Database - HW3

109550134 梁詠晴

Motivations

在我們的生活中，時常會有音樂伴隨其中，而音樂中蘊含的情緒也往往會引起人們的共鳴。同時我也注意到，人們有時候並不能成功地辨識、察覺到自己的的情緒，因此在本次實作中，我希望能透過分析句子情緒，推薦使用者相符情緒的音樂。

Application description

流程:

1. 預先擷取Spotify歌單並輸入至DBMS處理資料
2. 預先將Emotions dataset中的資料用sklearn中的DecisionTreeClassifier訓練情緒分類器
3. 使用者輸入句子
4. 用情緒分類器判斷輸入句子的情緒，得到emotion tag
5. 連線database，用query隨機找出符合emotion tag的一首歌曲，並輸出歌名、歌手、url等資訊
6. 利用webbrowser模組根據url開啟Spotify新分頁

Code:

1.Spotify to DBMS (詳見data source)

2.Train Classifier:

```
def basic_info ():
    train_data = pd.read_csv("train.txt", header=None, sep=";", names=["sentence","emo"], encoding="utf-8")
    test_data = pd.read_csv("test.txt", header=None, sep=";", names=["sentence","emo"], encoding="utf-8")
    val_data = pd.read_csv("val.txt", header=None, sep=";", names=["sentence","emo"], encoding="utf-8")
    df = pd.concat([train_data,test_data, val_data])
    df['emo'].unique()
    labelencoder = LabelEncoder()
    df['emo_encoded'] = labelencoder.fit_transform(df['emo'])
    df[['emo','emo_encoded']].drop_duplicates(keep='first')
    sentences = df.sentence.values
    y = df.emo.values
    x_train,x_test,y_train,y_test = train_test_split(sentences,y, test_size=0.1,shuffle=True, random_state=41)
    return x_train,x_test,y_train,y_test
```

```
def decision_tree(x_train,x_test,y_train,y_test):
    print("-------------Decision Tree-------------")
    clf=Pipeline(steps=[("cv",CountVectorizer(stop_words='english', ngram_range=(1,1))),
                ("Dt",tree.DecisionTreeClassifier(criterion="entropy",random_state=50,min_samples_leaf=6))])#,max_depth=MAX_DEPTH
    clf.fit(x_train, y_train)
    y_pred_test = clf.predict(x_test)
    print(classification_report(y_test, y_pred_test))
    print("\nAccuracy for Decision Tree model:",sklearn.metrics.accuracy_score(y_test, y_pred_test))

    acc_score = sklearn.metrics.accuracy_score(y_pred_test,y_test)
    prec_score = precision_score(y_test,y_pred_test, average='macro')
    recall = recall_score(y_test, y_pred_test,average='macro')
    print('Decision Tree Model')
    print(str('Accuracy: '+'{:04.2f}'.format(acc_score*100))+'%')
    print(str('Precision: '+'{:04.2f}'.format(prec_score*100))+'%')
    print(str('Recall: '+'{:04.2f}'.format(recall*100))+'%')

    return clf
```

3.user input -> predict with clf -> sql choose a random song

```
def ask():
    example = input("Hi, share what's in your mind:")
    tree_result = tree_clf.predict([example])[0]
    print("Your mood is "+ tree_result)
    song,artist,url=database.db_random(tree_result)
    print("We recommand you this song: "+song+"---"+artist+"    "+url)
    database.open_web(url)
```

```
def db_random(emo):
    command = "select i.song, ar.artist, i.url " \
        "from song_info i, song_emo e, artist ar where e.emotion = \'"+emo+"\' "\
        "and e.song_id = i.song_id and i.artist_id = ar.artist_id ORDER BY RANDOM() LIMIT 1"
    con.execute(command)
    query_results = con.fetchall()
    return query_results[0][0],query_results[0][1],query_results[0][2]
```

4. Open web

```
def open_web(url):
    open = input("May I play the song for you? (y/n)")
    if(open=='y'or open=='Y'):
        webbrowser.open(url,new=1)
    elif(open=='n'or open=='N'):
        print("That's fine,maybe you need something new...")
    else:
        open_web(url)
```

