

Spam Filtering Techniques for Short Message Service

Adrien Besson and Dimitris Perdios

Signal Processing Laboratory (LTS5)
École Polytechnique Fédérale de Lausanne (EPFL)

Adaptation and Learning Project
Lausanne, Switzerland
June 4, 2018

Introduction

- Context

- Data Exploration

Feature Extraction

- The Bag-of-Words Representation

- The Term Frequency-Inverse Document Frequency Weighting

Comparison of Several Classifiers

- Considered Classifiers

- Experimental Settings

- Results

On Sensitivity and Specificity

- Increasing the Sensitivity

- Results

Online Learning Strategies

- Motivations and Strategies

- Experiments

Conclusion

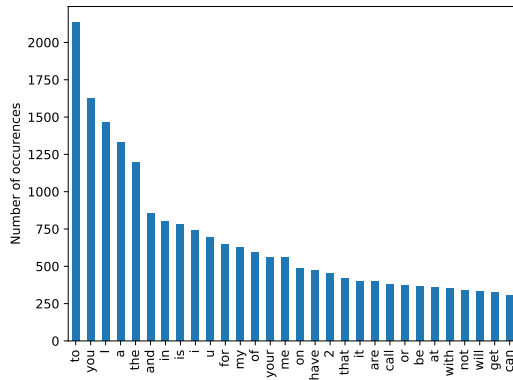
- ▶ Short message service (SMS) very popular communication service ➔ 15 M SMS sent per minute in the world in 2017
- ▶ Spam is unwanted message sent in bulk to a group of recipients
 - ▶ Intrusive
 - ▶ Waste of money, time and network bandwidth
- ▶ SMS filtering still at relatively early stage
 - ▶ No publicly available dataset of sufficient size
 - ▶ Current methods inherit from Email spam filtering techniques
- ▶ Supporting code: <https://github.com/dperdios/sms-spam-filtering>

SMS spam collection dataset

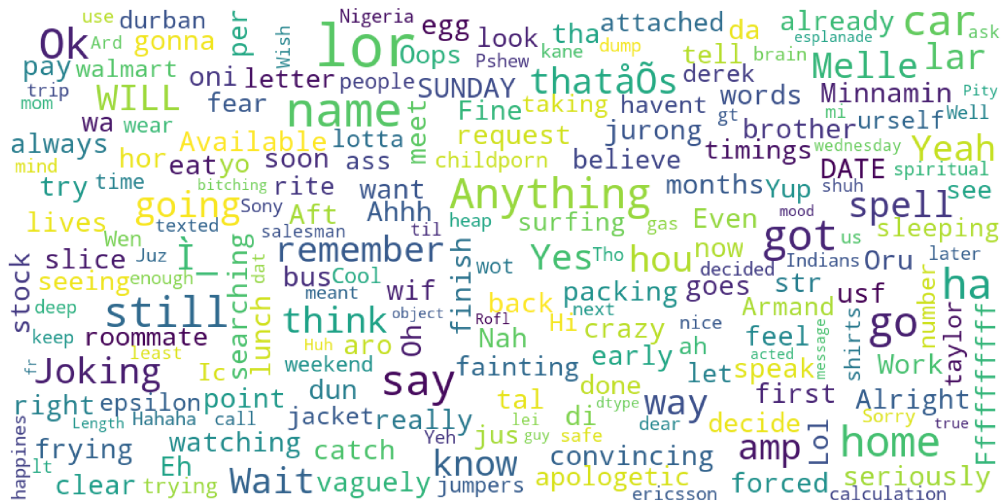
- ▶ 5572 tagged SMS extracted from Grumbletext Web site / Caroline Tag's PhD thesis / SMS Spam Corpus v. 0.1 Big
- ▶ Dataset composed of 4825 **ham** (86 %) and 747 **spam** (14 %)

Most represented words in the dataset

- ▶ Stop-words: Most commonly used english words
- ▶ Uninformative words that have to be removed



Data Exploration – Ham



Data Exploration – Spam



Feature Extraction

The Bag-of-Words Representation

Motivation

A document considered as a sequence of symbols cannot be used as input of a machine learning algorithm → need a **representation**

The Bag-Of-Words representation

- ▶ **Document** composed of D words and **vocabulary** composed of M words:

$$\mathbf{x} = \sum_{i=1}^D \mathbf{x}_{d_i},$$

$\mathbf{x}_{d_i} \in \mathbb{R}^M$ one-hot vector with 1 at the entry of d_i in the vocabulary

- ▶ Given a **collection** of N documents → $\mathbf{X} \in \mathbb{R}^{N \times M}$

