Compitition2 Report

Description

1.[分析方法]

→LogisticRegression, svm.SVC, LinearSVC, svm.LinearSVR 做結合。

2.[code]

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn import svm

#讀檔
df_train = pd.read_csv('training_data.csv')
df_test = pd.read_csv('test_data.csv')

text_train=df_train['text']

Vectorizer=TfidfVectorizer()
Vectorizer.fit(text_train) #用 TfidfVectorizer 算出每個字的重要性
X_train=Vectorizer.transform(text_train) #把資料變成 document-term matrix

y_train=df_train['stars']
text_test=df_test['text']
```

```
X_test=Vectorizer.transform(text_test).toarray() #資料變成 document-term matrix 的陣列

y_id= df_test['review_id'].values #取出 df_test['review_id']的數據

#使用 LogisticRegression 最好的參數做預測
classifier2=LogisticRegression(penalty='l1',C=2.0,intercept_scaling=0.25,warm_start=True)
classifier2.fit(X_train,y_train)
y_test2=classifier2.predict(X_test)

#使用 svm.SVC 最好的參數做預測
classifier5=svm.SVC(kernel='linear',C=1.15,probability=True)
classifier5.fit(X_train,y_train)
y_test5=classifier5.predict(X_test)

#使用 svm.LinearSVC 最好的參數做預測
classifier7=svm.LinearSVC(loss='hinge')
classifier7.fit(X_train,y_train)
y_test7=classifier7.predict(X_test)

#使用 svm.LinearSVR 最好的參數做預測
classifier8=svm.LinearSVR(C=0.95,loss='squared_epsilon_insensitive',epsilon=0.2,dual=False,fit_intercept=True,intercept_scaling=2)
classifier8.fit(X_train,y_train)
y_test8=classifier8.predict(X_test)

for i in range(len(y_test8)):
        if y_test8[i]>5:
```

```
            y_test8[i]=5
        if y_test8[i]<1:
            y_test8[i]=1

#結合多個模型做預測
pred=[]
for i in range(df_test.shape[0]):
    if y_test2[i]==y_test5[i]==y_test7[i]: #若 LogisticRegression、svm.SVC、svm.LinearSVC 三者的預測值相等
        pred.append(y_test2[i]) #則採取此預測值
    else:
        pred.append(y_test8[i]) #若其中一不成立 則採用 svm.LinearSVR 的預測值

