R10725011 陳佑甄

作業三

1. 執行環境

   本機開 Anaconda，建立虛擬環境，下 python 去跑.py 檔。

   用 Visual Studio Code 撰寫 python 程式碼。

2. 程式語言 (請標明版本)

   Python 3.7.11

3. 執行方式

   本機開啟 Anaconda Powershell Prompt，Anaconda 版本是 conda 4.10.3，

Anaconda Powershell Prompt (anaconda)

應用程式

使用 nltk 套件。

- 創建虛擬環境：**(以下我有先創虛擬環境，但也可以不創建)**

```
$conda create --name IR python=3.7
```

下`$conda env list`可以看到剛剛創建的虛擬環境。

```
(base) PS C:\Users\[redacted]> conda env list
# conda environments:
#
base                  *  D:\anaconda
AIHW1                    D:\anaconda\envs\AIHW1
IR                       D:\anaconda\envs\IR
pyhon3.7                 D:\anaconda\envs\pyhon3.7
```

用`$ conda activate IR`可以啟動剛剛創好的虛擬環境。

- 安裝 nltk 套件：**(若沒有 nltk 套件一定需要 install)**

然後由於有用到 nltk 套件抓 stopword list ，

所以要下`$conda install -c anaconda nltk`。

然後可以透過`$conda list`看目前環境中載的所有套件。

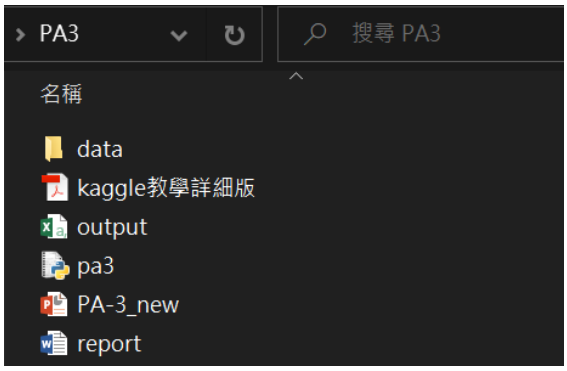
其他套件是在 create 虛擬環境時有問要不要安裝，我當時選 yes。

```
(IR3) PS D:\document\研究所\碩一上\資料檢索與文字探勘\PA3> conda list
# packages in environment at D:\anaconda\envs\IR3:
#
# Name                    Version                   Build  Channel
ca-certificates           2020.10.14                    0    anaconda
certifi                   2020.6.20                py37_0    anaconda
click                     7.1.2                      py_0    anaconda
joblib                    0.17.0                     py_0    anaconda
nltk                      3.5                        py_0    anaconda
openssl                   1.1.1                h2bbff1b_0
pip                       21.2.4           py37haa95532_0
python                    3.7.11               h6244533_0
regex                     2020.10.15       py37he774522_0    anaconda
setuptools                58.0.4           py37haa95532_0
sqlite                    3.37.0               h2bbff1b_0
tqdm                      4.50.2                     py_0    anaconda
vc                        14.2                 h21ff451_1
vs2015_runtime            14.27.29016          h5e58377_2
wheel                     0.37.0             pyhd3eb1b0_1
wincertstore              0.2              py37haa95532_2
(IR3) PS D:\document\研究所\碩一上\資料檢索與文字探勘\PA3>
```

- 執行程式：

去到放 pa3.py 的資料夾，`$python .\pa3.py`就可以執行程式碼。

由於會用到先前的資料，放在與 py 檔同一層資料夾中的 data 資料夾。



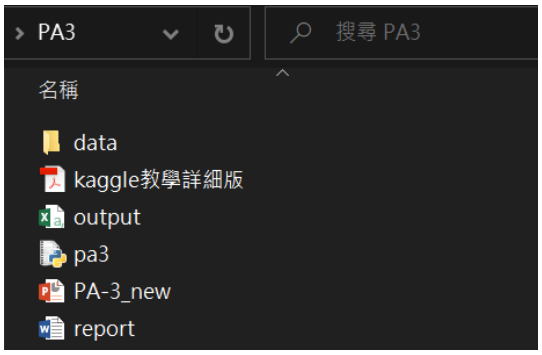
執行結果如下圖，會印出執行到哪一階段，分別是 GetAllTermTable、ExtractVoc、Training、Testing。

```
(IR3) PS D:\document\研究所\碩一上\資料檢索與文字探勘\PA3> python .\pa3.py
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Annie\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
GetAllTermTable class: 1
GetAllTermTable class: 2
GetAllTermTable class: 3
GetAllTermTable class: 4
GetAllTermTable class: 5
GetAllTermTable class: 6
GetAllTermTable class: 7
GetAllTermTable class: 8
GetAllTermTable class: 9
GetAllTermTable class: 10
GetAllTermTable class: 11
GetAllTermTable class: 12
GetAllTermTable class: 13
```

```
ExtractVoc class: 1
ExtractVoc class: 2
ExtractVoc class: 3
ExtractVoc class: 4
ExtractVoc class: 5
ExtractVoc class: 6
ExtractVoc class: 7
ExtractVoc class: 8
ExtractVoc class: 9
ExtractVoc class: 10
ExtractVoc class: 11
ExtractVoc class: 12
ExtractVoc class: 13
```

