資訊檢索語文字探勘導論  PA3

1. 執行環境：VS Code

2. 程式語言：Python 3.10.9

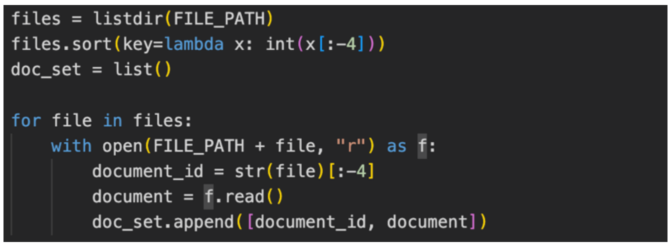
3. 執行方式

（1）在 terminal 使用 pip install nltk 和 re 套件

（2）執行 pa3.py

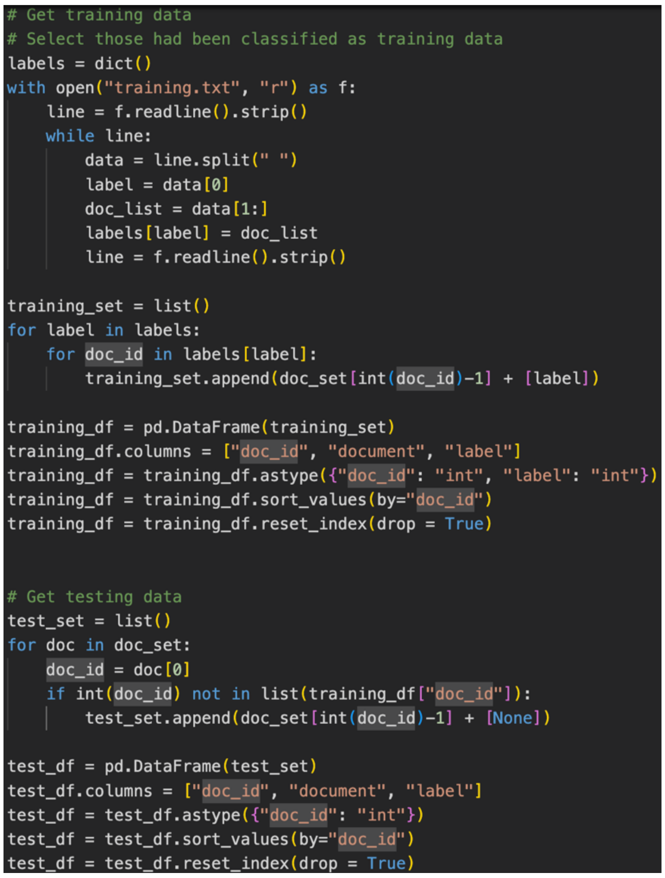
4. 處理邏輯

（1）讀入所有 data 資料夾中的所有 txt 檔



```python
files = listdir(FILE_PATH)
files.sort(key=lambda x: int(x[:-4]))
doc_set = list()

for file in files:
    with open(FILE_PATH + file, "r") as f:
        document_id = str(file)[:-4]
        document = f.read()
        doc_set.append([document_id, document])
```