SMS dataset

- ▶ A SMS is a document
- ▶ The vocabulary is extracted from the training set

Feature Extraction

The Term Frequency-Inverse Document Frequency Weighting

Motivation

- ▶ Preferable to work with frequencies than occurrences
- ▶ Need for normalized features

Tf-idf weighting

- ▶ Term-frequency of the document n of size D_n in the collection

$$tf_n = \frac{1}{D_n}$$

- ▶ Inverse-document-frequency of the word d_i in the collection

$$idf_{d_i} = \log \frac{N}{df_{d_i}}$$

$$df_{d_i} = \# \text{ documents containing } d_i$$

- ▶ Tf-idf weighting of the word d_i in the document n

$$T(n, d_i) = tf_n \times idf_{d_i}$$

Comparison of Several Classifiers

Considered Classifiers

Type	Method	Main parameters
k-nearest neighbors	kNN	Number of neighbors
Multinomial Naive Bayes	MNB	Additive smoothing parameter
Logistic Regression	LR	–
	LR-ℓ_2	Regularization parameter
	LR-ℓ_1	
	SVM-L	
Support Vector Machine	SVM-R	Reg. param. on slack variable
	SVM-S	
	DT	Minimum number of samples at leaf
Trees	RF	Number of trees in the forest
	AB	Number of estimators
Neural Network	MLP	Hidden-layer sizes

Comparison of Several Classifiers

Experimental Settings

General remarks:

- ▶ No dimensionality reduction of the data
- ▶ Even though data imbalanced we use a threshold of 0.5 for our Bayes plug-in estimator
- ▶ Dataset divided into a training set (80 %) and test set (20 %)
- ▶ Hyper-parameter grid search using 10-fold cross-validation on training set

Metrics:

- ▶ Misclassification error (ME): $ME = \frac{1}{N_t} \sum_{i=1}^{N_t} \mathbb{I}(c(\mathbf{h}_i) \neq \gamma(i))$
- ▶ Sensitivity (SE): Probability of predicting **spam** given that it is **spam**

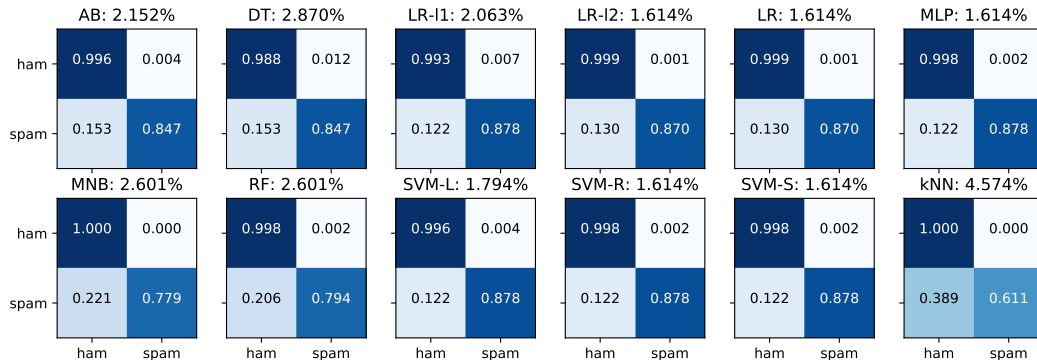
$$SE = \frac{1}{\#_{\text{spam}}} \sum_{i=1}^{N_t} \mathbb{I}(c(\mathbf{h}_i) = \gamma(i)) \times \mathbb{I}(\gamma(i) = \text{spam})$$

- ▶ Specificity (SP): Probability of predicting **ham** given that it is **ham**

$$SP = \frac{1}{\#_{\text{ham}}} \sum_{i=1}^{N_t} \mathbb{I}(c(\mathbf{h}_i) = \gamma(i)) \times \mathbb{I}(\gamma(i) = \text{ham})$$

Comparison of Several Classifiers

Results



On Sensitivity and Specificity

Increasing the Sensitivity

Motivations

- ▶ Optimization of a symmetric loss on an imbalanced dataset → good predictive performance on the majority class and poor result on the other class
- ▶ One may want to increase sensitivity