#用 4 個模型:LogisticRegression,svm.SVC,LinearSVC,svm.LinearSVR  Rmse:0.8661 accurancy:0.5352
result= pd.DataFrame(np.column_stack((y_id.tolist(),pred)))
result.to_csv('result2.csv',index=False)
```

Analysis

★Analysis Suite

1.K-Neighbors Regression

2. KNeighborsClassifier

3. LogisticRegression

4. RandomForestClassifier

5. LinearRegression

6. RandomForestClassifier

7.TfidfVectorizer

8.GridSearchCV

★methods we have tried

一、單一 model

1.使用 KNeighborsClassifier fit “text”、`user_id”、`review_id”、`user_id”不論分開做或合在一起，最低的 Rmse 還是高達 1.19。

2.去除一些 stopwords 再次用 KNeighborsClassifier fit Rmse 稍微降至 1.18。

3.使用 LinearSVR，若預測值

小於 1:使其預測值等於 1

大於 5:使其預測值等於 5

藉此方法將數據合理化

→Rmse:0.8561、Accuracy:0.4473

4.使用 SVR，若預測值

小於 1:使其預測值等於 1

大於 5:使其預測值等於 5

藉此方法將數據合理化

→Rmse:0.8740、Accuracy:0.4703

二、多個 model

▲取預測值為類別資料的做相等，機率較高；否則，若預測值是連續資料，等式成立機率很低，便失去意義。

▲若結合等式不成立時，則取回歸做出的預測值，因為比起類別，連續資料可以盡可能降低誤差。

1.【 SVC、 Logistic Regression 、SVR】

如果 SVC =Logistic Regression(預測值)，便取此預測值

如果不成立，便取 SVR 的預測值。

→Rmse:0.8872 、Accuracy:0.5467

2.【 SVC、 Logistic Regression 、linear SVR】

如果 SVC =Logistic Regression(預測值)，便取此預測值

如果不成立，便取 linear SVR 的預測值。

→Rmse:0.8832 、Accuracy:0.5402

3.【結合 linear SVC、 SVC、linear SVR】

如果 linear SVC =SVC(預測值)，便取此預測值

如果不成立，便取 linear SVR 的預測值。

→Rmse:0.8852 、Accuracy:0.5352

4.【結合 Logistic Resgression、 SVM、linear SVR】

如果 Logistic Resgression =SVM(預測值)，便取此預測值

如果不成立，便取 linear SVR 的預測值。

→Rmse:0.9039 、Accuracy:0.5377

5.【結合 linear SVC、 SVC、 Logistic Regression 、LinearSVR】

如果 linear SVC =SVC =Logistic Regression(預測值)，便取此預測值

如果不成立，便取 LinearSVR 的預測值。

→Rmse:0.8661 、Accuracy:0.5352

6.**【結合 linear SVC、 SVC、 Logistic Regression 、SVR】**

如果 linear SVC =SVC =Logistic Regression(預測值)，便取此預測值

如果不成立，便取 SVR 的預測值。

→Rmse:0.8718 、Accuracy:0.5412

→【結論】單一模板最好的成績是使用 LinearSVR(Rmse:0.8561、Accuracy:0.4473)，Rmse 是所有方法最低，但 Accuracy 的表現並不好；而多個模板最好的成績是結合 linear SVC、 SVC、 Logistic Regression 、LinearSVR(Rmse:0.8661 、Accuracy:0.5352)，雖然 Rmse 非最低，但綜合來說是最好的。

三、選字及參數處理

1.用 TfidfVectorizer 檢驗字的重要性。

2.用 GridSearchCV 檢驗參數，找到最好的參數值。

→每一個 model 我們都用 GridSearchCV 自動幫我們選出最好的參數，再接下去做處理。

★problems we met

1.用 GridSearchCV 檢驗參數，因為數據太大，只能用片段資料做檢驗，以避免電腦無法負荷。

2.記憶體空間不足

3.一開始有試著處理 stopword，但後來發現與未處理的資料相較之下，rmse 及準確率不降反升。

4.我們只做預測值為類別資料的方法，若預測值是連續資料，該怎麼 combine model?

5.無法分辨評論較好及評論較差的商家；無法分辨習慣給高分及習慣給低分的使用者。

★the strong and weak points of our methods

【advantage】

1.設定必須為三個模板所預測出的值皆相同才採用此預測值，可多層判斷此預測值是否準確。

2.用回歸模板作為最後選擇可以降低誤差。

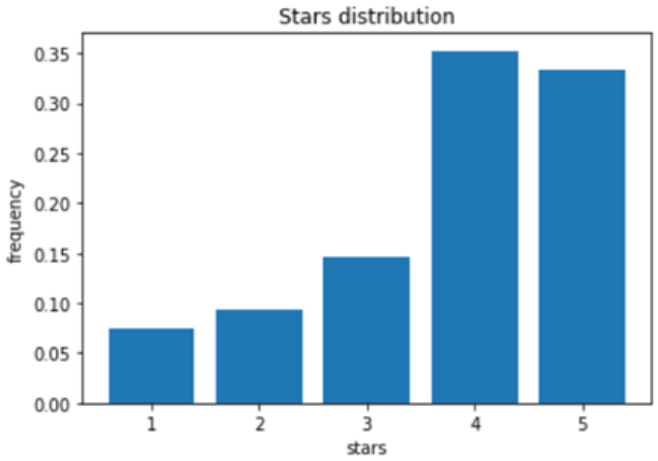
【disadvantage】

1.如果三個模板的結合都設相等，會忽略等式關係為一組相等兩組不等。

2.沒有考慮時間因素的影響。

Visualization

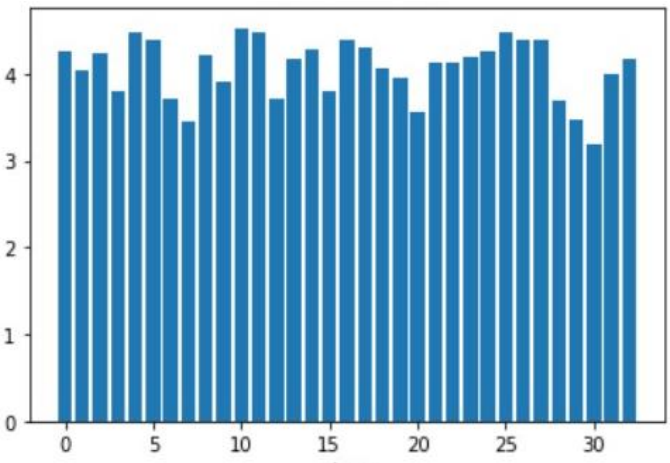
1.Stars distribution




```
count    7997.000000
mean        3.777667
std         1.214629
min         1.000000
25%         3.000000
50%         4.000000
75%         5.000000
max         5.000000
Name: stars, dtype: float64
```

→評論平均數高達 3.77 顆星，可見使用者給分較寬鬆。而這張圖是從 train data 畫出來的，雖然應該用 test data 做比較準。

2.Business_id and stars distribution



→這張圖是取評論超過 13 次的店家作圖，原本的想法是評論較多會不會評價較高，不過從圖看起來是沒有如此相關性。

備註:橫軸應該放 business_id 但因為序號太多所以重新編號。

References

1. https://www.jianshu.com/p/912d4f722e4c

2.http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

3.http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html

4.http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

5.http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVR.html

6.http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

7.http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

8.http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

9.http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html

10.http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html