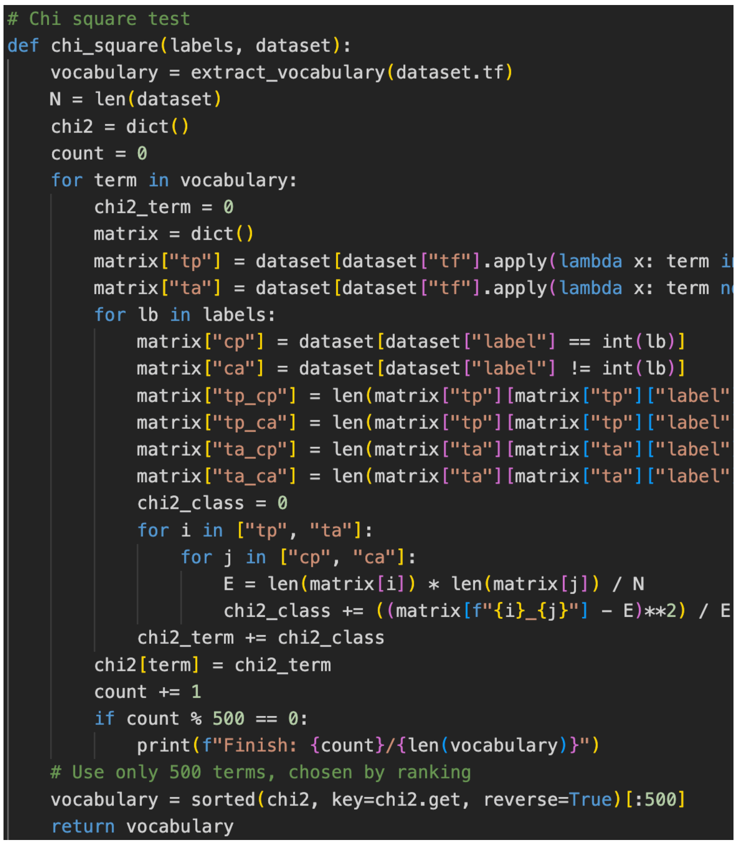
（2）將文件依照已標籤和未標籤，分為  training data  和  testing data



```python
# Get training data
# Select those had been classified as training data
labels = dict()
with open("training.txt", "r") as f:
    line = f.readline().strip()
    while line:
        data = line.split(" ")
        label = data[0]
        doc_list = data[1:]
        labels[label] = doc_list
        line = f.readline().strip()

training_set = list()
for label in labels:
    for doc_id in labels[label]:
        training_set.append(doc_set[int(doc_id)-1] + [label])

training_df = pd.DataFrame(training_set)
training_df.columns = ["doc_id", "document", "label"]
training_df = training_df.astype({"doc_id": "int", "label": "int"})
training_df = training_df.sort_values(by="doc_id")
training_df = training_df.reset_index(drop = True)


# Get testing data
test_set = list()
for doc in doc_set:
    doc_id = doc[0]
    if int(doc_id) not in list(training_df["doc_id"]):
        test_set.append(doc_set[int(doc_id)-1] + [None])

test_df = pd.DataFrame(test_set)
test_df.columns = ["doc_id", "document", "label"]
test_df = test_df.astype({"doc_id": "int"})
test_df = test_df.sort_values(by="doc_id")
test_df = test_df.reset_index(drop = True)
```

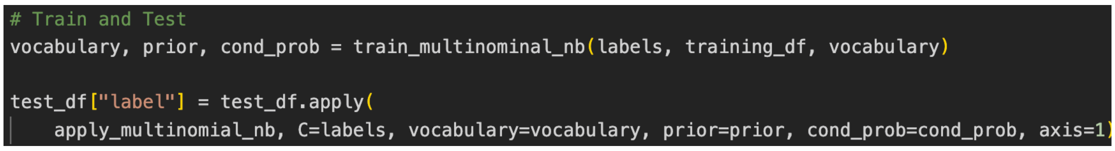
（3）利用先寫出的「前處理」、「計算 tf 數量」函數對對文件進行處理和更新。

（4）利用「計算 chi-square」的函數計算出各文章的 chi-square 並進行排序，再挑選出前 500 個 term



```
# Chi square test
def chi_square(labels, dataset):
    vocabulary = extract_vocabulary(dataset.tf)
    N = len(dataset)
    chi2 = dict()
    count = 0
    for term in vocabulary:
        chi2_term = 0
        matrix = dict()
        matrix["tp"] = dataset[dataset["tf"].apply(lambda x: term i
        matrix["ta"] = dataset[dataset["tf"].apply(lambda x: term n
        for lb in labels:
            matrix["cp"] = dataset[dataset["label"] == int(lb)]
            matrix["ca"] = dataset[dataset["label"] != int(lb)]
            matrix["tp_cp"] = len(matrix["tp"][matrix["tp"]["label"]
            matrix["tp_ca"] = len(matrix["tp"][matrix["tp"]["label"]
            matrix["ta_cp"] = len(matrix["ta"][matrix["ta"]["label"]
            matrix["ta_ca"] = len(matrix["ta"][matrix["ta"]["label"]
            chi2_class = 0
            for i in ["tp", "ta"]:
                for j in ["cp", "ca"]:
                    E = len(matrix[i]) * len(matrix[j]) / N
                    chi2_class += ((matrix[f"{i}_{j}"] - E)**2) / E
            chi2_term += chi2_class
        chi2[term] = chi2_term
        count += 1
        if count % 500 == 0:
            print(f"Finish: {count}/{len(vocabulary)}")
    # Use only 500 terms, chosen by ranking
    vocabulary = sorted(chi2, key=chi2.get, reverse=True)[:500]
    return vocabulary
```

（5）寫出 NB 分類器的模型，並進行 training 和 testing



```
# Train and Test
vocabulary, prior, cond_prob = train_multinominal_nb(labels, training_df, vocabulary)

test_df["label"] = test_df.apply(
    apply_multinomial_nb, C=labels, vocabulary=vocabulary, prior=prior, cond_prob=cond_prob, axis=1)
```

（5）將結果輸出至 output.csv