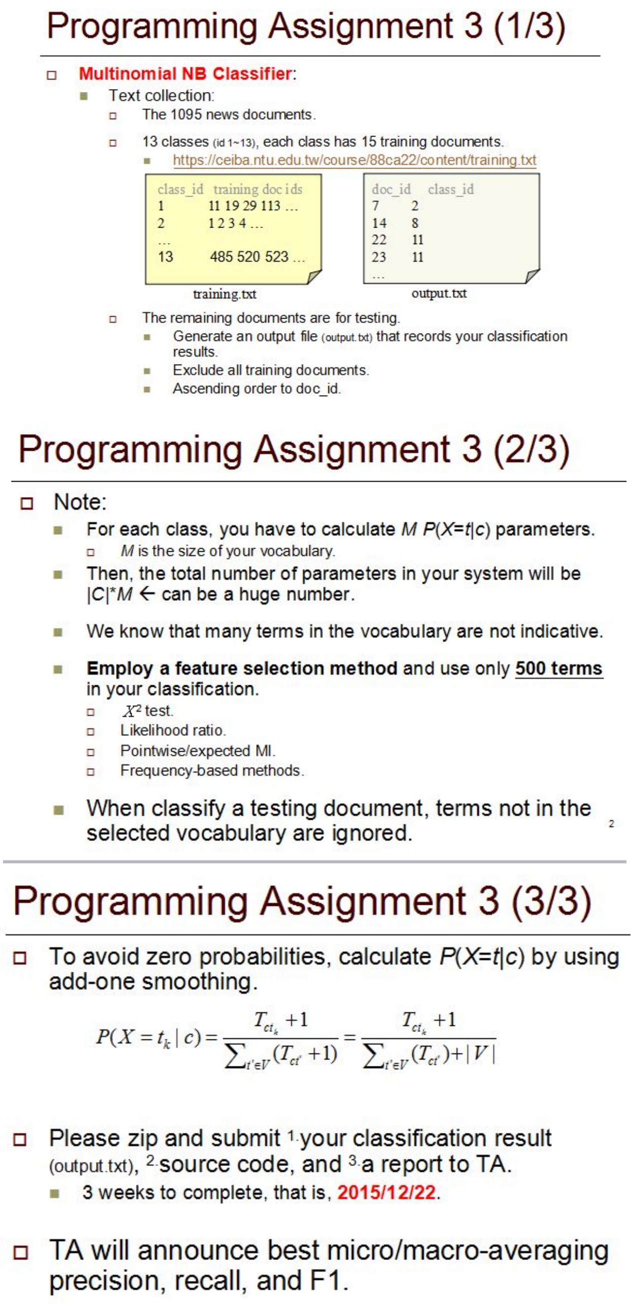
==

Name：　李桂欽　ID：R04725050　Department：資訊管理 碩一 Homework：　3

==

Programming Assignment 3:



Programming Assignment 3 (1/3)

- **Multinomial NB Classifier**:
  - Text collection:
    - The 1095 news documents.
    - 13 classes (id 1~13), each class has 15 training documents.
      - https://ceiba.ntu.edu.tw/course/88ca22/content/training.txt

training.txt

class_id	training doc ids
1	11 19 29 113 …
2	1 2 3 4 …
…	
13	485 520 523 …

output.txt

doc_id	class_id
7	2
14	8
22	11
23	11
…	

- The remaining documents are for testing.
  - Generate an output file (output.txt) that records your classification results.
  - Exclude all training documents.
  - Ascending order to doc_id.

Programming Assignment 3 (2/3)

- Note:
  - For each class, you have to calculate *M* *P*(*X*=*t*|*c*) parameters.
    - *M* is the size of your vocabulary.
  - Then, the total number of parameters in your system will be |*C*|**M* ← can be a huge number.
  - We know that many terms in the vocabulary are not indicative.
  - **Employ a feature selection method** and use only **500 terms** in your classification.
    - X^2 test.
    - Likelihood ratio.
    - Pointwise/expected MI.
    - Frequency-based methods.
  - When classify a testing document, terms not in the selected vocabulary are ignored.

