TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN – ĐHQG TP.HCM

Khoa: KHOA HỌC MÁY TÍNH

Báo cáo đồ án

MÁY HỌC TRONG XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Phân loại văn bản – text classification

Sinh Viên:

- Trịnh Ngọc Hiếu – 16520418

Mục lục:

I. Tổng quan về đề tài

1. Giới thiệu đề tài

- Phân Loại Văn bản là một vấn đề quan trọng trong xử lý ngôn ngữ tự nhiên
- Với các nhiệm vụ gán các tài liệu văn bản vào nhóm các chủ đề cho trước
- Rất thường gặp trong thực tế, điển hình như:

    - Một nhà văn muốn viết tiểu thuyết về câu chuyện lấy cảm hứng từ một giai đoạn, bối cảnh nào đó đã xảy ra từ xa xưa nhưng không có thời gian để tìm và đọc hết các tài liệu để chọn ra tài liệu phục vụ cho nhu cầu ấy (số lượng bài viết hiện nay quá nhiều)

    - Phân loại spam email (chỉ muốn tập trung vào email có ích hoặc tương tự).

- Để giải bài toán này thì có rất nhiều phương pháp được đưa ra ví dụ như thuật toán Naive Bayes, K-NN, Cây quyết định, Mạng Neuron nhân tạo,…

- Trong đồ án lần này em xin phép được thực hiện theo phương pháp Naive Bayes vì những ưu điểm của nó:

    - - Dễ dàng tính toán
    - - Xử lý được số đặc trưng rất lớn
    - - Với tập dữ liệu lớn, độ chính xác của phân lớp sẽ cao.
    - - Được xem là thuật toán đơn giản nhất.

2. Bài toán phân loại văn bản

2.1 Giới thiệu

Như đã trình bày, bài toán phân loại văn bản là một bài toán quan trọng trong xử lý ngôn ngữ. Có khá nhiều bài toán phân loại trong lĩnh vực này như: gán nhãn từ loại (POS tagging), khử nhập nhằng nghĩa từ vựng, gán nhãn tính từ,…

Mỗi bài toán phân loại đều có các đối tượng và mục tiêu phân loại khác nhau. Trong bài toán gán nhãn và khử nhập nhằng nghĩa từ vựng thì từ được xem là đối tượng nội dung cần thao tác. Trong gán nhãn tính từ thì một ngữ là đối tượng nội dung cần thao thác, còn trong bài toán phân loại văn bản thì văn bản là đối tượng cần thao tác.

2.2 Phát biểu bài toán

Bài toán phân loại văn bản có thể được phát biểu như sau:

Cho trước một tập văn bản D = {d1, d2,……,dn} và tập chủ đề được định nghĩa C= {c1, c2,…., cn}
Với nhiệm vụ gán lớp di thuộc về lớp cj đã được định nghĩa.

2.3 Giải quyết bài toán bằng phương pháp Naive Bayes

2.3.1 Naive Bayes Classifier

Xét bài toán classification với CC classes 1,2,…,C1,2,…,C. Giả sử có một điểm dữ liệu x∈Rdx∈Rd. Hãy tính xác suất để điểm dữ liệu này rơi vào class cc. Nói cách khác, hãy tính:

$$p(y = c|\mathbf{x}) \quad (1)$$

hoặc viết gọn thành p(c|x).

Tức tính xác suất để đầu ra là class cc biết rằng đầu vào là vector x.

Biểu thức này, nếu tính được, sẽ giúp chúng ta xác định được xác suất để điểm dữ liệu rơi vào mỗi class. Từ đó có thể giúp xác định class của điểm dữ liệu đó bằng cách chọn ra class có xác suất cao nhất:

$$c = \arg \max_{c \in \{1,\ldots,C\}} p(c|\mathbf{x}) \quad (2)$$

Biểu thức (2) thường khó được tính trực tiếp. Thay vào đó, quy tắc Bayes thường được sử dụng:

$$\begin{aligned} c &= \arg\max_c p(c|\mathbf{x}) && (3) \\ &= \arg\max_c \frac{p(\mathbf{x}|c)p(c)}{p(\mathbf{x})} && 4) \\ &= \arg\max_c p(\mathbf{x}|c)p(c) && (5) \end{aligned}$$

Từ (3) sang (4) là vì quy tắc Bayes. Từ (4) sang (5) là vì mẫu số p(x) không phụ thuộc vào c.

