Rating Prediction with User Business Review

目錄

1. 預測方法

這次的競賽主要是用「商業評論」來預測對應到「星等數」，這是一種屬於回歸的預測，而這次我們有嘗試了不同的模型，主要是以 Regression (回歸)的資料處理種類，而上網查了許多模型後，我們選擇拿來預測的模型有 Logistic Regression(LR)、Random Forest Classifier(RF)、Stochastic Gradient Descent(SGD)、Super Vector Machines(SVM)等等，接下來我們會分別去說明這四種模型的原理使用還有個別的優缺點。

Logistic Regression(LR)的優點是實現較簡單，常廣泛的運應在工業問題，而且可以結合 L2 正則化解決多重共線性的問題，在分類時計算量偏小，速度很快，存儲資源較低，缺點是無法處理大量多樣的特徵或者變量，容易欠缺擬合，所以導致一般準確度會較低，而當特徵空間很大時，邏輯回歸的性能也無法發揮得好，只能處理兩分類的問題且必須線性可分。

Random Forest Classifier(RF) Random 代表是隨機抽取，Forest 就是說這裡不止一棵树，而由一群決策樹所组成的一片森林，連起来就是利用隨機抽取的方法訓練出一群決策樹來完成分類的任務。優點是能夠處理較高維度的數據，並不用做特徵選擇，對於不平衡的數據集來說，可以平衡誤差，缺點是在某些特徵值影響較大時在處理回歸問題會過擬。

Stochastic Gradient Descent(SGD)隨機梯度下降算法每次只隨機選擇一個樣本來更新模型參數，因此每次的學習是非常快速的，並且可以進行在線更新。此模型經常被用在文本分類及自然語言處理，假如我們的資料是稀疏的，不太容易辨別，但此模型的分類器可以輕鬆的解決這樣的問題，尤其是對於大數據量訓練集（ >10000）的回歸問題很合適。優點是在處理數據量較大的情況下，以上兩種模型要花的時間非常的久，但此模型可以讓時間複雜度大幅減少（每次只隨機看一個點），最後花很短的時候就有可以有預測結果，缺點是每次更

新時可能不會按照正確的方向走，但帶來的優化波動，可以從很多的局部極小值的點，找到更優化的方向，跳到另外一個更好的局部極小值點，最終可以收斂到全局極小值的點。

Super Vector Machines(SVM)是種分類算法，基本模型定義為特徵空間上的間隔最大的線性分類器，及支持向量機的學習策略便是間隔最大化，最終可轉化為一個二次規劃問題的求解。優點是可用於非線性分類，低泛化的誤差，計算的複雜度較低，可以解決較高維度的問題，但缺點在於對參數和函數的選擇較為敏感，原始的 SVM 只比較擅長處理二分類的問題。

我們這次預測的資料給的是要從文字方面來分析，所以處理過後的資料量較為龐大，特徵值也非常多樣，而我們主要看的不是預測的準確率有多高，而是想讓我們預測的精準度提高。雖然 Logistic Regression 在處理多重共線性的時候可以得到很好的預測，但因為我們的特徵值太多，而導致此模型在預測時無法讓精準度提高，所以不太適合運用在此次的預測，再來是 Random Forest Classifier 是多棵決策樹，雖然可以處理較高維度的資料，但是在特徵值過大時運用回歸的方式會使得答案過擬，也沒辦法得到好的精準度，而 Stochastic Gradient Descent 很適合處理文本分析的預測，剛好這次也給了很多文字要我們分析，而在特徵值過大，其他演算法跑較久的情況下，可以利用較短的時間得到更好的結果，我們只要找到較好的參數，就可以得到很好的精準度，而 Super Vector Machines(SVM)有較低泛化的誤差，較適合用於高維度的問題，我們覺得這兩種模型都各有利弊，所以我們會以這兩種模型來做我們這次的預測。

接下來的程式將針對測試資料內的文本資料進行處理，也針對模型內的參數做調整，找出我們認為最好的參數組合。

```
18  id_train=(df.iloc[:,:3]-df.iloc[:,:3].min())/(df.iloc[:,:3].max()-df.iloc[:,:3].min())
19  id_test=(df_test.iloc[:,:3]-df_test.iloc[:,:3].min())/(df_test.iloc[:,:3].max()-df_test.iloc[:,:3].min())
```

因為編碼過後的文字都在 0-1 之間，假如前面的數值 ID 過大會影響到預測，所以將他們每個數值都變為 0-1 之間的數值。

```
22  vectorizer=TfidfVectorizer(max_df=0.8,min_df=3).fit(text)
```