2

Programming Assignment 3 (3/3)

- To avoid zero probabilities, calculate *P*(*X*=*t*|*c*) by using add-one smoothing.

$$P(X=t_k\,|\,c)=\frac{T_{ct_k}+1}{\sum_{t'\in V}(T_{ct'}+1)}=\frac{T_{ct_k}+1}{\sum_{t'\in V}(T_{ct'})+|V|}$$

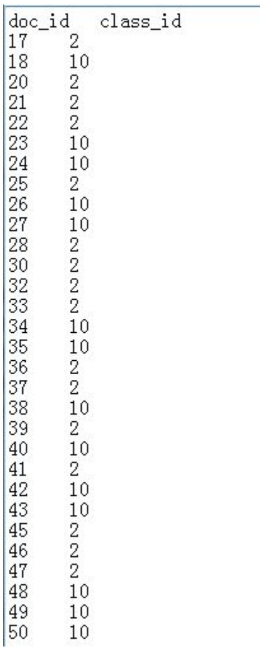
- Please zip and submit 1.your classification result (output.txt), 2.source code, and 3.a report to TA.
  - 3 weeks to complete, that is, 2015/12/22.
- TA will announce best micro/macro-averaging precision, recall, and F1.

My program result:

Step1：部署 Hw3

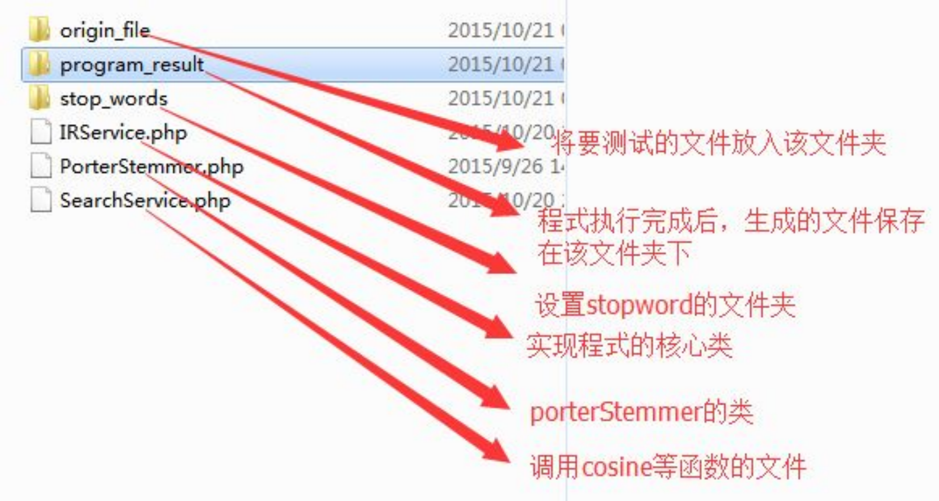
Step2：在流覽器輸入 http://www.mytest.com/SearchService.php

生成的 Result 文檔，詳見 program_result 檔夾，大致如下：



doc_id	class_id
17	2
18	10
20	2
21	2
22	2
23	10
24	10
25	2
26	10
27	10
28	2
30	2
32	2
33	2
34	10
35	10
36	2
37	2
38	10
39	2
40	10
41	2
42	10
43	10
45	2
46	2
47	2
48	10
49	10
50	10

My program architecture:



origin_file 2015/10/21
program_result 2015/10/21
stop_words 2015/10/21
IRService.php
PorterStemmer.php 2015/9/26
SearchService.php

