2EL1730: MACHINE LEARNING

CENTRALESUPÉLEC

Kaggle Challenge

Instructors: Fragkiskos Malliaros and Maria Vakalopoulou

Competition Assistant: Sagar Verma (sagar.verma@centralesupelec.fr)

Team name : **3-NearestNoobs**

Students: **Akiyo Worou, Alexandre Muller, Mattéo Faelli**

Section 1: Feature engineering

1)

First, we split the features into two categories: numerical features and categorical features. We then focused on the categorical features which are date, org, tld and mail_type.

Intuitively, a date can be important to predict the type of a mail. For example, during the sales there are more promotional emails, the email in the middle of the night may probably be a spam, etc. Thus, we deleted the date column and added new columns more useful such as day ('Mon','Tue','Wed',…), month('Jan','Feb',…), time(the normalize time in GMT) , year, number_date (1,2,…).

We did not have enough information about tld and org but we can only say that some organizations and some domain names are more frequent than others.

The mail_type was on of the most interesting features. By using the unique method of pandas we realized that instead of using mail_type we can create new features with Boolean values(1 or 0). The features are: multipart, alternative, text, html, mixed, plain, related, signed, report, calendar, idm.

Finally, we only had tld and org as categorical features. All the others were numerical features.

We also thought about creating some features based on different ratios. For instance, the ratio between char_in_subject and char_in_body, urls multiplied by a mean length of urls and char_in_body.

For selecting the most important features we used a variance threshold to eliminate the features with low variance (0.01 for instance).

2)

We have tried to train different models for different variance thresholds. With simple models without tuning we obtained the following tables

	Logistic Regression				
Variance threshold	0	0.05	0.10	0.15	0.01
Train log loss	1.6286	1.5793	1.6103	1.3512	1.5512
Test log loss	1.6325	1.5707	1.6187	1.3810	1.5621

	XGB Classifier				
Variance threshold	0	0.05	0.10	0.15	0.01
Train log loss	1.0990	1.1022	1.1994	1.2042	1.0973
Test log loss	1.1769	1.1747	1.2709	1.2963	1.1755

We then chose a threshold of 0.01, which seemed a good trade-off.

	Decision Tree Classifier				
Variance threshold	0	0.05	0.10	0.15	0.01
Train log loss	0.9023	0.9027	0.9009	0.8987	0.9019
Test log loss	4.5256	4.5530	6.1415	6.2555	4.6842

	Random Forest Classifier				
Variance threshold	0	0.05	0.10	0.15	0.01
Train log loss	0.9872	0.9886	0.9936	0.9835	0.9870
Test log loss	1.2133	1.1954	1.2805	1.3076	1.2150

Section 2: Model tuning and comparison

1)

The comparison between the different model without tuning was already done before. After that we decided to pursue with only XGB Classifier, Random Forest Classifier and Decision Tree Classifier.

2)

We mainly focused on ensemble models. For all the models we used cross validation with RandomizedSearchCV and gridSearch to tune the hyperparameters.

3)

The best model was a model with a XGB classifier.

base_estimator=XGBClassifier(max_depth= 9, min_child_weight= 1 , gamma= 0.4, colsample_bytree= 0.7, subsample= 1, reg_alpha= 0.0001, learning_rate= 0.2, n_estimators= 350)

The cross-validated model score is 0.99722 and the test set score on Kaggle is 0.05635

4)

We tried to implement a neural network with a multi class classifier. But, in order to do so, we had to remove “tld” and “org” features because we could not convert strings into float.
We failed to train the neural network because all predictions had approximately the same probabilty for every categorial output, whatever the features we gave as entries of the neural network.