Tiếp tục xét biểu thức (5)(, p(c) có thể được hiểu là xác suất để một điểm rơi vào class cc. Giá trị này có thể được tính bằng MLE, tức tỉ lệ số điểm dữ liệu trong tập training rơi vào class này chia cho tổng số lượng dữ liệu trong tập traing; hoặc cũng có thể được đánh giá bằng MAP estimation. Trường hợp thứ nhất thường được sử dụng nhiều hơn.

Thành phần còn lại p(x|c), tức phân phối của các điểm dữ liệu trong class cc, thường rất khó tính toán vì xx là một biến ngẫu nhiên nhiều chiều, cần rất rất nhiều dữ liệu training để có thể xây dựng được phân phối đó. Để giúp cho việc tính toán

được đơn giản, người ta thường giả sử một cách đơn giản nhất rằng các thành phần của biến ngẫu nhiên x là độc lập với nhau, nếu biết c (given c..) Tức là:

$$p(\mathbf{x}|c) = p(x_1, x_2, \ldots, x_d|c) = \prod_{i=1}^{d} p(x_i|c) \quad (6)$$

Giả thiết các chiều của dữ liệu độc lập với nhau, nếu biết cc, là quá chặt và ít khi tìm được dữ liệu mà các thành phần hoàn toàn độc lập với nhau. Tuy nhiên, giả thiết *ngây ngô* này lại mang lại những kết quả tốt bất ngờ. Giả thiết về sự độc lập của các chiều dữ liệu này được gọi là *Naive Bayes* (xin không dịch). Cách xác định class của dữ liệu dựa trên giả thiết này có tên là *Naive Bayes Classifier (NBC)*.
NBC, nhờ vào tính đơn giản một cách *ngây thơ*, có tốc độ training và test rất nhanh. Việc này giúp nó mang lại hiệu quả cao trong các bài toán large-scale.

Ở bước training, các phân phối p(c)p(c) và p(xi|c),i=1,…,dp(xi|c),i=1,…,d sẽ được xác định dựa vào training data. Việc xác định các giá trị này có thể dựa vào Maximum Likelihood Estimation hoặc Maximum A Posteriori.

Ở bước **test**, với một điểm dữ liệu mới xx, class của nó sẽ được xác đinh bởi:

$$c = \arg \max_{c \in \{1,\ldots,C\}} p(c) \prod_{i=1}^{d} p(x_i|c) \quad (7)$$

Khi d lớn và các xác suất nhỏ, biểu thức ở vế phải của (7) sẽ là một số rất nhỏ, khi tính toán có thể gặp sai số. Để giải quyết việc này, (7) thường được viết lại dưới dạng tương đương bằng cách lấy log của vế phải:

$$c = \arg \max_{c \in \{1,\ldots,C\}} = \log(p(c)) + \sum_{i=1}^{d} \log(p(x_i|c)) \quad (7.1)$$

Việc này không ảnh hưởng tới kết quả vì loglog là một hàm đồng biến trên tập các số dương.
Mặc dù giả thiết mà Naive Bayes Classifiers sử dụng là quá phi thực tế, chúng vẫn hoạt động khá hiệu quả trong nhiều bài toán thực tế, đặc biệt là trong các bài toán

phân loại văn bản, ví dụ như lọc tin nhắn rác hay lọc email spam. Trong phần sau của bài viết, chúng ta cùng xây dựng một bộ lọc email spam tiếng Anh đơn giản.

Cả việc training và test của NBC là cực kỳ nhanh khi so với các phương pháp classification phức tạp khác. Việc giả sử các thành phần trong dữ liệu là độc lập với nhau, nếu biết class, khiến cho việc tính toán mỗi phân phối p(xi|c)p(xi|c) trở nên cực kỳ nhanh.
Mỗi giá trị p(c),c=1,2,…,Cp(c),c=1,2,…,C có thể được xác định như là tần suất xuất hiện của class cc trong training data.
Việc tính toán p(xi|c) phụ thuộc vào loại dữ liệu. Có ba loại được sử dụng phổ biến là: Gaussian Naive Bayes, Multinomial Naive Bayes, và Bernoulli Naive .