将要测试的文件放入该文件夹

程式执行完成后，生成的文件保存在该文件夹下

设置stopword的文件夹

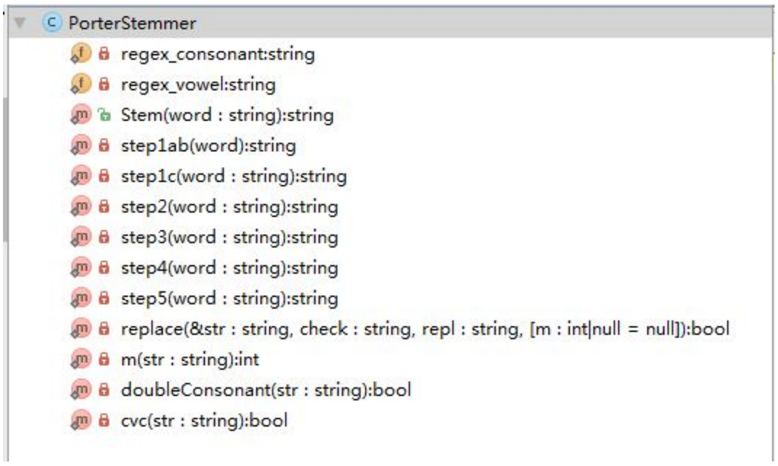
实现程式的核心类

porterStemmer的类

调用cosine等函数的文件

My program main class:

PorterStemmer Class Structure：



PorterStemmer
- regex_consonant:string
- regex_vowel:string
- Stem(word : string):string
- step1ab(word):string
- step1c(word : string):string
- step2(word : string):string
- step3(word : string):string
- step4(word : string):string
- step5(word : string):string
- replace(&str : string, check : string, repl : string, [m : int|null = null]):bool
- m(str : string):int
- doubleConsonant(str : string):bool
- cvc(str : string):bool

演算法實現過程：

第一步，處理複數，以及 ed 和 ing 結束的單詞。

第二步，如果單詞中包含母音，並且以 y 結尾，將 y 改為 i。

第三步，將雙尾碼的單詞映射為單尾碼。

第四步，處理-ic-，-full，-ness 等等尾碼。

第五步，在<c>vcvc<v>情形下，去除-ant，-ence 等尾碼。

第六步，也就是最後一步，在 m()>1 的情況下，移除末尾的“e”。

演算法使用說明：

傳入的單詞必須是小寫

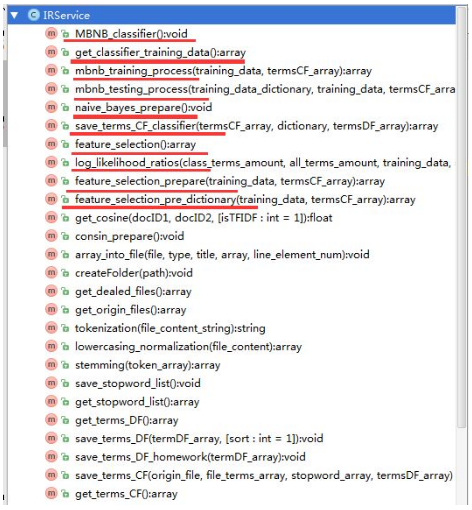
參考學習網站：

http://tartarus.org/~martin/PorterStemmer/

http://snowball.tartarus.org/algorithms/english/stemmer.html

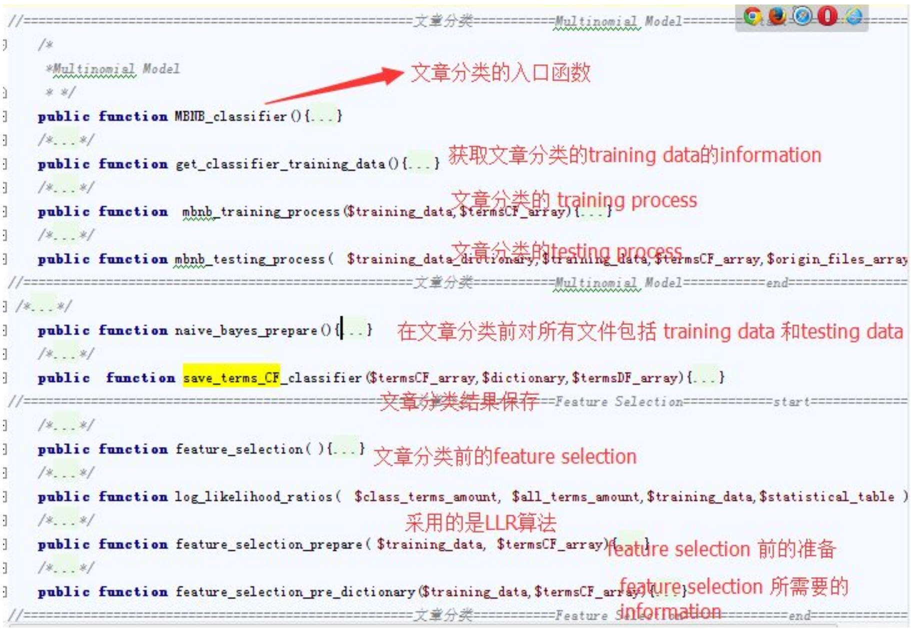
http://blog.csdn.net/noobzc1/article/details/8902881

