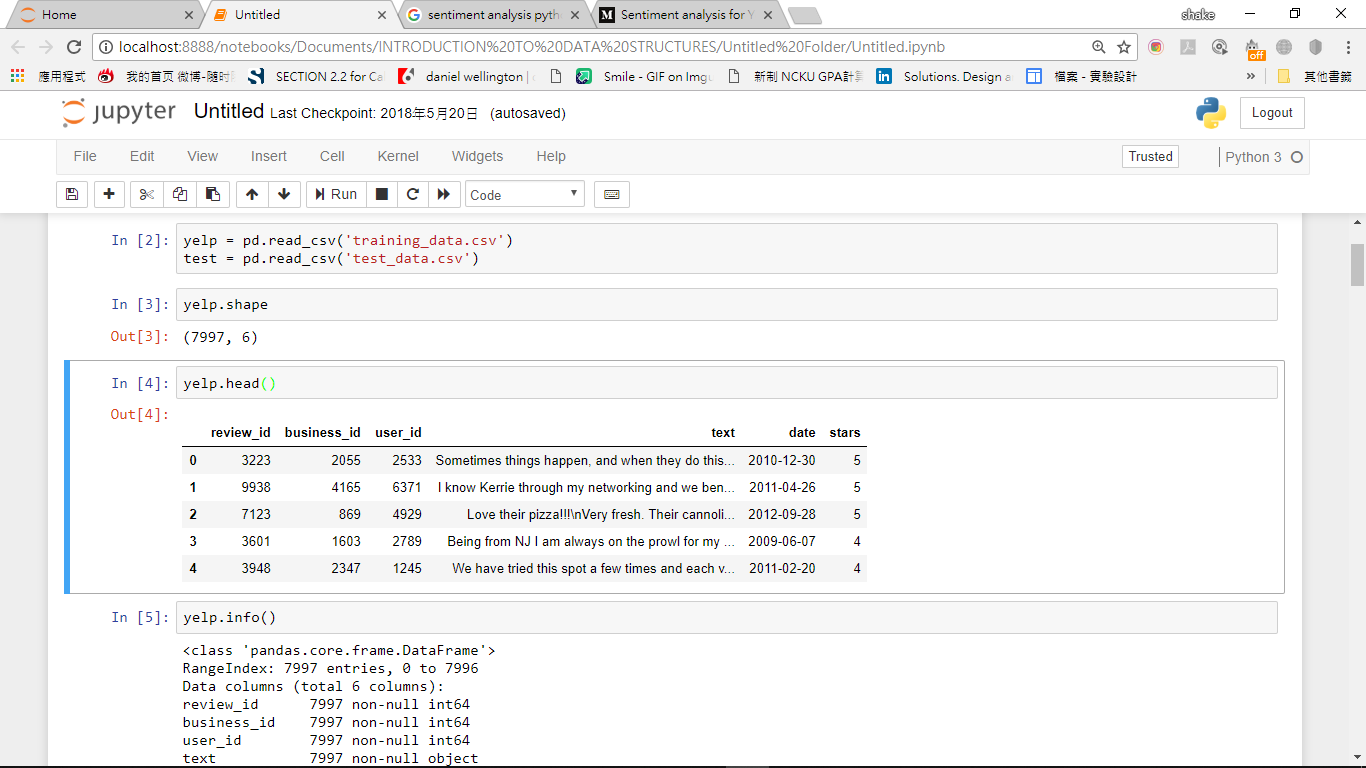
