B02902015 資工三 梁智泓
Programing Hw2

- ## How to Compile & Execute:

Compile: 只需要在檔案的路徑下輸入: ./compile.sh，因為是使用 Python 實作，所以 compile.sh 並不會做任何事

Run: 依序輸入

./naivebayes.sh -i $(DataDir) -o $(output_file) [-n $(Label_num)]

./EM.sh -i $(DataDir) -o $(output_file) [-n $(Label_num)]

即可分別執行 Naïve Bayes 及 EM algorithm

- ## Naïve Bayes Classifier:

$$P(C_k|\theta)\prod P(x_{ij}|C_k;\theta)$$

前項: P(Ck | theta) 為各種類中的文章數機率，也就是特定種類的文章數除以總文章數

後項: P(xij | Ck; theta) 為各種類中對某文字 x 的 unigram 機率加上 x 在 background 中的機率(smoothing)，式子如下

$$\prod_{i=1}^{|V|}[\lambda_i p(w_i \mid C)+(1-\lambda_i)p(w_i \mid Background)]^{n_i}$$

Lamda 我取 0.8，因經過測試之後，發現 0.8 時整體的 performance 會最好

Implement detail: 一開始先將 Train 和 Unlabel 的文件中的所有文字都讀出來，只留下英文，並且將其全部轉成小寫，再對文字做 stemming，將同意義但不同形式的文字縮成同一個形式，建立一個 BackgroundVocab 的總表，以便算出各文字在 background 的機率，然後再讀 Train data 中的所有檔案，建立一個各種類中的所有文字的 Model，Model 上的機率為上面數學式中所示，然後取 Log 以便計算，最後再將 Test 中的每篇 Document 透過 Model 去 Predict 出文章是各種類的機率，並且以機率最大者作為 Predict 的結果，最後將所有 Test Document Predict 的結果輸出到指定的 Output_file 中

- ## EM algorithm

E step:

Initial: 用一開始的 Naïve Bayes Model

對每篇 Unlabel 的文件，透過 Fine tune 過的 Model 去 predict 出屬於哪個 category

M step:

透過 E-step 的結果，將 Unlabel Doc Predict 出來的結果加到 Category 中，透過一開始的 Naïve Bayes Model 去重建出一個新 Model，然後跑下一輪

Termination:

當新一輪重建的 Model 與上一輪重建的 Model 長得一樣時，就終止，並且將重建的 Model 作為最後的 Model 去 Predict Test

Experiment

	N=1	N=10	N=20	N=50	N=100
Naïve	0.157448	0.474467	0.594755	0.708674	0.752946
EM	0.276038	0.604841	0.681813	0.743922	0.781824

Technique of Implement

Smoothing: 做了上述的 Smooth，讓整體的準確度上升，因為一篇文章中，如果有越多相同字，理論上應該要越有可能是相同的總類，但如果是直接將每個字的機率相乘，越多相同字的文章，整體的機率就會越來越低，因為機率都是小於 1 的，越乘只會越小，所以才使用了上述的 Smoothing，將每個字在 Background 的機率也考慮進去，使預測的結果更符合預期與直覺

Stemming: 除了 Smooth，我還將每個讀進來的文字只留下英文，並且其全部轉成小寫，再透過 NLTK 的套件，對文字做了 stemming，將同意義但不同形式的文字轉成同形式，以便減少 Vocab 的總數量，從 77024 個不同字減少到 59314，降低整體 Model 的複雜度，也加速整個 Program，更提升了準確度。

	Vocab Size	Precision
No stemming	77024	0.750929
Stemming	59314	0.752946

Observation

由上述的實驗，其實可以很清楚地看到，當 Label 文件數量越少時，EM 所提升得準確度便越多，然而相反的，當 Label 的文件數越少時，EM 的提升就變得不顯著，甚至有可能會降低準確度，因為 Label 的文件已足夠完整，也就是足以詮釋出完整的 Hidden Variable，Unlabel 能做到的提升便越來越有限。

這樣的現象，更可以從下表中在做 EM 時要達到 Optimal 所需要的 Iteration 數可以看到，當 Label 的數量越來越多時，需要達到的 iteration 數就越少，也就是原本的 model 就已經足夠接近 Optimal。

Label_docment_Num	N=1	N=10	N=20	N=50	N=100
Iteration	21	15	12	6	7