IRService Class Structure：



```
IRService
    MBNB_classifier():void
    get_classifier_training_data():array
    mbnb_training_process(training_data, termsCF_array):array
    mbnb_testing_process(training_data_dictionary, training_data, termsCF_arra
    naive_bayes_prepare():void
    save_terms_CF_classifier(termsCF_array, dictionary, termsDF_array):array
    feature_selection():array
    log_likelihood_ratios(class_terms_amount, all_terms_amount, training_data,
    feature_selection_prepare(training_data, termsCF_array):array
    feature_selection_pre_dictionary(training_data, termsCF_array):array
    get_cosine(docID1, docID2, [isTFIDF : int = 1]):float
    consin_prepare():void
    array_into_file(file, type, title, array, line_element_num):void
    createFolder(path):void
    get_dealed_files():array
    get_origin_files():array
    tokenization(file_content_string):string
    lowercasing_normalization(file_content):array
    stemming(token_array):array
    save_stopword_list():void
    get_stopword_list():array
    get_terms_DF():array
    save_terms_DF(termDF_array, [sort : int = 1]):void
    save_terms_DF_homework(termDF_array):void
    save_terms_CF(origin_file, file_terms_array, stopword_array, termsDF_array)
    get_terms_CF():array
```

類的主要函數：



```
//================================================文章分类==========Multinomial Model==========================
    /*
     *Multinomial Model
     * */
    public function MBNB_classifier(){...}
    /*...*/
    public function get_classifier_training_data(){...}
    /*...*/
    public function  mbnb_training_process($training_data,$termsCF_array){...}
    /*...*/
    public function mbnb_testing_process(  $training_data_dictionary,$training_data,$termsCF_array,$origin_files_array
//================================================文章分类==========Multinomial Model==========end==============
 /*...*/
    public function naive_bayes_prepare(){...}
    /*...*/
    public  function save_terms_CF_classifier($termsCF_array,$dictionary,$termsDF_array){...}
//================================================文章分类==========Feature Selection==========start============
    /*...*/
    public function feature_selection( ){...}
    /*...*/
    public function log_likelihood_ratios(  $class_terms_amount, $all_terms_amount,$training_data,$statistical_table )
    /*...*/
    public function feature_selection_prepare( $training_data, $termsCF_array){...}
    /*...*/
    public function feature_selection_pre_dictionary($training_data,$termsCF_array){...}
//================================================文章分类==========Feature Selection==========end============
```

