一、執行環境 & 作業系統

Mac Command Shell

二、程式語言, 版本

Python 3.5.2

三、執行方式

(1) 安裝以下 package：tqdm@4.19.4、nltk@3.2.5

```
~/Desktop/IRTM/hw/hw3 sudo pip3 install 'tqdm==4.19.4' 'nltk==3.2.5'
```

sudo pip3 install 'tqdm==4.19.4' 'nltk==3.2.5'

(2) 安裝 nltk 裡的 stopWord Corpus

```
ken@Weal222C:~/IRTM/hw/hw3$ python3
Python 3.5.2 (default, Nov 23 2017, 16:37:01)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>> nltk.download()
NLTK Downloader
---------------------------------------------------------------------------
    d) Download   l) List    u) Update   c) Config   h) Help   q) Quit
---------------------------------------------------------------------------
Downloader> d

Download which package (l=list; x=cancel)?
  Identifier> stopwords
    Downloading package stopwords to /home/ken/nltk_data...
      Unzipping corpora/stopwords.zip.
```

python3 > import nltk > nltk.download() > d > stopwords

(3) (可省略此步驟)：**python3 Train.py (若要執行需新增一目錄 IRTM 裝所有的原始 txt 檔案)**

```
~/Desktop/IRTM/hw3 python3 Train.py
1. Establish the [vocabulary]: 100%|| 195/195 [00:05<00:00, 37.31it/s]
2. Select top 500 features: 100%|| 13/13 [00:04<00:00,  2.86it/s]
   Estimate & compate scores(threshold: appear 10 times): 100%| 5563/5563 [00:01<00:00, 3086.23it/s]
3. Calculate [prior] & [condition]: 100%|| 13/13 [00:05<00:00,  2.50it/s]

================================ Save parameters in './V_prior_condprob.pkl' ================================
```

(4) 執行：**python3 Test.py**

```
~/Desktop/IRTM/hw3 python3 Test.py
Generate [output.txt]: 100%| 1095/1095 [00:27<00:00, 39.72it/s]

============================== Save outputs in './output.txt' ==============================
```

四、作業處理邏輯說明

主邏輯程式主要分為三個檔案：(1) Normalize.py (2) Train.py (3) Test.py

(1) Normalize.py

將作業一的程式移植至作業三，主要做的事情為

- 切詞處理(tokenize)
- 將所有字符轉為小寫
- Stemming
- 移除特殊字元
- 移除 stopWords

(2) Train.py

- 以 **gatherAllTrainingDocId** 取得所有以分類的文件 id：**D**
- 以 **ExtractVocabulary** 建好根據 **D** 的詞庫：**preV**
- 以 **FeatureSelect** 對 **preV** 篩選前 500 對分類有鑑別力的詞(以 chi-square 計算)：**V**
- 以 **CountDocs** 計算出 **N**
- 以 **CountDocsInClass** 計算每個 class 不同的 prior 值：**prior**
- 以 **CountTokensOfTerm** 計算每個不同 **term** 出現在不同 class 的分數(出現在該 class 次數+1/出現在全部 class 的次數+1)：**condprob**
- 輸出 **V_prior_condprob.pkl**(為一個包含 **V**, **prior**, **condprob** 的 object)

(3) Test.py

- 以 **gatherAllTrainingDocId** 取得所有以分類的文件 id，排除之。
- 根據 **V_prior_condprob.pkl** 計算每個文件不同 class 的 score(prior * condprob(**getHighestScoreClass**))，再依最高的 score 選相對應的 class(**getClass**)。
- 輸出 **output.txt**。