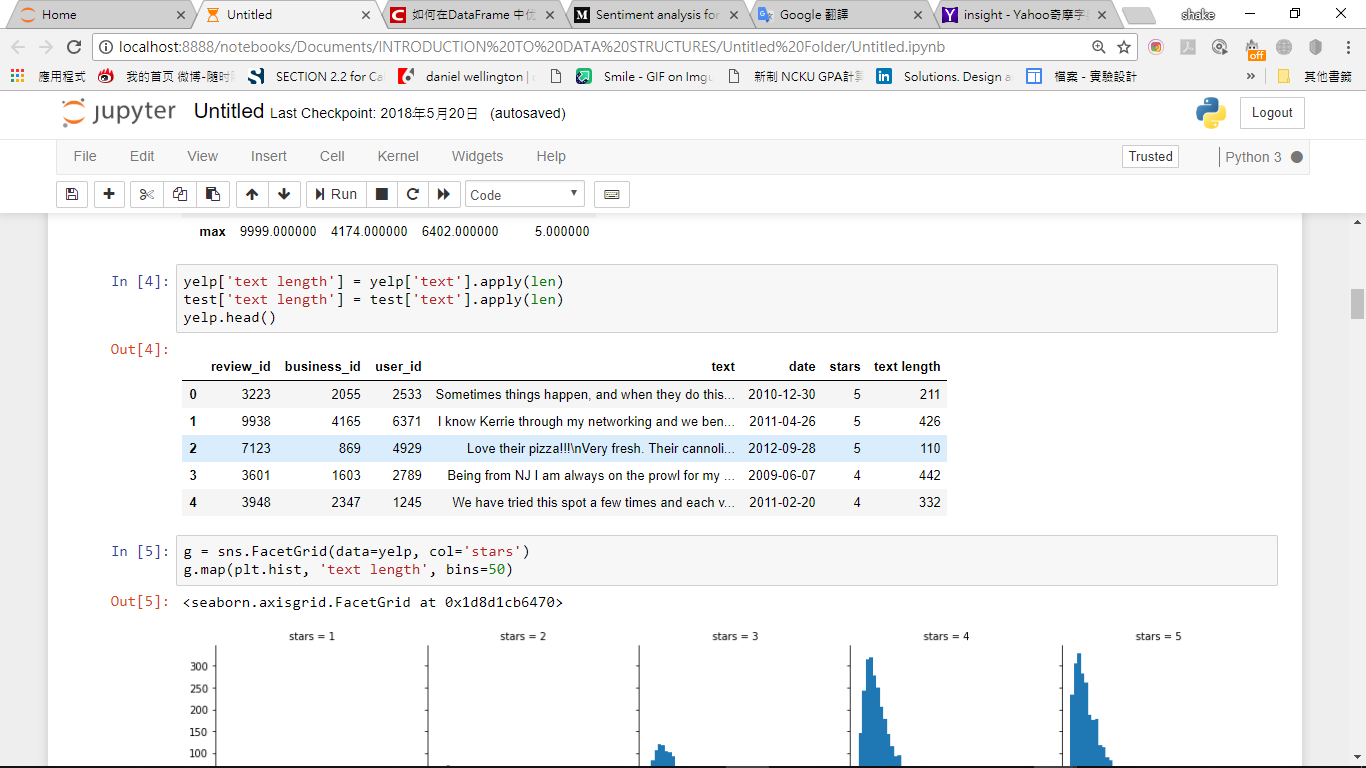