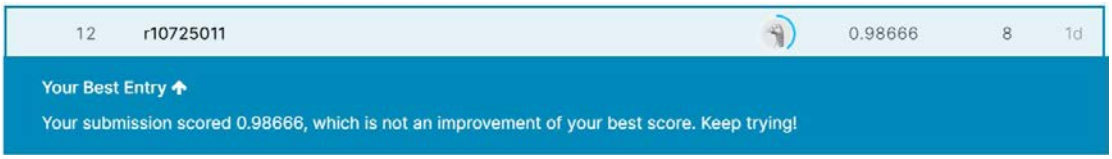
```
TrainMultinomialNB class: 1
TrainMultinomialNB class: 2
TrainMultinomialNB class: 3
TrainMultinomialNB class: 4
TrainMultinomialNB class: 5
TrainMultinomialNB class: 6
TrainMultinomialNB class: 7
TrainMultinomialNB class: 8
TrainMultinomialNB class: 9
TrainMultinomialNB class: 10
TrainMultinomialNB class: 11
TrainMultinomialNB class: 12
TrainMultinomialNB class: 13
Finish Training
Start Testing
Finish All Testing
Finish Writing output.csv
(IR3) PS D:\document\研究所\碩一上\資料檢索與文字探勘\PA3>
```

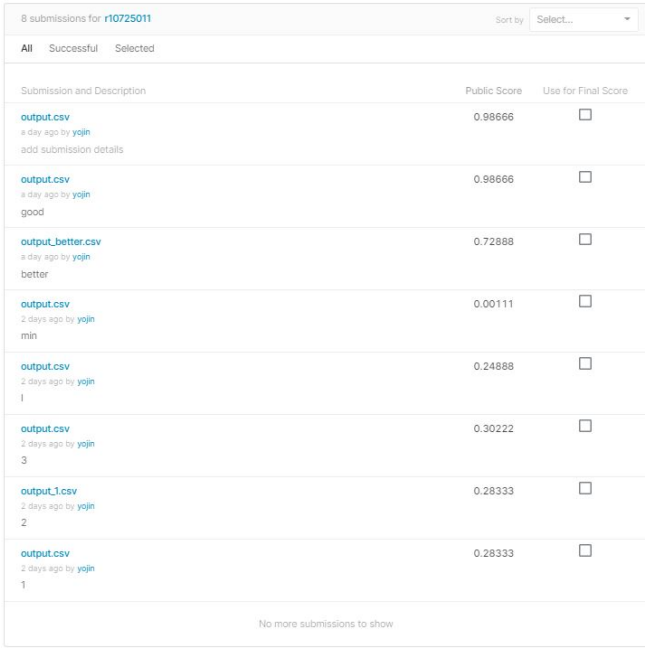
印出 Finish Writing output.csv 代表程式執行完成，會在與 py 檔的同一層資料夾產生出 output.csv。點開 csv 檔部分輸出如圖所示。



	Id	Value
2	17	2
3	18	2
4	20	2
5	21	2
6	22	2
7	23	2
8	24	2
9	25	2
10	26	2
11	27	2

最後放到 kaggle 上，成績(0.98666)與所有 submission 如圖。





一開始分數很爛是因為程式寫錯了，後來調整好就有變好。原本算 likelihood 的方法是算出 13 個結果後取平均再取前 500 高的當作 top features，但成效沒有很好，後來改成每個 class 都取前 38 個高的，最後再合併整理去除重複的作為 top features。

4. 作業處理邏輯說明

- import

```
import nltk
import urllib.request
import math
import csv
import copy
from nltk.stem import PorterStemmer
nltk.download('stopwords')
```

- 大綱

```
# Read training set from url
```

主程式一開始先從網站上抓取 training.txt，去抓出所有 training docID 與 class。透過 docID 從上次給的 data 中，將所有需要的 training data 存成一個叫做 class_docID_doc 的 dict。

```
 # Training
```

呼叫 TrainMultinomialNB function 進行 training，裡面會呼叫 ExtractVoc

function 利用 likelihood ratio 得出 top feature 再利用 NB 演算法進行訓練。

```
# Testing & Output csv
```

將非訓練資料以外的視為測試資料。透過 for 迴圈放入每一筆測試資料，每次都呼叫 ApplyMultinomialNB function 進行測試，並記錄結果。最後將結果輸出為 output.csv。

- 自訂義 function 說明
  - **def tokenize(text)**

進行資料處理。包含 Lowercasing、Tokenization、去除標點符號/數字/stopword 與利用 PorterStemmer 進行 stemming。

