資料科學導論hw5\_DL組

(1)**小組各成員的姓名、系級與學號。**

賴榆方 統計114 H24104034、林孟漩 統計114 H24101159、林佑融 測量111 F64071122

(2) **競賽敘述與目標**：

用train.cvs裡的資料嘗試使tweet以單字為偵測的單位，分成三個有關仇恨程度的級別，利用這些資料去訓練模型，並用此模型去偵測test.cvs裡tweet的仇恨程度。

(3) **資料前處理**：

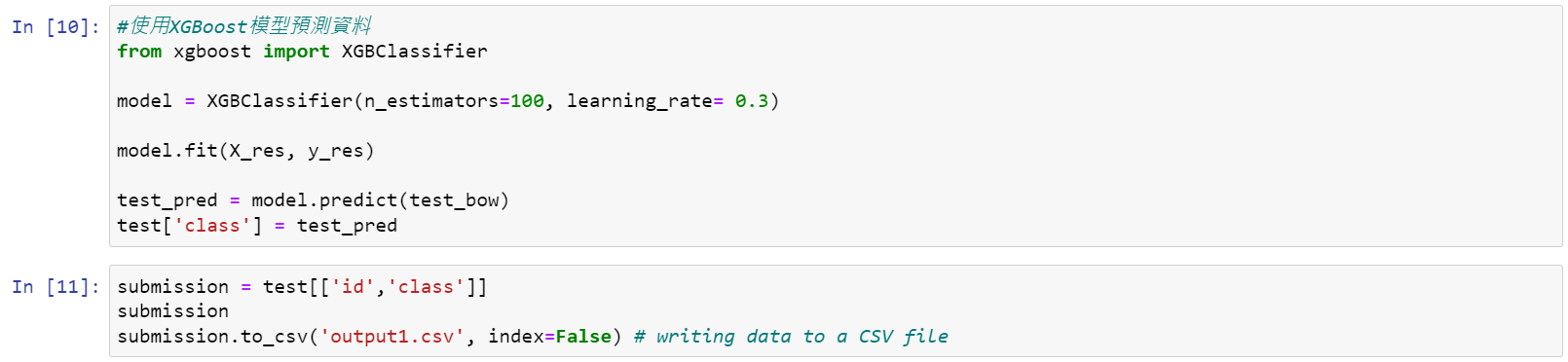
我們先將train.cvs和test.cvs的資料合併，以方便之後的資料處理，再去除資料中與我們想要資訊無關的部分，例如twitter handles,special characters, numbers, punctuationsc和Short Words等等，最後我們還利用欠採樣和過採樣兩種方式來解決資料不平衡的問題，一方面有放回地抽取少數類別的樣本，並將其複製後加入數據集當中，另一方面從多數類別的樣本中隨機刪除一部分的樣本，使三種仇恨程度的資料量盡量趨於平均(33.36%，33.31%和33.33%)以達到分類平衡。

(4) **特徵處理與分析**：

使用stemmer將不同的單字表示型態歸一化，降低文本的複雜度，並利用分詞器Tokenizer將這些單詞轉換成數字使這些資料能被後續的模型所接受，最後使用CountVectorizer計算特徵數值，將處理理好的資料轉換成詞頻矩陣，並透過fit\_transform函數去計算各個詞彙所出現的次數。

(5) **預測訓練模型**：

對於前處理完的資料，我們有嘗試過SVM、LogisticRegression、RandomForestClassifier、XGBoost等模型來進行訓練與預測，其中我們發現不論是使用哪個模型，在調參完後的預測結果，都不比原本預設的參數值佳，所以還是選擇不去調參數；另外，如果不搭配過採樣與欠採樣的話，我們的結果其實會是SVM的分數更高，但如果搭配採樣方法，則會是XGBoost的分數更優秀。而相互比較之下，XGBoost模型搭配採樣方法的結果略勝一籌，所以最終我們採用的模型及參數為：



(6) 預測結果分析：

1. XGBoost模型搭配過採樣(BorderlineSMOTE)與欠採樣(TomekLinks)方法
2. XGBoost模型只搭配過採樣(BorderlineSMOTE)方法
3. RandomForest模型只搭配過採樣(BorderlineSMOTE)方法
4. SVM模型只搭配過採樣(BorderlineSMOTE)方法

|  | HateFscore | AllFscore | Final |
| --- | --- | --- | --- |
| 1(最佳) | 0.7047 | 0.7004 | 0.7030 |
| 2 | 0.7035 | 0.6996 | 0.7019 |
| 3 | 0.6317 | 0.6542 | 0.6407 |
| 4 | 0.6155 | 0.6297 | 0.6212 |