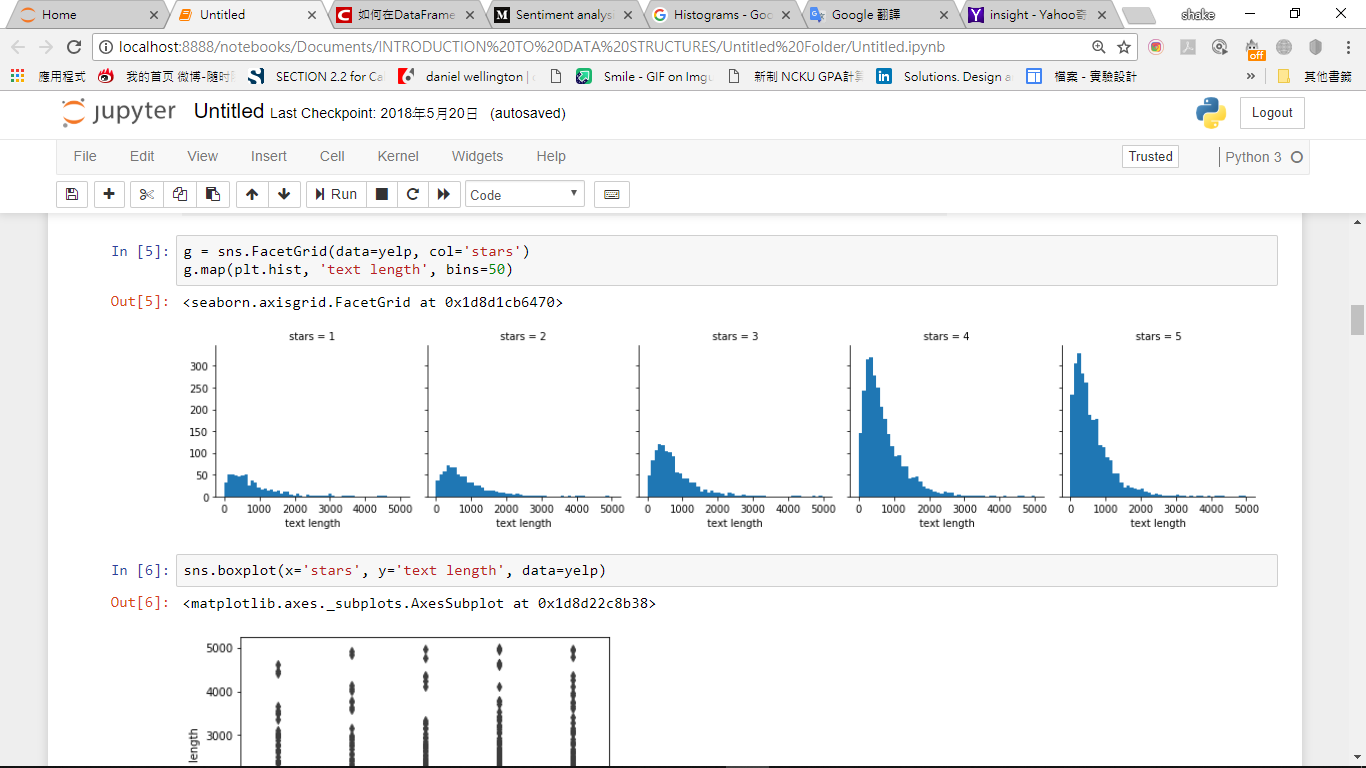
資料概述