2.3.2 Multinomial Naive Bayes

Đồ án lần này sẽ sử dụng Multinomial Naive Bayes để tính toán p(xi|c)

Mô hình này chủ yếu được sử dụng trong phân loại văn bản mà feature vectors được tính bằng Bags of Words. Lúc này, mỗi văn bản được biểu diễn bởi một vector có độ dài dd chính là số từ trong từ điển. Giá trị của thành phần thứ ii trong mỗi vector chính là số lần từ thứ ii xuất hiện trong văn bản đó.

Khi đó, p(xi|c) tỉ lệ với tần suất từ thứ ii (hay feature thứ ii cho trường hợp tổng quát) xuất hiện trong các văn bản của class cc. Giá trị này có thể được tính bằng cách:

$$\lambda_{ci} = p(x_i|c) = \frac{N_{ci}}{N_c} \quad (10)$$

Trong đó:

- Nci là tổng số lần từ thứ ii xuất hiện trong các văn bản của class c, nó được tính là tổng của tất cả các thành phần thứ ii của các feature vectors ứng với class c.

- Nc là tổng số từ (kể cả lặp) xuất hiện trong class c. Nói cách khác, nó bằng tổng độ dài của toàn bộ các văn bản thuộc vào class cc. Có thể suy ra rằng Nc=∑di=1Nci, từ đó ∑di=1λci=1.

Cách tính này có một hạn chế là nếu có một từ mới chưa bao giờ xuất hiện trong class cc thì biểu thức (10) sẽ bằng 0, điều này dẫn đến vế phải của (7) bằng 0 bất kể các giá trị còn lại có lớn thế nào. Việc này sẽ dẫn đến kết quả không chính xác (xem thêm ví dụ ở mục sau).
Để giải quyết việc này, một kỹ thuật được gọi là *Laplace smoothing* được áp dụng:

$$\hat{\lambda}_{ci} = \frac{N_{ci} + \alpha}{N_c + d\alpha}$$

Với αα là một số dương, thường bằng 1, để tránh trường hợp tử số bằng 0. Mẫu số được cộng với dαdαđể đảm bảo tổng xác suất ∑di=1^λci=1.
Như vậy, mỗi class cc sẽ được mô tả bởi bộ các số dương có tổng bằng 1: ^λc={^λc1,…,^λcd}.

II. Thu thập văn bản

1. Mục tiêu:

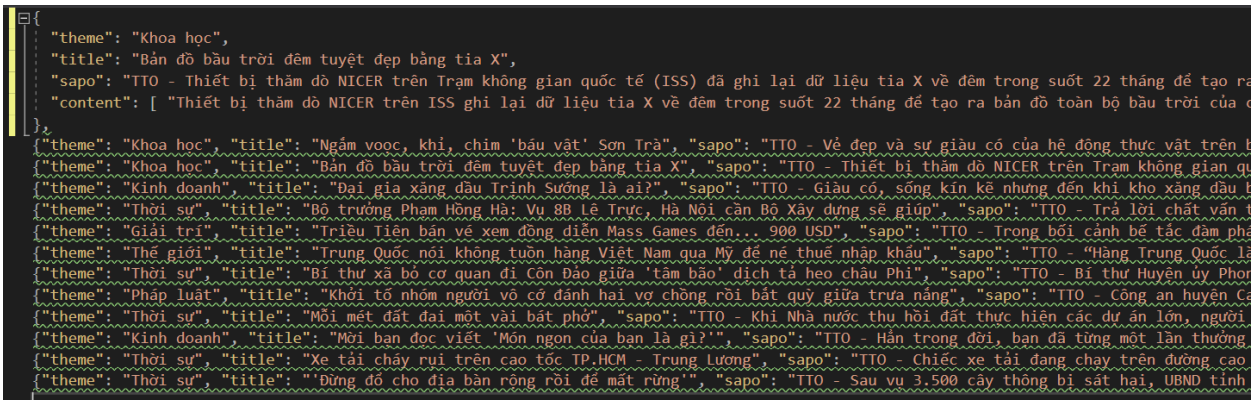
Thu thập văn bản thuộc hai lĩnh vực khác nhau Tổng số lượng văn bản thu thập được khoảng 80-100 văn bản với tỉ lệ mỗi loại tùy chọn nhưng không quá lệch về một lĩnh vực. Chia tổng số văn bản này thành hai phần: training và test. Train khoảng 80% và test khoảng 20% và giữ đúng tỉ lệ văn bản ở mỗi lĩnh vực như trong tổng số.

