**DOCUMENT CLASSIFICATION SYSTEM**

**User Manual**

**2014/15 Semester A**

**CS3343 LA1 – Acumen**

**Delivery Date:** 5th Dec 2014

**USER'S MANUAL**

**TABLE OF CONTENTS**

1.0 GENERAL INFORMATION

1.1 System Overview

1.2 Major functions performed by the system

2.0 SYSTEM SUMMARY

2.1 Configuration Folder Structure

2.1.1 Dictionary

2.1.2 map

2.1.3 matrix

2.1.4 vector

3.0 GETTING STARTED

3.1 Command Supported

3.2 Starting Program

# 1. GENERAL INFORMATION

## 1.1 System Overview

In this information age, increasing amount of information is conveyed in digital form. Our system aims to act as a smart “librarian” who can tell the catalogue of the document belongs to within the binary codes. The process of classifying documents to catalogues is no longer painstaking and time-consuming with help of this system. User can immediately get the catalogue of the document by simply input the location of the document, getting rid of the need to read the document from the beginning. Designed to be “smart”, the system will “learn” after each task it performs by expanding its own word library, making itself more and more powerful.

## 1.2 Major functions performed by the system

Functions:

1. The project can accept a user provided training data set and use it to train the matrix for classification. By default, the system will use training data set provided by us.

2. After building the matrix, we can use it to classify the category that user input file belongs to.

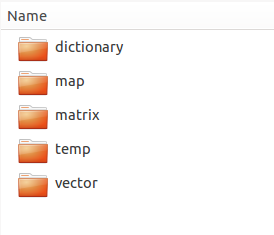
User Access Mode: command line.

Pre-requirement: jdk or jre installed.

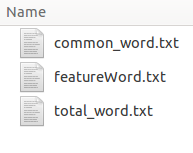
System name: Document Classification System

# **2. SYSTEM SUMMARY**

**2.1** **Configuration Folder Structure**



**2.1.1 dictionary**

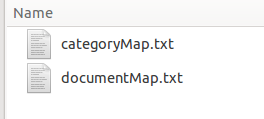


1. common\_word.txt contains words that are not useful for classification process, such as “a”, “the”. User may add words they do not want to include in the classification process to this file.

2. featureWord.txt contains words our program selected for classification.

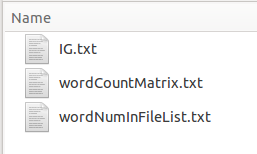
3. total\_word.txt contains all the words appeared in the training documents.

**2.1.2 map**



These two files contain the mapping relationship between file name and indexes.

**2.1.3 matrix**

****

1. IG.txt shows the information gain of each word which use the wordCountMatrix.txt to calculate.

2. wordCountMatrix.txt contains the word appearances in each file.

**2.1.4 vector**

vectorMatrix.txt uses all the feature words to calculate vector for training files. And using this vector matrix, we can calculate the similarity between new file and the existing files. Thus, we are able to classify the category of the input file.

# **3. GETTING STARTED**

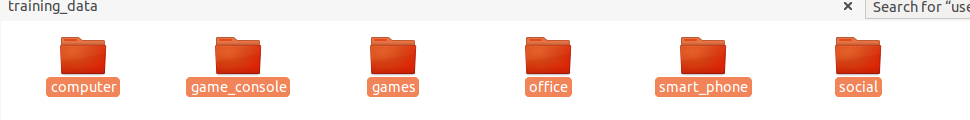
**3.1 Command Supported**

This section provides a general walk through of the project from initiation through exit. Basically, two commands are supported to run this jar file.