文章分类的入口函数

获取文章分类的training data的information

文章分类的 training process

文章分类的testing process

在文章分类前对所有文件包括 training data 和testing data

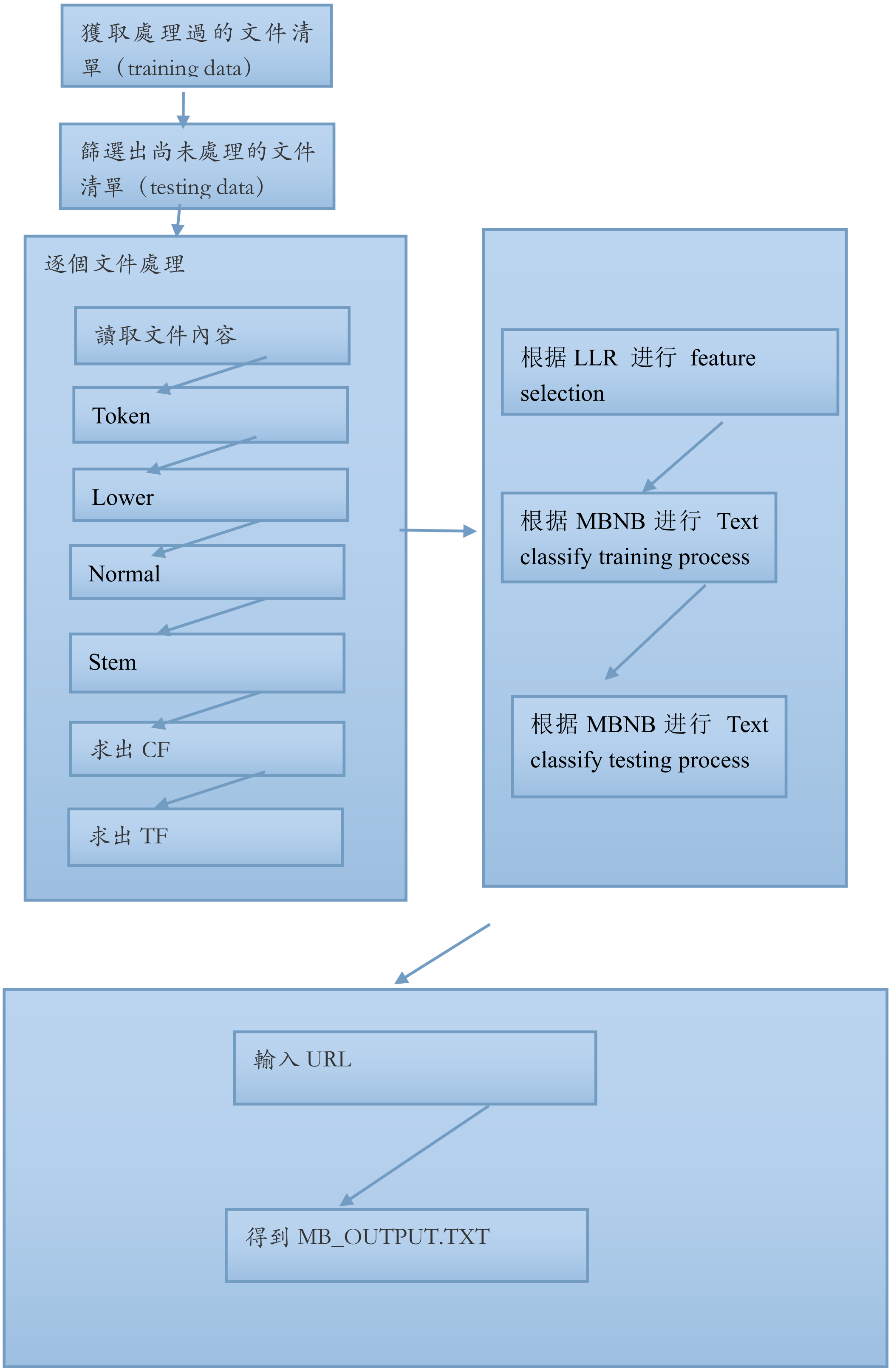
文章分类结果保存

文章分类前的feature selection

采用的是LLR算法

feature selection 前的准备

feature selection 所需要的 information


獲取處理過的文件清單（training data）
篩選出尚未處理的文件清單（testing data）
逐個文件處理
讀取文件內容
Token
Lower
Normal
Stem
求出 CF
求出 TF
根据 LLR 进行 feature selection
根据 MBNB 进行 Text classify training process
根据 MBNB 进行 Text classify testing process
輸入 URL
得到 MB_OUTPUT.TXT