2. Thực hiện:

Dùng hàm **Scrapy** để bóc tách dữ liệu trên báo *Dân trí* với*Vnexpress* lưu lại dưới file *data1.json* và *data2.json* lần lượt dùng để traning và test.

data1.json	6/8/2019 5:53 PM	JSON File	4,529 KB
data2.json	6/8/2019 3:20 PM	JSON File	2,090 KB

- Sau đó xử lý file .json về 2 Lớp lần lượt là ***Công nghệ*** và ***Pháp Luật*** đồng thời tiền xử lý văn bản về một dòng




```php
// Output one line until end-of-file
while(!feof($myfile)) {
  $jsonLineStr = fgets($myfile);
  $jsonLineStr = substr($jsonLineStr, 0, -2);
  $jsonData = json_decode($jsonLineStr);
  //echo $jsonLineStr. '<br>';
  //echo $jsonData->theme. '<br>';

  if($jsonData->theme == 'Công nghệ') {
    $content[$jsonData->theme] = empty($content[$jsonData->theme]) ? implode("|", $jsonData->content)
        : $content[$jsonData->theme] . '|'. implode("|", $jsonData->content);
    //echo $jsonData->theme.'<br>';
  }
}
```

III. Tiền xử lý văn bản

3.1 Mục tiêu:

Sau khi đưa về dạng văn bản "chuẩn" sau khi thực hiện các thao tác tiền xử lý sau:

1. Chuyển văn bản thành chữ viết thường.

2. Loại bỏ non-words: Số, dấu câu, ký tự ‘tabs’, ký tự ‘xuống dòng’ và các kí tự đặc biệt.
3. Loại bỏ stop-words: Những từ xuất hiện thường xuyên trong hầu hết các văn bản

3.2 Thực hiện

3.2.1 Chuyển văn bản về chử viết thường

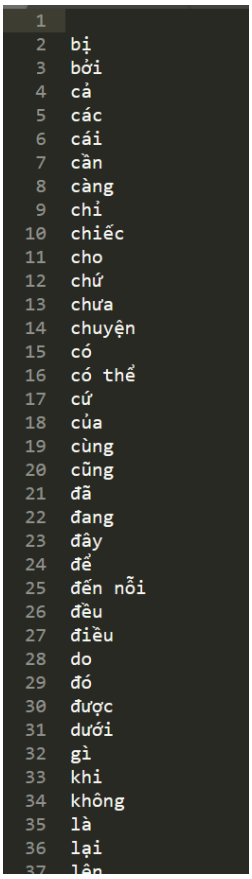
PHP có hỗ trợ hàm strtolower để chuyển văn bản về chữ viết thường

```php
$content = strtolower($content);
```

3.2.2 Loại bỏ non-words.

```php
function cleanSpecialCharacter ($string){
    $string = str_replace("\n",' ', $string);
    $string = str_replace("\r",' ', $string);
    $string = str_replace('\'',' ', $string);
    $string = str_replace('`',' ', $string);
    $string = str_replace(':',' ', $string);
    $string = str_replace('.',' ', $string);
    $string = str_replace(',',' ', $string);
    $string = str_replace('?',' ', $string);
    $string = str_replace('(',' ', $string);
    $string = str_replace(')',' ', $string);
    $string = str_replace('{',' ', $string);
    $string = str_replace('-',' ', $string);
    $string = str_replace('|',' ', $string);
    $string = str_replace(';','', $string);
    $string = str_replace('/',' ', $string);
    $string = str_replace('\\',' ', $string);
    $string = str_replace("\"",'', $string);
```

3.2.3 Loại bỏ stop-words



```
1
bị
bởi
cả
các
cái
cần
càng
chỉ
chiếc
cho
chứ
chưa
chuyện
có
có thể
cứ
của
cùng
cũng
đã
đang
đây
để
đến nỗi
đều
điều
do
đó
được
dưới
gì
khi
không
là
lại
lên
```