全部資料有10000筆的評論，包含了training data的7997筆和test data的2003筆，每筆資料都含有以下資訊。

1. review\_id (ID for the posted review)
2. business\_id (ID of the business being reviewed)
3. user\_id (User’s id)
4. text (Review text)
5. date (Day the review was posted)
6. stars (1–5 rating for the business)

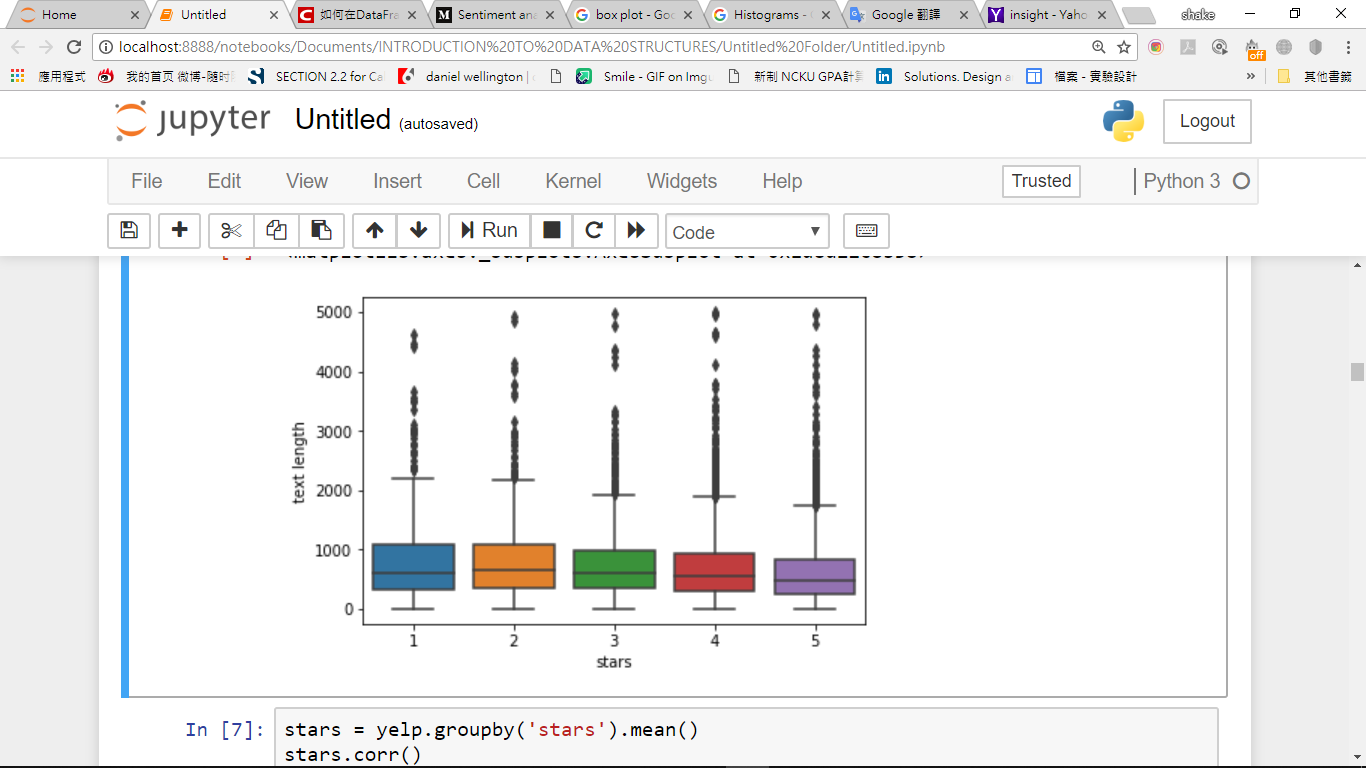
檢視training data的型態:

為了更加了解text故我們多加入了一行text length儲存每個評論的字數，如下圖所示:

接著我們透過圖來進一步了解資料資訊，首先利用直方圖來觀察text length分別在五個stars裡的分布狀況，如下圖所示:

可以觀察text length的分布情形在五個星等中大致類似，但是明顯地可以看出在四星和五星中的評論數量偏多。

接著我們畫出text length和星等的箱型圖，如下所示:

.

可以看出一星和二星的text length較其他三個星等略長，不過也有許多outliers，所以觀察完圖形後可以發現text length對於預測星等來說可能不是一個很好的特徵。

資料預處理

因為一些字詞對於分析上無實質幫助，故我們必須先將之做文章清理的動作，例如停頓詞（Stop Words）和標點符號，結果如下所示:

**範例:**

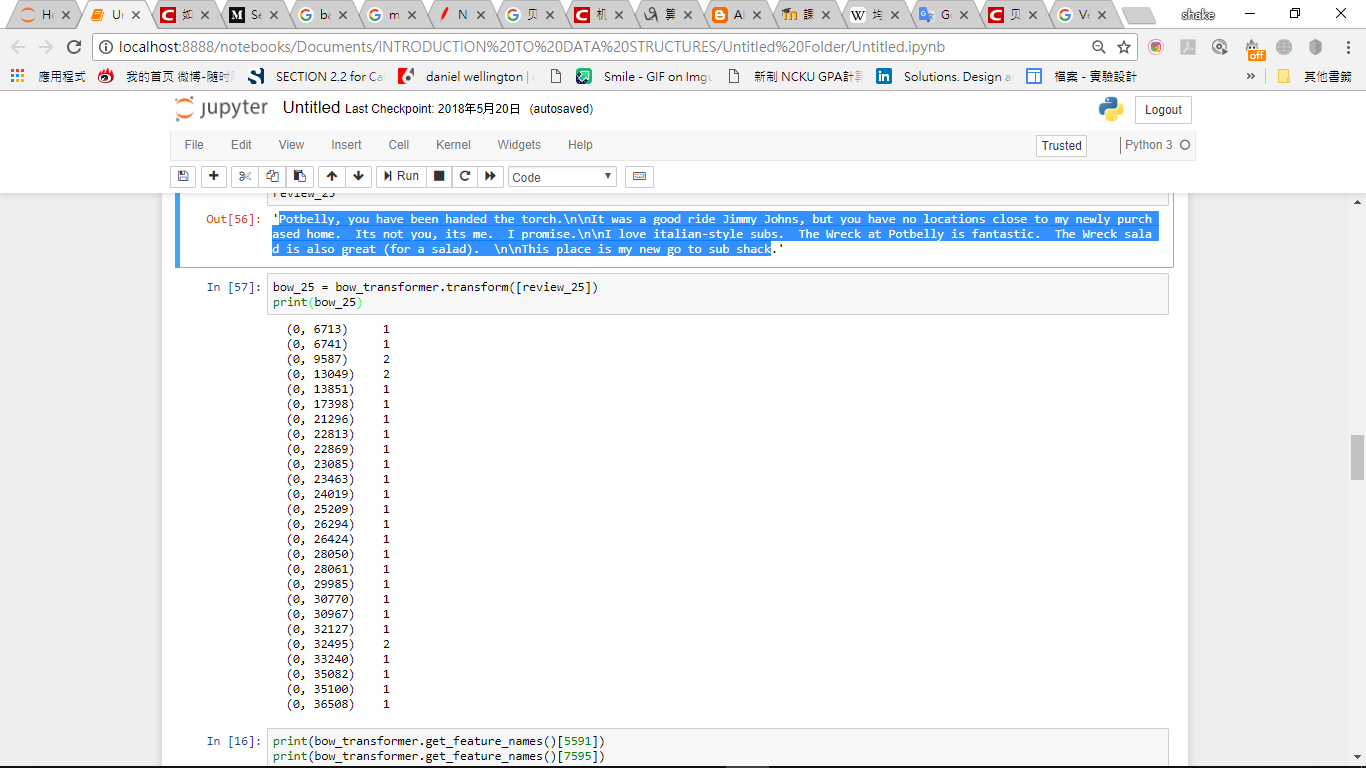
Sometimes things happen, and when they do this is the place where you want them taken care of. The orthopedics department and trauma department have some of the best, and nicest Doctors you'll find in the state.

**清理後結果:**

'Sometimes', 'things', 'happen', 'place', 'want', 'taken', 'care', 'orthopedics', 'department', 'trauma', 'department', 'best', 'nicest', 'Doctors', 'youll', 'find', 'state'

可以看出清理後的結果都只剩下重要的單詞，冗言贅字與標點符號皆以刪除，如此一來便有利後續分析。

向量化

 我們先將train data所有的text做文章清理後，將所有unique的單字丟進詞袋(Bag-of-words)裡，發現總共有39002個單詞，接著以train data的第25則評論為例，可以轉換成以下結果:

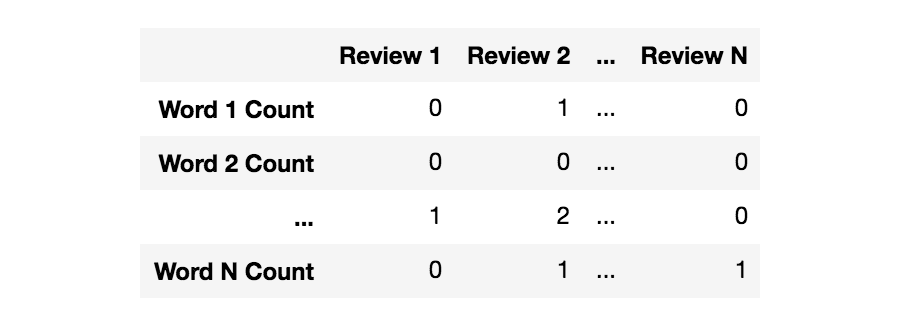
Potbelly, you have been handed the torch. It was a good ride Jimmy Johns, but you have no locations close to my newly purchased home. Its not you, its me. I promise. I love italian-style subs. The Wreck at Potbelly is fantastic. The Wreck salad is also great (for a salad). This place is my new go to sub shack.