**1. java -jar Classifier.jar <training data root path> <file to be classified>**

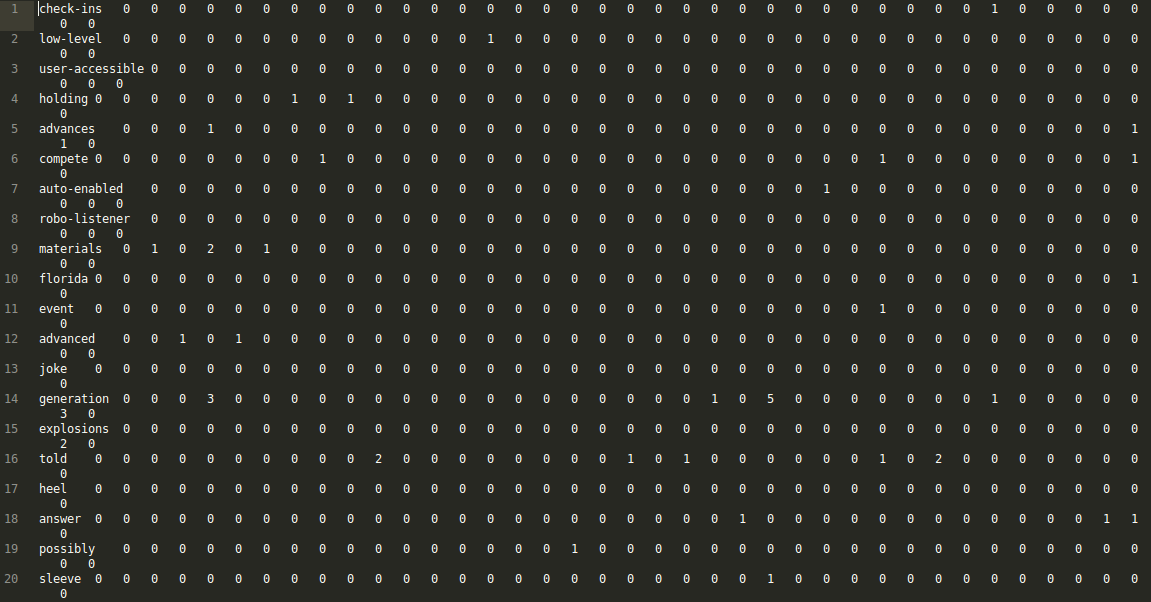
<training data root path>: the folder that contains different categories.

For example: **./training\_data** (which contains 6 categories)

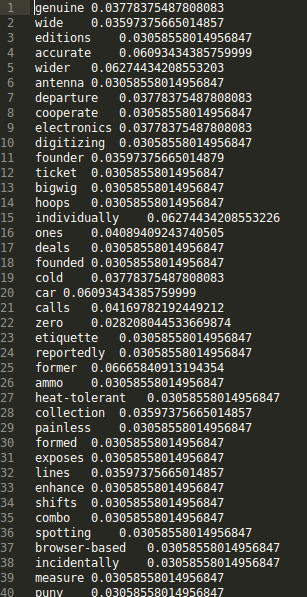
****

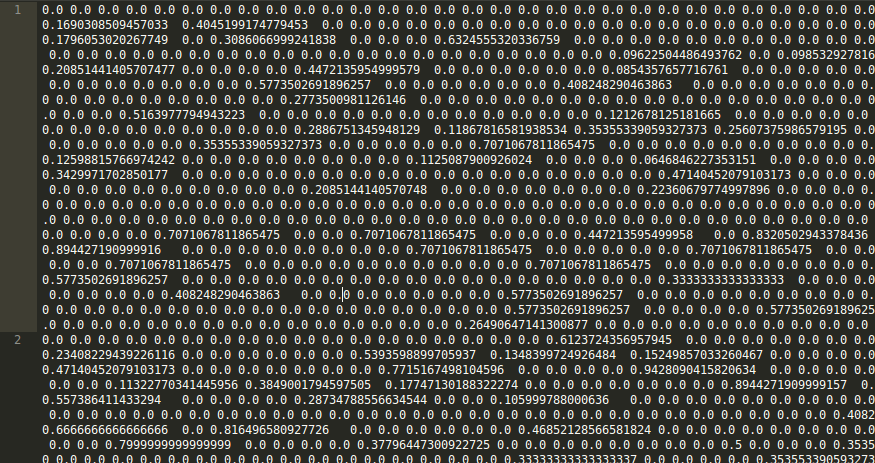
<file to be classified>:the file needs to be classified.

The program will first generate the word count matrix and then calculate the information gain for each word.

****

Information Gain:



Then, the program will work out the vector matrix for comparison.

**2. java -jar Classifier.jar <file to be classified>**

If user do not define the training data set, the program will use the default training set for classification.

**3.2 Starting Program**

In console, we can run the program through command easily.

In this example, we have a training data set – training\_data and a file test.txt for testing purpose.

Case 1: NO argument provided



Case 2: Only one file provided

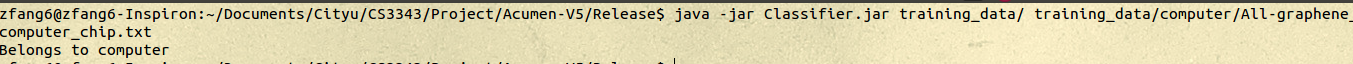
java -jar Classifier.jar test.txt



The program correctly classify the document as computer category.

Case 3: Training data set and file provided

java -jar Classifier.jar training\_data/ test.txt



The program correctly classify the document as computer category.