Sưu tập các stop-words của tiếng việt xử lý để loại bỏ.
Nguồn: Gifthub

(Xử lý loại bỏ stop – words – PHP)

```php
//remove stop word
$myfile = fopen("upload/stopwordvietnamese.txt", "r") or die("Unable to open file!");
$stopWordContent = array();
// Output one line until end-of-file

while(!feof($myfile)) {
  $stopWordContent[] = fgets($myfile);
}

fclose($myfile);
```

File được lưu trữ sau khi thực hiện các quá trình trên (Ảnh minh hoạ thể hiện nội dung File):

```
{"ba":144,"đối":132,"tượng":52,"công":604,"an":1444,"cam":38,"lộ":46,"
```

IV. Huấn luyện bộ phân lớp theo Naive Bayes

1. Ví dụ (nguồn: internet)

Giả sử trong tập training có các văn bản d1, d2, d3, d4 như trong bảng dưới đây. Mỗi văn bản này thuộc vào 1 trong 2 classes: B (*Bắc*) hoặc N (*Nam*). Hãy xác định class của văn bản d5.

	Document	Content	Class
Training	d1	hanoi pho chaolong hanoi	B
	d2	hanoi buncha pho omai	B
	d3	pho banhgio omai	B
	d4	saigon hutiu banhbo pho	N
Test	d5	hanoi hanoi buncha hutiu	?

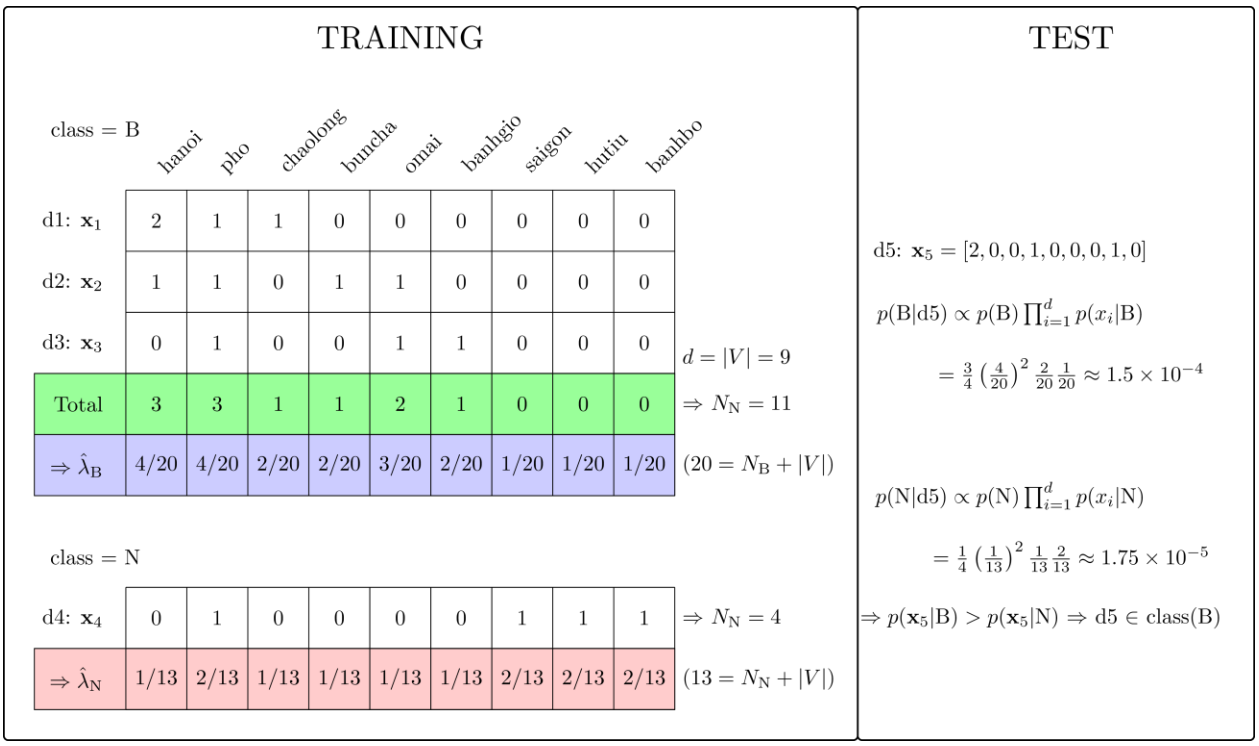
Ta có thể dự đoán rằng d5 thuộc class *Bắc*.

