feature selection

The impact of the main features is evaluated using the crosstab function. Organisation is a critical feature for email classification. There are 649 distinct categories in the training data, 474 in the testing data and 409 present in

both. Additional features for organisations are therefore necessary. However, creating the same feature for highly differentiative organisation (example: 123 will always be promotional while LinkedIn may be promotional or social) is not ideal. A first approach to dealing with organisation features was to include a threshold of 5 i.e. create a new category for value appearing at least five times for a unique label. A category 'other_feature' is created for values appearing in only one of the data sets.

A second approach considers the misclassifications observed in both sets. For example, the segment 'mail.paypal.com' classifies 'mail' as the organisation and 'paypal.com' as the top-level domain. To resolve this frequent issue, the organisation and top-level domain recorded are merged into one and then re-divided into columns (MultiLabelBinarizer). Taking the previous example, 'mail.paypal.com' is separated into 'mail', 'paypal' and '.com'. Each new column is filled with 0 and 1. The downside to this approach is that the matrix becomes very sparse. An additional feature is created summing the 1 and 0 recorded for each email, in this example, 3.

Furthermore, to get a sense of the importance of each feature, a score is attributed to each one via a Random Forest algorithm. Features with importance scores lower than 0.01 such as designation, salutations, date, mail_type_1_freq and mail_type_1 were dropped. Duplicate features i.e. features with the exact same distribution in the training and test sets are removed as well as columns only filled with zeros in either of the training or test sets.

Numerical encoding is applied to the mail types and time zones and the frequency is calculated for each one. The data is re-divided into the original training / test sets.

Additional experiments were led on the date and time features. The US holiday calendar was imported for comparison and analysis of the correlations if any with the labels of the emails. However, this feature was not very correlated with the labels and therefore discarded from the final model.

2. Model tuning and comparison

Three main models were tested and evaluated for the task of classifying multi-class emails correctly. The baseline model used in the skeleton.py code is a k-nearest neighbour algorithm. It uses a single feature and yields a very low F1-score of 0.28052. Gradient Boosting and Majority voters were also explored but to a smaller extent given their initial performance.

2.1. Random Forest

The first model tested is a Random Forest algorithm using 300 estimators with only basic feature engineering and scored 0.94614 on Kaggle. The relatively high score of this initial model led to further testing of the Random Forest Algorithm. The model is tested after the Principal Component Analysis dimensionality reduction on the full data set. With 300 estimators, the F1-score averaged 0.95113 using 5-folds cross-validation (CV).

After the implementation of feature engineering, the number of estimators was reduced from 300 to 250 and the maximum number of features from 250 and 200 before dropping duplicates to 200 and 175 after, both iteratively. The model performance was evaluated using 5 folds CV of the model. The F1-score recorded averaged at 0.95509 for the final Random Forest model.

To prevent overfitting, a grid search cross-validation of three inter-cross validations using all processors was applied to the random forest parameters.

2.2. Support Vector Machines (SVM)

The second model uses SVM to predict the classes of the emails. The algorithm was tested with additional pre-processing (the NaN values are replaced with -1 and the data is standardize using a StandardScaler), as well as the implementation of some feature engineering (one hot encoding of categorical data, addition of organisation and top-level domain frequencies and removal of so called non-important features). The tested SVM algorithm uses a penalty parameter of 100 and a linear kernel. With the same 5-folds CV and F1-scoring as the Random Forest algorithm, the average performance was lower, only 0.93471. Since the data is multi-categorical, the result obtained from this algorithm is expected.

2.3. Neural Networks

Neural networks were also explored. The data is processed as described in section 1 and 2.2. Additionally, the logarithm of the characters in the subject and body of the emails and of the urls is taken. The data is standardised. The Neural Network tested is designed with outputs of each inner layer set to 150, 450, 50 and 200, and the activation to softmax. The score yielded was very low, 0.4167. A second set of parameters was tested with three layers of 100 outputs, initialised with the total number of features, outputting the four categories and with a sigmoid activation. The final accuracy is 0.9427 so not as accurate as the Random Forest algorithm.

2.4. Gradient boosting classifiers

Moreover, gradient boosting classifiers were tested. It is designed with 300 estimators and the same processing and feature engineering as in section 2.2 and 2.3. Its average F1-score using a 5-folds cross validation is 0.94410. Not only is the F1-score lower but also this type of algorithm does not seem well suited for the task at hand. Since each classifier is trained based on the knowledge of previous performance, the algorithm is more sensitive to noise and parameter tuning.

2.5. Majority voters

The final model tested is a majority voters algorithm including the results from random forest, gradient boosting and Adaboost classifiers, each initialised at 100 estimators. The hard and soft votes are employed with customised weights for each classifier. The relatively low F1-score after the 5-folds cross validation suggests it should be better optimised. The added complexity of this model does not result in rise of the F1-score. Hence, the model was not selected as the final version of the mail classifier.

Conclusion

The final model designed to classify emails is a Random Forest Algorithm using 175 features. It yields an F1-score of 0.95585 on Kaggle and uses specific features, namely ccs, images, urls, characters in subject, character in body, year, month, day, hour, minute, second, weekday, time zone, mail_type_2, multiple company names, mail_type_2 frequency, time zone frequency and the sum of binary organisation and top-level domain features. The model was tested using grid search cross validation and 5-folds cross validation to avoid over-fitting.