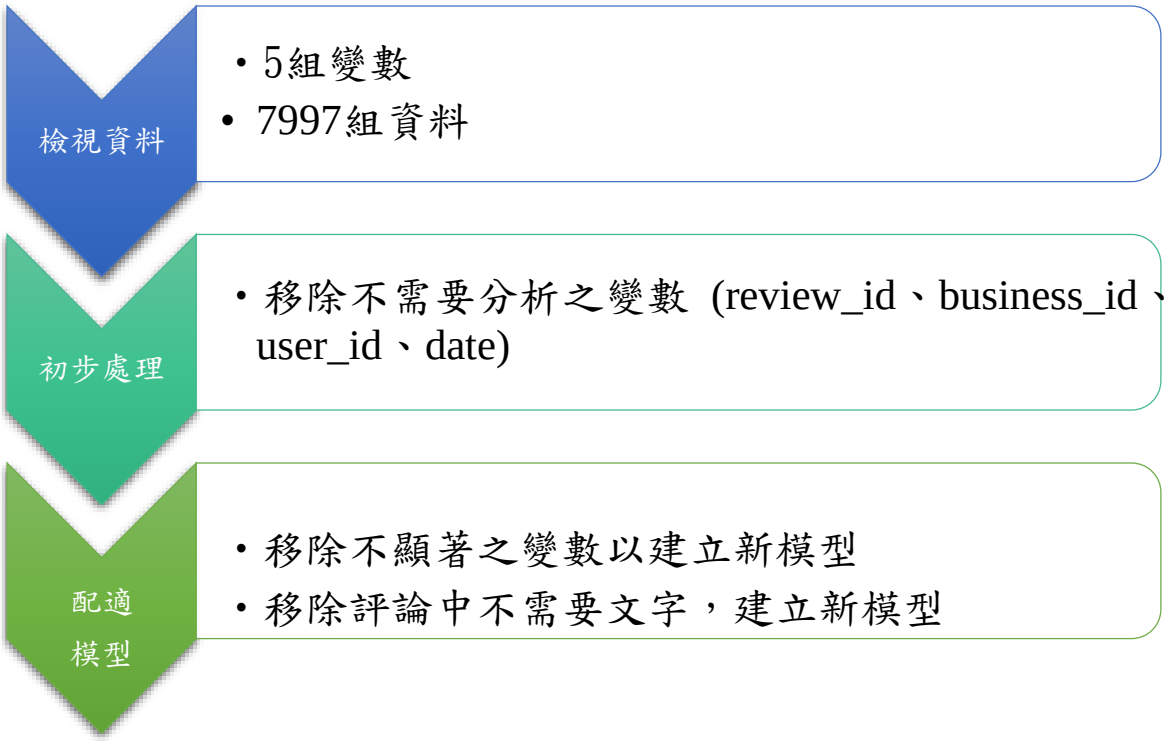
組別：Group11

組員：H24051273 梁雅婷、H24055015 黃瀞瑩、H24051304 陳冠頻

一、思考脈絡與分析

流程圖：



先「主觀性的」移除一些我們初步認為影響成分較低的因子如下：

- review_id
- business_id
- user_id
- date

以資料中「text」欄位中內的評論做為主要變數，以訓練模型

在訓練模型中，使用以下幾種方法:

- logistic regression
- KNeighborsClassifier

第一次嘗試:

因在選取時碰到了 dimension 不同而造成的錯誤，因此在首次嘗試時只取 train data 中前 2003 筆資料(和 test 資料相同大小)來進行訓練模型。<如圖一、二>

其所使用的模型為 logistic regression，所輸出的準確率為 0.4558、RMSE 為 1.3130<如圖三>

第二次嘗試:

在詢問過助教後，改善了因 dimension 不同所造成的程式上的錯誤，因此選取範圍為 train data 的全部資料，以 train data 所有資料來進行模型的訓練。<如圖四>

所使用的模型為 logistic regression，其輸出的準確率為 0.5242、RMSE 為 1.0781<如圖五>

由第一次和第二次嘗試後發現，train data 所選取的大小，對模型的準確率和 RMSE 有極大的影響，雖使用同樣的模型來進行資料訓練，但卻有顯著的不同。

第三次嘗試:

此次使用的模型為 KNeighborClassifier 來進行模型的訓練，所使用的資料和第二次嘗試時相同。<如圖六>

其輸出的準確率為 0.3120、RMSE 為 1.4103。<如圖七>

由第二次嘗試和第三次嘗試做比較，使用 logistic regression 可以得到較高的準確率以及較低的 RMSE，由此可知 logistic regression 比較適合來訓練此種資料。

第四次嘗試:

本次使用 stopwords，來刪除評論中多餘的字，同樣使用 logistic Regression 來進行模型訓練。<如圖八>

其輸出的準確率為 0.5142、RMSE 為 1.1010。<如圖九>

和第二次嘗試做比較，其準確率以及 RMSE 並沒有比第二次嘗試理想，其數值並沒有極大的差距。

二、輸出結果

〈圖一〉第一次嘗試-程式碼

```
import numpy as np
np.set_printoptions(precision=2)
r = tfidf.transform(x).toarray()[0:2003]
print(r)
```

〈圖二〉第一次嘗試-程式碼

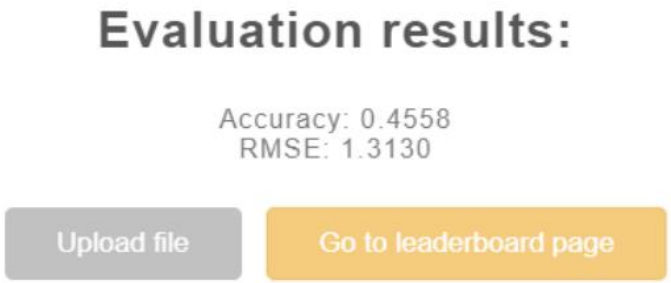
```
xbof = vt.transform(x2)
np.set_printoptions(precision=2)
r2 = tfidf.transform(x2).toarray()
print(len(r2))
```

```
2003
```

```
s = data['stars'][0:2003]
```

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression().fit(r,s)
A = list(pd.DataFrame(lr.predict(r2))[0])
```

〈圖三〉第一次嘗試-輸出結果

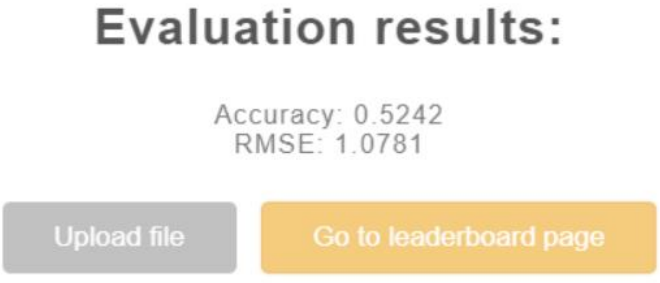


〈圖四〉第二次嘗試-程式碼

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression().fit(r,s)
A = list(pd.DataFrame(lr.predict(r2))[0])
```

```
idd = data2['review_id']
result = pd.DataFrame(A,index = idd)
z = result[0]
z.to_csv(r"C:\Python35\workspace\out.csv")
```

〈圖五〉第二次嘗試-輸出結果



〈圖六〉第三次嘗試-程式碼

```
from sklearn.neighbors import KNeighborsClassifier
clsf = KNeighborsClassifier()
ff = clsf.fit(r,s)
res = clsf.predict(r2)
```

```
idd = data2['review_id']
result = pd.DataFrame(res,index = idd)
z = result[0]
print(z)
z.to_csv(r"C:\Python35\workspace\out2.csv")
```

〈圖七〉第三次嘗試-輸出結果

〈圖八〉第四次嘗試-程式碼

```
import nltk
nltk.download('stopwords')
```

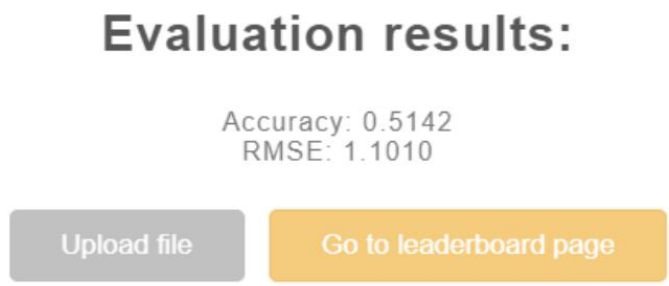
```
[nltk_data] Error loading stopwords: <urlopen error [WinError 10060] A
[nltk_data]     connection attempt failed because the connected party
[nltk_data]     did not properly respond after a period of time, or
[nltk_data]     established connection failed because connected host
[nltk_data]     has failed to respond>

False
```

```
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
xxx = []
stopWords = stopwords.words('english')
for i in range(len(xx)):
    words = word_tokenize(xx[i])
    xxx.append(words)

for i in range(len(xxx)):
    for j in range(len(stopWords)):
        if stopWords[j] in xxx[i]:
            xxx[i].remove(stopWords[j])
```

〈圖九〉第四次嘗試-輸出結果



三、結論

這次分別使用 KNeighborsClassifier 以及 LogisticRegression 兩種不同的訓練模型，由輸出的結果可以得知邏輯式迴歸的方式比較適合這次所要預測的資料，而在 stop words 的條件加進去之後，卻發生準確率降低以及 RMSE 變高的現象，因此可以猜測 stop words 當中有些字對於要預測的資料是重要的。