  - **def likelihood(n11, n10, n01, n00)**

根據講義公式，實作 likelihood ratio。每一個 term 都會透過此 function 得到一個他在每一個 class 的 likelihood ratio。

```
# Likelihood
def likelihood(n11, n10, n01, n00):
  N = n11 + n10 + n00 + n01
  pt = (n11 + n01) / N
  p1 = n11 / (n11 + n10)
  p2 = n01 / (n01 + n00)
  # print(N, pt, p1, p2, n11, n10, n01, n00)
  # -2logλ = -2log(L(H1) / L(H2))
  result = 0
  up = math.pow(pt, n11) * math.pow((1-pt), n10) * math.pow(pt, n01) *
math.pow((1-pt), n00)
  down = math.pow(p1, n11) * math.pow((1-p1), n10) * math.pow(p2, n01) *
math.pow((1-p2), n00)
  result = (-2) * math.log(up/down, 10)
  return result
```

  - **def ExtractVoc(D, all_term_table)**

利用雙層 for 迴圈，每個 class 都去跑一次所有 term，呼叫 likelihood function 得到結果存在 V 這個 dict 中。每個 class 都取出前 38 高的 term，最後合併起來並去除重複的 term 作為 top features (共 484 個)。

```
# Extract Vocob and output top features
def ExtractVoc(D, all_term_table):
```

```
# D is dictionary {class:{docIDs:[doc text]}}
# all_term_table = {term: [0[pre, abs],...,12[pre, abs]]}
  # Prepare for calling Likelihood
  N = CountDocs(D)
  V = {}
  for c in range(len(D)):
    print("ExtractVoc class:", c+1)
    V[c+1] = {}
    for term in all_term_table:
      n11 = all_term_table[term][c][0]
      n10 = all_term_table[term][c][1]
      n00 = 0
      n01 = 0
      # print(len(all_term_table[term]))
      for i in range(len(all_term_table[term])):
        if i != c:
          n01 += all_term_table[term][i][0]
          n00 += all_term_table[term][i][1]
      V[c+1][term] = likelihood(n11, n10, n01, n00)

    # Get (500 - (500%13))/13 highest value from every classes'
likelihood dic V = {class:{term:val}}
    # (500 - (500%13))/13
    feature_num_perclass = int(( 500 - ( 500 %
len(all_term_table[term]) ) )/ len(all_term_table[term]))
    # Sort by value
    V[c+1] = dict(sorted(V[c+1].items(), key=lambda item: item[1],
reverse=True)[:feature_num_perclass])
  # print(V)

  # Get top_feature by merging all terms in V
  top_feature = []
  for c in range(len(V)):
    for term, val in V[c+1].items():
      top_feature.append(term)
  top_feature = list(set(top_feature))
  return  top_feature # top_feature = [term*484]
```

- **def CountDocs(D)**：用來算總共有幾篇文章

- **def CountDocsInClass(D, c)**：用來算某個 class 有幾篇文章
- **def ConcatTextOfAllDocsInClass(D, c)**：用來合併某 class 的文章
- **def CountTokensOfAllTerm(textc, top_feature)**：用來產生在某個 class 中所有的 term 與他們的 tf。
- **GetTermDfOfAllDocs(D)**：用來拿到所有 term 與他們的 df。
- **def GetAllTermTable(D)**：去算所有 term 在所有 class 的 present 與 absent df。
- **def TrainMultinomialNB(Class, D)**

實作 add one 的 Naïve Bayes Training。

```
# Train of Multinomial NB
def TrainMultinomialNB(Class, D): # D is dictionary {class:{docIDs:[doc
text]}}
  all_term_table = GetAllTermTable(D) # all_term_table = {term: [c1[pre,
abs],...,c13[pre, abs]]}
  top_feature = ExtractVoc(D, all_term_table) # top_feature = [term*484]
  N = CountDocs(D)
  prior = []
  Tct = {}
  condprob = [[0 for i in range(Class)] for j in
range(len(top_feature))]
  # condprob = {} #{term:[class_prob*13]}
  for c in range(Class):
    print("TrainMultinomialNB class:", c+1)
    Nc = CountDocsInClass(D, c+1)
    prior.append(Nc / N)
    textc = ConcatTextOfAllDocsInClass(D, c+1)
    Tct = CountTokensOfAllTerm(textc, top_feature)
    total_tf = sum(Tct.values())
    for i in range(len(top_feature)):
      if Tct.get(top_feature[i], 0) == 0:
        condprob[i][c] = (0 + 1) / (total_tf + len(top_feature))
      else:
        condprob[i][c] = (Tct[top_feature[i]] + 1) / (total_tf +
len(top_feature))
```

```
  # print("condprob: ", condprob)
  # print("prior: ", prior)
  print("Finish Training")
  return top_feature, prior, condprob
```

- **def ApplyMultinomialNB(Class, V, prior, condprob, d)**

實作 add one 的 Naïve Bayes Testing。

```
# Test of Multinomial NB
def ApplyMultinomialNB(Class, V, prior, condprob, d):
  tokens = tokenize(d)
  W = []
  for t in tokens:
    if t in V:
      W.append([t, V.index(t)])

  score = copy.deepcopy(prior) # prior = [class_val*13]
  for c in range(Class):
    score[c] = math.log(prior[c], 10)
    for i in range(len(W)):
      term_index = W[i][1]
      score[c] += math.log(condprob[term_index][c], 10)
  return score.index(max(score))+1
```