左邊的圖代表第25則評論裡有26個唯一的字元，有三個單字出現了兩次，其餘皆只出現一次。

分別是 Potbelly

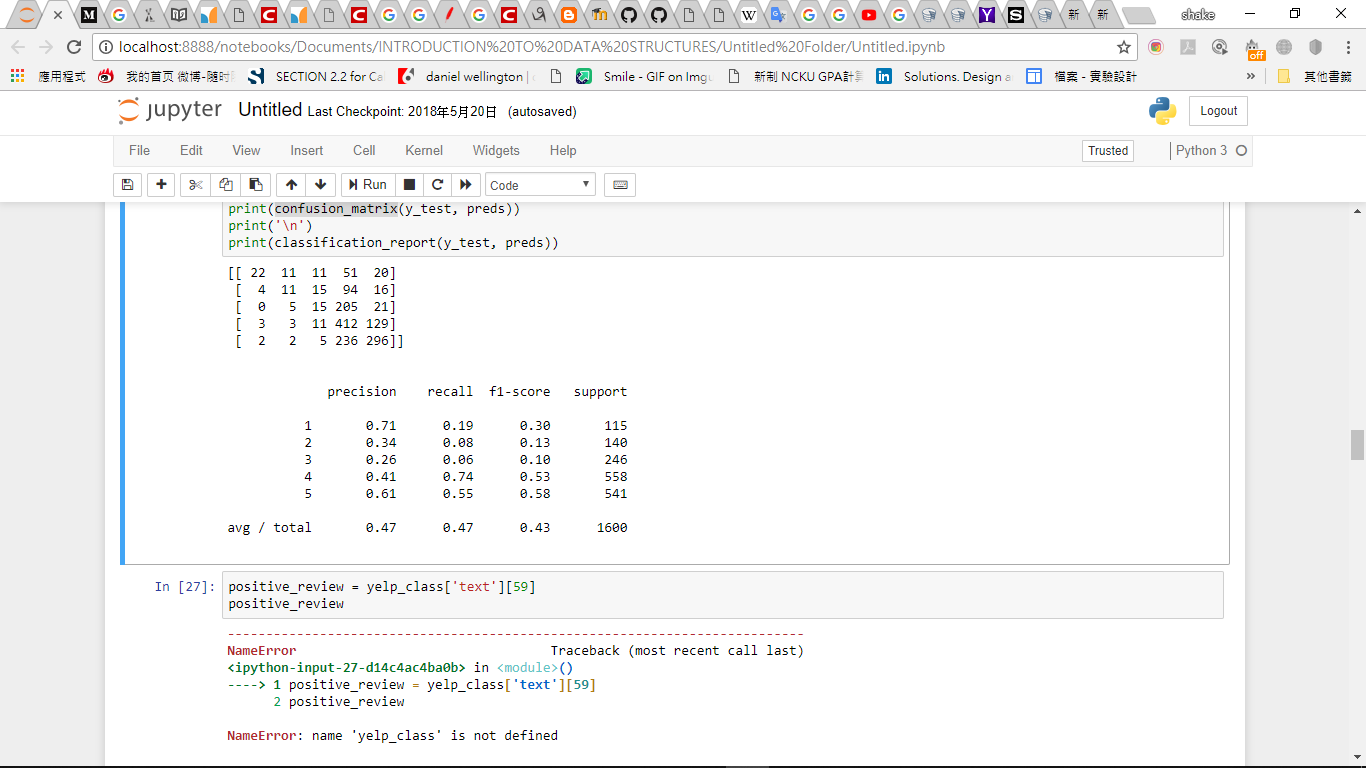