使用 TfidfVectorizer 函數，並將 max_df 參數調整為 0.8，在文檔中出現的關鍵詞超過這一比例的刪除，因為出現次數過於平凡，而 min_df 調為 3，在文檔之中低於這一數量的關鍵詞刪除掉，因為次數太少過於獨特，所以對我們的預測沒有幫助，原本我們的資料量有兩萬多筆，可以藉由我們調整的參數，把次數過多和過少的刪掉，能夠讓我們的資料量降為一萬兩千多筆。

```
23  text=vectorizer.transform(text)
24  fea=vectorizer.transform(new_text)
```

將訓練資料與測試資料的每一句文字，做編碼處理，依照他們出現的頻率與在文本的次數，會有相對應的數值頻率

```
27  textid=pd.DataFrame(id_train)
28  textname=pd.DataFrame(text.toarray())
29  text=pd.concat([textid,textname],axis=1)
30  text=text.values
31  newid,newname=pd.DataFrame(id_test),pd.DataFrame(fea.toarray())
32  fea=pd.concat([newid,newname],axis=1)
33  fea=fea.values
```

將上述我們轉換過後的數值與 ID，加入 Dataframe 中。並在最後轉為 array 的形式，因為 array 在做預測的速度會比 dataframe 還要來得迅速許多。

```
36  param_grid={"C":[0.1,1,10],"gamma":[0.1,1,10]}
37  cv=KFold(shuffle=True)
38  grid=GridSearchCV(SVR(),param_grid=param_grid,cv=cv,verbose=3)
```

```
41    grid.fit(text,y)
42    pred=grid.predict(fea)
```

以先看出 SVR 模型中，所包含的可調整參數有哪些，我們針對了 C 和 gamma 這些以數字為主的參數，以隨機設定數字的方式來做調整，而 KFold 是對交叉驗證做洗牌的處理，verbose=3 是將資料分割為三群並做交叉驗證，進而選出最好的參數模型。驗證完之後的模型就可直接 fit 測試資料和預測資料囉。

2. 預測分析

起初在調整對文字處理時，假如都不改 CountVectorizer 和 TfidfVectorizer 的參數時，編碼完的文字約為兩萬六千個字，對一個資料來說，這實在是太多了，因此藉由上網搜尋，知道了可調整 max_df 和 min_df 將文字的行數控制在一萬初，這個大幅度的減少，不僅可將一些多餘的雜字刪除掉，也能在將來的模型預測中，可大幅提高速度，方便我們學習。

第一個模型使用的是 Logistic Regression，使用 CountVectorizer 來做文字處理，對於測試資料的 RMSE 為 1.03 左右，但在 TfidfVectorizer 就表現得更不理想了，然而藉由上次競賽中隨機森林模型表現的還不錯，因此在此次競賽我們也嘗試使用來做預測，但表現出來的 RMSE 為 1.3 左右，比起第一個表現不佳，再調整完參數之後，進步的幅度也不大，因此藉由網路上的搜尋以及同學們的討論，知道了隨機梯度下降 SGD 模型，在未調整之前，直接將資料帶入模型，就讓我們大吃一驚，竟然能將 RMSE 降為 0.96，並且模型運行速度也比其他快速許多，並藉由調整參數後，約略下降，但比起其他同學還是略遜許多，因此嘗試使用 SVM 的 package SVR，但在一開始使用 SVR 時，資料量太大，筆電運行的速度不夠，因此曾使用過 PCA 讓資料降到 1000 維度，的確讓 SVR 運行速度加快，但表現出來的 RMSE 不佳，讓我大失所望，但在有機會使用運行較快的桌電

時，決定使用 10000 維度的資料去跑 SVR，並且使用 CountVectorizer 和 TfidfVectorizer 嘗試不同的結果，也使用 GridSearchCV 調整 C 和 gamma 的參數，最終得到 0.87 的結果。

總結而言，我認為我們的分析有幾項可以改善的地方，首先是有些模型適合 CountVectorizer，有些反而適合 TfidfVectorizer ，這是我們還必須要去了解的。因為訓練的資料是實際的資料，所以可能有一些重要的文字特徵。;另外是模型的部分，我們用的大多數都是直接下去測試，但還是必須要去了解哪些資料適合使用什麼模型。

3. 參考文獻

（說明 Stochastic Gradient Descent 模型）
https://blog.slinuxer.com/2016/09/sgd-comparison
（說明 Decision Tree 模型）
https://medium.com/@yehjames/資料分析-機器學習-第 3-5 講-決策樹-decision-tree-以及隨機森林-random-forest-介紹-7079b0ddfbda
（說明 Super Vector Machines 模型）
https://blog.csdn.net/qq_16633405/article/details/70243030

4. Github 的連結(URL)

https://github.com/h24054043/Rating-Prediction-with-User-Business-Review