Bài toán này có thể được giải quyết bởi hai mô hình: Multinomial Naive Bayes và Bernoulli Naive Bayes. Tôi sẽ làm ví dụ minh hoạ với mô hình thứ nhất và thực hiện code cho cả hai mô hình. Việc mô hình nào tốt hơn phụ thuộc vào mỗi bài toán. Chúng ta có thể thử cả hai để chọn ra mô hình tốt hơn.

Nhận thấy rằng ở đây có 2 class B và N, ta cần đi tìm p(B) và p(N). à dựa trên tần số xuất hiện của mỗi class trong tập training. Ta sẽ có:

$$p(\mathrm{B}) = \frac{3}{4}, \quad p(\mathrm{N}) = \frac{1}{4} \qquad (8)$$

Hình dưới đây minh hoạ quá trình Training và Test cho bài toán này khi sử dụng Multinomial Naive Bayes, trong đó có sử dụng Laplace smoothing với α=1.



Chú ý, hai giá trị tìm được 1.5×10−41.5×10−4 và 1.75×10−51.75×10−5 không phải là hai xác suất cần tìm mà chỉ là hai đại lượng tỉ lệ thuận với hai xác suất đó. Để tính cụ thể, ta có thể làm như sau:

$$p(\mathrm{B}|\mathrm{d5}) = \frac{1.5 \times 10^{-4}}{1.5 \times 10^{-4} + 1.75 \times 10^{-5}} \approx 0.8955, \quad p(\mathrm{N}|\mathrm{d5}) = 1 - p(\mathrm{B}|\mathrm{d5}) \approx 0.1045$$

2. Thực Hiện đối với file training

2.1 Tạo từ điển (tập hợp các từ trong văn bản)

Kí hiệu: |V| = d (với alpha = 1)

```php
$mergeArray = $contentClass1Arr;

foreach ($contentClass2Arr as $key => $value) {
    if($mergeArray[$key]) {
        $mergeArray[$key] += $value;
    } else {
        $mergeArray[$key] = $value;
    }
}

$totalD = count($mergeArray);
$totalNClass1 = array_sum($contentClass1Arr);
$totalNClass2 = array_sum($contentClass2Arr);
```

2.2 Tính lamda

```php
//calculate lamda class 1
$lamdaClass1 = array();

foreach ($mergeArray as $key => $value) {
    $lamdaClass1[$key] = ($contentClass1Arr[$key] + 1)/ ($totalD + $totalNClass1);
}

//save lamda file class 1
$filename = "training/lamda" . $class1 . ".json";

unlink($filename);
$classFile = fopen($filename, "w") or die("Unable to open file!");
fwrite($classFile, json_encode($lamdaClass1, JSON_UNESCAPED_UNICODE));
fclose($classFile);
```

Kết quả của quá trình được lưu lại như sau:


```
{"Ảnh":0.0009255850644228241,"minh":0.0007362608466999737,"họa":9.466210886142519e-5,
```

3 Thực hiện đối với file test

3.1 Tiền xử lý file test tương tự như file training

```php
//get test content
$fileName = 'test/test.txt';
$testContent = getContentFromTestFile($fileName);
$testContent = strtolower($testContent);
$testContent = cleanSpecialCharacter($testContent);
```

3.2 Loại bỏ những từ không có trong từ điển

```php
// những từ không có trong file merge sẽ bỏ đi
$mergeContent = file_get_contents($fileName);
$mergeContent = file_get_contents("training/merge.json");
$mergeContentArr = json_decode($mergeContent, true);

foreach ($countEachCharacter as $key => $value) {
    if(!$mergeContentArr[$key])
        unset($countEachCharacter[$key]);
}
```