Wreck

salad

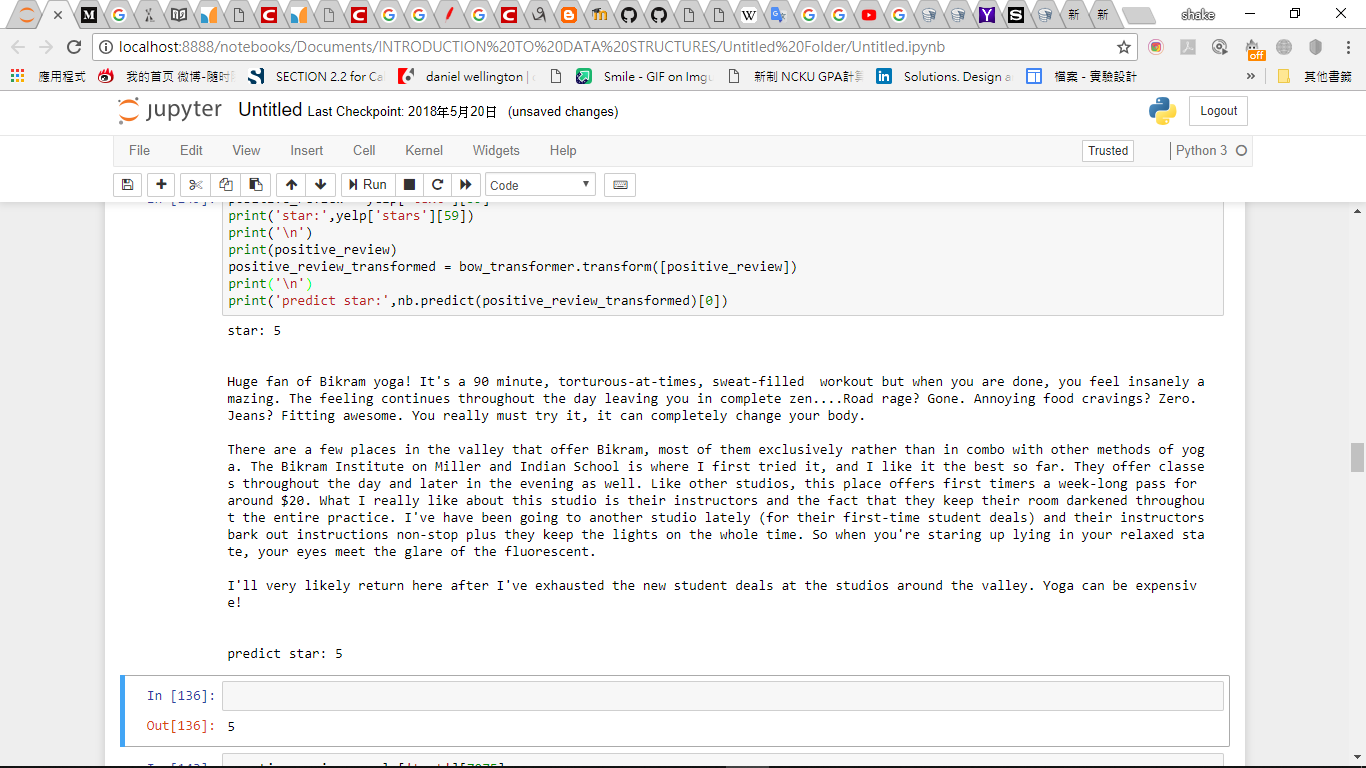
接著我們就可以將全部的train data轉成類似下圖的形式:

Multinomial Naive Bayes

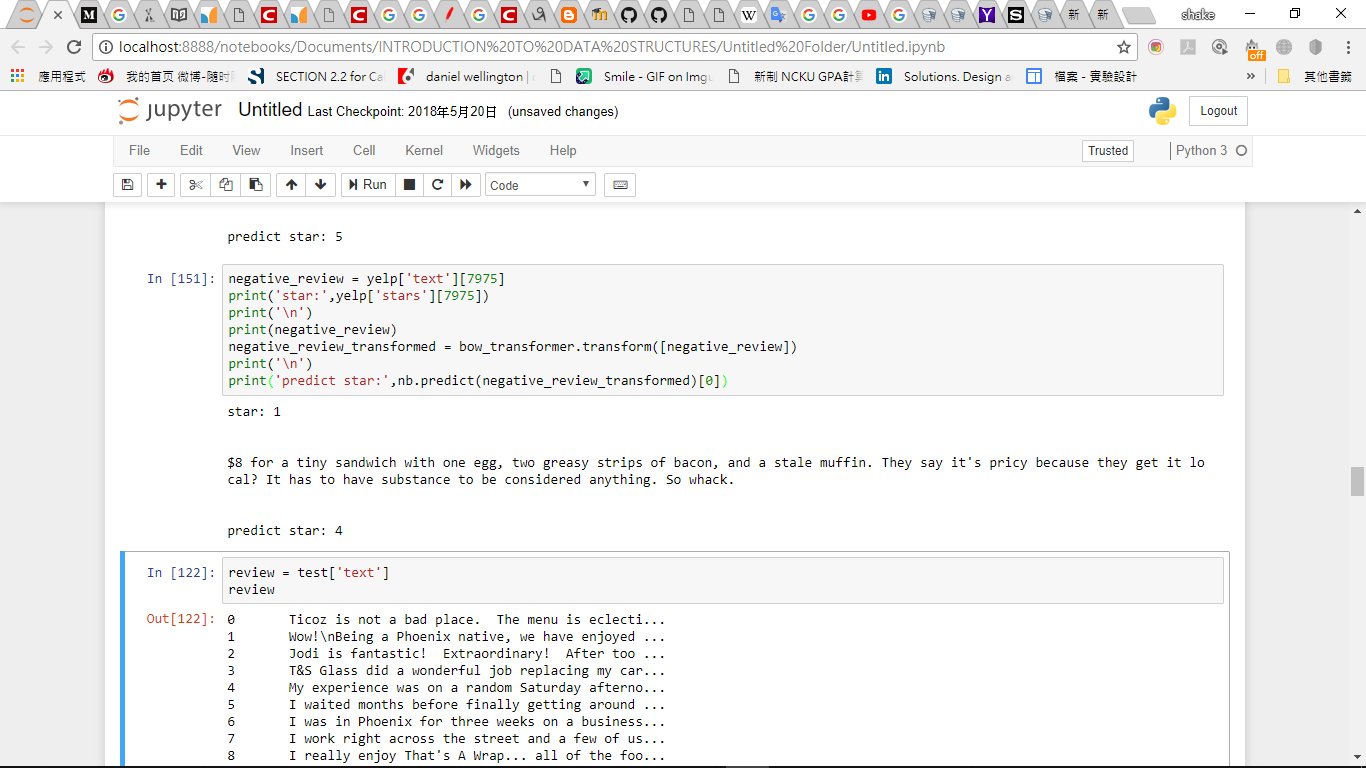
因為上一步在做向量化時，我們將資料去除停頓詞（Stop Words）和標點符號後轉換成詞頻的形式，故在這邊我們選擇使用sklearn裡的Multinomial Naive Bayes模型去做分類，其中我們將資料再分割成80%的訓練集和20%的測試集，接著我們用測試集測試MODEL做出混淆矩陣和分類報告，如下圖所示。



從混淆矩陣可以看出除了第一星等外，我們的模型很容易將資料分到上下的兩個星等，例如第四星等便有20%分到第三星等23%分到第五星等，所以從分類報告可看出大約只有47%的正確率，我們猜測可能是資料量太少，或是使用者的評論雖然是好的，但給的星數不一定如同評論預期的那麼高所致。

接著我們用train data實際測試一下我們的模型，先使用五星的正評當例子，如下圖所示:

可看出我們的模型預測正確，那麼接下來我們來測試看看1星的評論

可看出我們的模型將一星的分類成了四星，至於為什麼會如此，我們猜測可能是資料多偏向四星和五星，故訓練出來的模型也會更傾向將資料分類至四星及五星。

Github

https://github.com/shakewang11/CP2.git

Reference

1.https://medium.com/tensorist/classifying-yelp-reviews-using-nltk-and-scikit-learn-c58e71e962d9