1. 未做調整下，使用SVM模型預測
2. 未做調整下，使用多元LogisticRegression模型預測
3. 未做調整下，使用XGBoost模型預測

(嘗試過用RandomizedSearchCV調參，發現都是未調參數較佳)

|  | HateFscore | AllFscore | Final |
| --- | --- | --- | --- |
| 5 | 0.6750 | 0.6996 | 0.6848 |
| 6 | 0.6555 | 0.6921 | 0.6701 |
| 7 | 0.6320 | 0.6745 | 0.6490 |

1. 切割訓練資料集來測試，使用SVM模型預測
2. 切割訓練資料集來測試，使用多元LogisticRegression模型預測
3. 切割訓練資料集來測試，使用RandomForestClassifier模型預測
4. 切割訓練資料集來測試，使用XGBoost模型預測

(看起來這麼做結果沒有比較好，猜測可能是因為訓練的資料變少了)

|  | HateFscore | AllFscore | Final |
| --- | --- | --- | --- |
| 8 | 0.6307 | 0.6668 | 0.6451 |
| 9 | 0.6272 | 0.6691 | 0.6440 |
| 10 | 0.6118 | 0.6534 | 0.6285 |
| 11 | 0.6221 | 0.6672 | 0.6401 |

根據上傳到網站的預測結果，我們覺得在面臨資料不平衡的狀況下，可以搭配過採樣或欠採樣的方式來解決這方面問題，能夠使得準確率提高。而在不搭配過採樣與欠採樣方法的話，我們的結果其實會是SVM的分數更高；但如果有搭配採樣方法，則會是XGBoost的分數更佳。我們最終上傳的結果所使用的模型為XGBoost與其預設參數：

model = XGBClassifier(n\_estimators=100, learning\_rate= 0.3)

(7) 感想與心得：

賴榆方：

這一次的競賽是有關仇恨言論的預測，需要對文字型資料進行大量前處理，才能將處理過後的資料套進模型中，這樣也使得這次作業的難度比上次作業更上一層。

在一開始，我們也遭遇過很多挫折，尤其是在處理文字資料這方面，因為在以往沒有這方面的練習，所以花了我們很多時間去找資料與理解，如果最初的這步沒有完成，剩下的步驟也就沒有辦法進行下去，但幸好的是，使用模型這部分在之前的作業已經有實作過，幫我們節省了不少時間。

說實話，我覺得這是一堂很充實的課程，雖然確實滿累的，但也學到很多知識，對於資料科學也有了更深入的了解。

林孟漩:

做競賽的過程只能以坎坷來形容，第一次小組討論的時候花了三個小時以失敗告終，只獲得了一個怎麼debug都跑不出來的廢稿，第二次小組討論同樣以失敗收場，我們參考的代碼算是非常接近我們想要的型態，資料也想好要怎麼處理，但偏偏那個代碼要用到tensorflow這個套件，結果無論是anaconda或者是直接在cmd都無法下載(我懷疑是當時網路太慢，因為到家我就下載好了)

在家寫新的程式時還發生train和test處理到最後features不相符的窘境，幸好最後參考了同組組員的做法在處理資料前先把兩者資料合併，最後終於跑出結果了(但精確值還是很低…)。

林佑融：

這次的作業五比起上次作業四單純分類，還要先針對文字前處理，在除掉許多不重要的字元像是後，減少記憶體使用量。之後進行nltk套件的使用像是PorterStemmer。第一次接觸處理文字資料真的是手忙腳亂，即便找到了非常接近本次作業五的kaggle文字處理教學，也嘗試了好多方法，還是經常出現謎之bug，包括教學中的套件版本過舊，以至於在執行時找不到函式。這也了解到，做資料科學最麻煩的永遠是資料前處理，只要資料前處理妥當，之後模型的結果才會符合期待。此外，過程中有發現大多數的仇恨字眼詞性應該是名詞或是形容詞，可惜沒有進一步使用到nltk中的POS進行判斷。這次的作業五內容真的比前幾次硬很多！