# Automatic documents classification
# SortDC

Godefroy de Compreignac Ronan Letellier

July 5, 2011

## Contents

# Introduction

Nowadays, the always increasing amount of information available on the Internet makes it impossible to let it be managed exclusively by humans. One important remaining challenge in computer sciences is making it possible to use computers to accomplish tasks that would normally require free will. Indeed, some tasks can require a significant amount of time without producing any added value. Automating some of these process would allow one to concentrate on more complicated issues, thus gaining in productivity.

Our work during this semester (February 2011 - June 2011) consisted in finding a mathematically correct algorithm in order to automatically sort « human-readable » documents in different categories, and programming an application using this algorithm to actually accomplish the task.

# 1 Needs and goals

## 1.1 What did we want to achieve

Our intention was to conceive an Open Source product that would be easy to configure and use, as we could not find an existing one. To reach these requirements, we had to be able to provide a REST API that would allow anyone to try and use our software features. To make it really flexible, we also wanted it to work on different DBMS.

## 1.2 Why do we think this is useful

As said previously, we have not been able to find any Open Source software matching all our expectations.

**description d'un ou deux outils avec leurs défauts**

Apache Mahout

While the internet content keeps growing, we need powerful devices to keep some order in all human oriented items. Automatically understanding what a text document is about allows search engines to determine wether a website suits one's key words or not, our mailboxes to send junk mail directly in the spam folder, and so on. However, such automated softwares are not available to anyone, and we still have to order our own web contents by hand.

# 2 Studied technologies

Sorting documents is not a one step process. We decided to go for texts stemming and tokenization. Therefore, we tried to find existing solutions that would help us to do so.

## 2.1 Existing solutions

Stemming seemed to be the hardest part, for we had to take in consideration that texts may be in different languages. Finding a word's root depends on text language and for

we cannot speak whole of them, using a pre-existing solution appeared to be the best solution to us. Various open source applications use stemming, but a lot of them use it for a specific purpose in a specific language (for instance, Sphinx full-text-search). Snowball Stemmer, on the other hand, has specialized in stemming text documents only. Therefore, the available application supports more than 15 latin languages.

### 2.2 Our selection

## 3 Bayes classifier

### 3.1 Considerations

### 3.2 Algorithm

## 4 The application

### 4.1 Quick overview

### 4.2 Features

### 4.3 Some tests

## Conclusion

# Bibliography