Methods

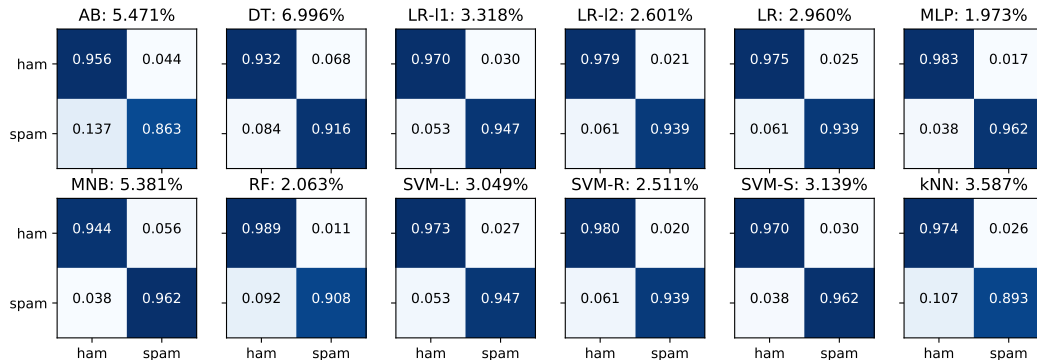
- ▶ Decreasing the threshold of the Bayes classifier
- ▶ **Resampling:**
 - ▶ Downsampling of the majority class
 - ▶ Upsampling of the minority class

Proposed experiments

1. Random downsampling of the **ham** class to have a balanced dataset
2. Random upsampling with replacement of the **spam** class to have a balanced dataset

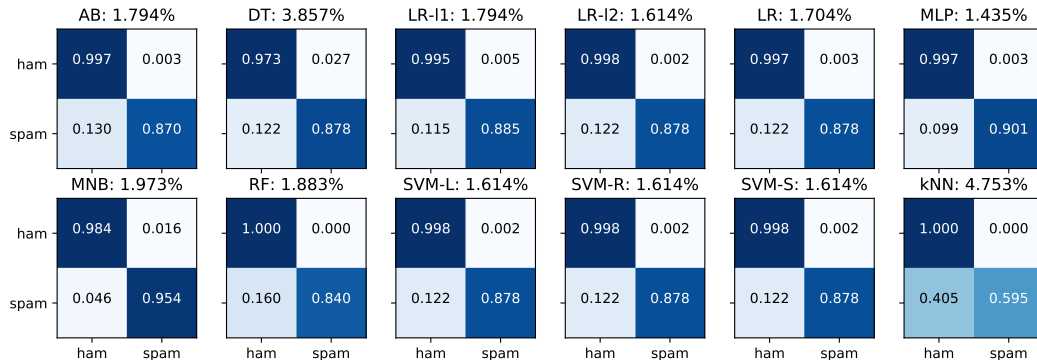
On Sensitivity and Specificity

Results – Downsampling the Majority Class



On Sensitivity and Specificity

Results – Upsampling the Minority Class



Motivations

- ▶ Spam content is obviously changing with time: spammers try to fool spam filtering algorithms
- ▶ Useful to design algorithm able to adapt to new content labeled by users
- ▶ Main difficulty in designing an online strategy: the vocabulary size

Naive strategy

- ▶ Increasing vocabulary size using all the words of the training set
- ▶ Not viable in longer-term due to memory and computational issues

Windowed-time strategy

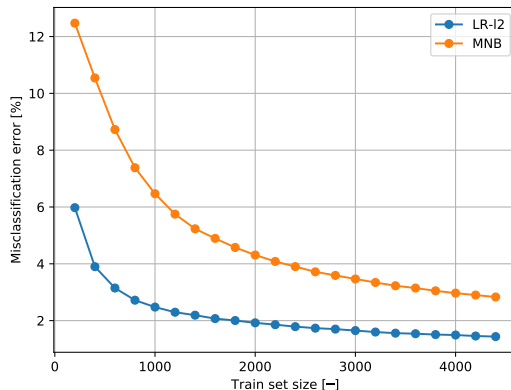
- ▶ Fixed vocabulary size using only the most frequent words of the training set
- ▶ Training set windowed in time
 - ▶ Should be quite efficient
 - ▶ **But** requires to be fitted quite often

Considerations

- ▶ We do not have access to time information
- ▶ Not possible to analyze the evolution of the vocabulary
- ▶ Relatively small dataset

Online scenario


- ▶ Naive strategy (increasing training set)
- ▶ Labeled messages received by batches of 200
- ▶ Test set fixed at the beginning
- ▶ Fast classifiers (LR- ℓ_2 , MNB)





- ▶ Several SMS spam filtering methods have been studied from the feature extraction step to the classification
- ▶ State-of-the-art classifiers work well with a misclassification error of less than 5 %
- ▶ Resampling methods can be used to counter class imbalance but impact the classification error ➔ trade-off required
- ▶ Online strategies may have to be adapted to deal with the increasing vocabulary size
- ▶ More advanced feature extraction methods (e.g. word embeddings) and classification methods (e.g. deep neural networks)


THANK YOU FOR YOUR ATTENTION!

Adrien Besson and Dimitris Perdios

 <https://github.com/dperdios/sms-spam-filtering>

 Signal Processing Laboratory (LTS5)

 <https://lts5www.epfl.ch>

 École Polytechnique Fédérale de Lausanne