3.3 Tính và so sánh xác xuất

```php
//calculate class1
$contentClass1 = file_get_contents("training/". $class1 .".json");
$contentClass1Arr = json_decode($contentClass1, true);
$class1Percent = array_sum($contentClass1Arr) / array_sum($mergeContentArr);
$lamda = 1;

//get lamda class1
$lamdaClass1 = file_get_contents("training/lamda". $class1 .".json");
$lamdaClass1Arr = json_decode($lamdaClass1, true);


foreach ($countEachCharacter as $key => $value) {
    $temp = $lamda;
    $t = pow($lamdaClass1Arr[$key], $value);
    $temp *= $t;
    if ($temp == 0) {
        $lamda = $lamda * pow(10,300) * $t;
    } else {
        $lamda = $temp;
    }

    //$lamda = $lamda * pow($lamdaClass1Arr[$key], $value);
}

$resultClass1 = $lamda * $class1Percent;
```

3.4 Tính tỉ lệ thuộc văn bản có sẵn

Nếu đoạn văn bản liên quan đến chủ đề khác chủ đề có sẵn?

```
$ratio1 = array_sum($countEachCharacterclass1) / array_sum($countEachCharacteroriginal);
$ratio2 = array_sum($countEachCharacterclass2) / array_sum($countEachCharacteroriginal);
$ratio = ($ratio1 > $ratio2) ? $ratio1 : $ratio2 ;
```

- Dựa vào quá trình test cho thấy đa số kết quả phân đúng đều có tỉ lệ $ratio > 80%

4 Đánh giá kết quả phân lớp văn bản

Chuẩn bị 40 bài test: - 20 bài thuộc lớp chủ đề có sẵn (50:50).
- 20 bài thuộc chủ đề khác.

	Gán đúng Lớp	Gán sai Lớp
Lớp chủ đề có sẵn	a: *Thuộc chủ đề có sẵn và gán đúng Lớp*	c: *Thuộc chủ đề có sẵn gán sai lớp*
Lớp chủ đề không có sẵn	b: *Không thuộc chủ đề có sẵn mà vẫn gán lớp*	d: *Không thuộc chủ đề có sẵn nên không gán lớp*

Các giá trị sau khi test lần lượt là: a: 19 c: 1
b: 8 d: 12

- Để đánh giá chất lượng của bộ phân lớp ta đánh giá qua tiêu chí sau:
  - Độ chính xác (accuracy) $\frac{a+d}{a+b+c+d}$= 0,775
  - Độ sai lỗi (Error) $\frac{b+c}{a+b+c+d}$ = 0,225

- Trong trường hợp chỉ xem xét đối tượng thuộc về lớp phân lớp đúng do đó một số độ đo khác được định nghĩa:
    - Độ chính xác (Precision): P = $\frac{a}{a+c}$ = 0,95
    - Độ bao phủ (Recall): R = $\frac{a}{a+b}$ = 0,703
    - Độ loại bỏ (fallout): $\frac{b}{b+d}$ = 0.4

- Tuy nhiên trong thực tế độ chính xác (Precision) và độ bao phủ (Recall) sẽ không cân đối vì vậy để thuận tiện họ kết hợp lại thành một đơn vị đo tổng quát duy nhất. người ta sử dụng đơn vị đo lường F được định nghĩa như sau:

$$F = \frac{1}{\alpha \frac{1}{P} + (1-\alpha)\frac{1}{R}}$$

    - Giá trị alpha = 5 thường được dùng để duy trì cân bằng giữa P và R với giá trị này độ đo được tính đơn giản như sau: 2PR/(P+R) = 0,81.

- Trên thực tế đối với bộ phân lớp chia thành nhiều lớp để đánh giá sau khi tập hợp thành bảng thống kê 2 phương pháp được đưa ra là
    - Macro-Avedagin: đây là phương pháp tính trung bình độ chính xác (Precision) và độ bao phủ (Recall)

$$P_{macro} = \frac{1}{|C|}\sum_{i=1}^{|C|}\frac{a_i}{a_i + b_i}$$

$$R_{macro} = \frac{1}{|C|}\sum_{i=1}^{|C|}\frac{a_i}{a_i + c_i}$$

- |C|: Là số lớp cần phân loại

- Micro-Avedagin: Đây là phương pháp tính trung bình kết quả thống kê của từng lớp được tính sau khi lập bảng thống kê:

$$P_{micro} = \frac{\sum_{i=1}^{|C|} a_i}{\sum_{i=1}^{|C|}(a_i + b_i)}$$

$$R_{micro} = \frac{\sum_{i=1}^{|C|} a_i}{\sum_{i=1}^{|C|}(a_i + c_i)}$$

-

- |C|: Là số lớp cần phân loại