使用範例:

```
PS D:\database\hw3> python main.py
tree start
-------------Decision Tree-------------
              precision    recall  f1-score   support

       anger       0.86      0.88      0.87       298
        fear       0.82      0.88      0.85       225
         joy       0.87      0.89      0.88       666
        love       0.79      0.77      0.78       161
     sadness       0.92      0.87      0.89       581
    surprise       0.73      0.70      0.71        69

    accuracy                           0.87      2000
   macro avg       0.83      0.83      0.83      2000
weighted avg       0.87      0.87      0.87      2000


Accuracy for Decision Tree model: 0.8655
Decision Tree Model
Accuracy: 86.55%
Precision: 83.18%
Recall: 83.09%
Hi, share what's in your mind:The happy Christmas time is almost here!
Your mood is joy
We recommand you this song: I'll Be There for You - Theme From "Friends"---The Rembrandts   https://open.spotify.com/track/15tHagkk8z306XkyOHqiip
May I play the song for you? (y/n)y
-------------------------
try again?(y/n)n
ok! enjoy your day
```

Data source

Spotify歌單:

用python的spotipy模組連線Spotify API，以Emotions dataset中的六種情緒為關鍵字搜尋現有歌單，擷取歌單中每首歌的資訊 (歌名、歌手、id、url…)，並標上歌曲所屬歌單情緒，輸出成csv檔並輸入到DBMS中的table。

使用歌單時，利用psycopg2連線database，並利用psycopg2中的function執行query。

擷取、輸入至DBMS:

artist	artist_id	name	song_id	emo	duration	popularity	url
Lewis Cap	4GNC7G	Before Yo	2gMXnyr	sadness	00:03:35	82	https://open.spotify.com/track/2gMXnyrvIjhVBUZwvLZDMP
Shawn Me	7n2wHs1	It'll Be Ok	2KnLkZ3z	sadness	00:03:42	82	https://open.spotify.com/track/2KnLkZ3z7PO3kgVGHGqDpD
Lauren Sp	79AyR6A	Fingers Cr	5S9Zs5g9l	sadness	00:02:55	45	https://open.spotify.com/track/5S9Zs5g9lTWnLIboN1pdlU

```python
import spotipy
from spotipy.oauth2 import SpotifyClientCredentials
import csv

sp = spotipy.Spotify(auth_manager=SpotifyClientCredentials(client_id="bd74c8c910954fc09415cfea94d13a26",
                                                           client_secret="b3ce6caec2904c009dac6e776ff49bd1"))
song_dict = {'sadness' : [],'joy' : [],'anger' : [],'love' : [], 'surprise' : [],'fear' : [] }
song_csv = []
playlists = {
            'sadness' : ["https://open.spotify.com/playlist/3c0Nv5CY6TIaRszlTZbUFk",
                "https://open.spotify.com/playlist/70VaOHyjpsWcmA4gxosExZ"],

            'joy' : ["https://open.spotify.com/playlist/1h90L3LP8kAJ7KGjCV2Xfd",
                "https://open.spotify.com/playlist/4AnAUkQNrLKlJCInZGSXRO"],

            'anger' : ["https://open.spotify.com/playlist/5O12S9z3O8dEhHWt3bPbxm",
                    "https://open.spotify.com/playlist/23lYtJMh8NWBahDc4FX8Ee"],

            'love' : ["https://open.spotify.com/playlist/6oNsYDhN95gkENsdFcAwTh",
                    "https://open.spotify.com/playlist/37i9dQZF1DX50QitC6Oqtn"],

            'surprise' : ["https://open.spotify.com/playlist/4o5SxWNsTNLOi9ahHeJF8A"],

            'fear' : ["https://open.spotify.com/playlist/5SJpZzc6nhaezmRDFOm3Bt",
                    "https://open.spotify.com/playlist/6VcH2xJR5uvi8NJ1YlUO74"]
            }
```

```python
for emo, links in playlists.items():
    for i in links:
        id = i.split("/")[4]
        emo_playlist = sp.user_playlist_tracks(playlist_id=id)
        track = emo_playlist['items']
        for song in track :
            artist= song['track']['artists'][0]['name']
            artist_id = song['track']['artists'][0]['id']
            name = song['track']['name']
            url = song['track']['external_urls']['spotify']
            song_id = song["track"]["id"]
            duration = convert(int(song['track']['duration_ms']))
            popularity = song['track']['popularity']

            temp = [artist,artist_id,name,song_id,emo,duration,popularity,url]
            song_csv.append(temp)
```

```sql
create table all_data(artist varchar(150),  artist_id varchar(50),  song varchar(150),  song_id varchar(50),
                      emotion varchar(20),  duration time  popularity integer  url varchar(150))
```

```
\copy all_data FROM 'D:\database\hw3\song.csv' DELIMITER ',' CSV header
```

python應用時連線DBMS:

```python
import psycopg2
import webbrowser

ENDPOINT="database-3.cjxnchctpjfe.us-east-1.rds.amazonaws.com"
PORT="5432"
USER="postgres"
REGION="us-east-1b"
DBNAME="dbhw3"
PASSWORD = "carrie0622"
db = psycopg2.connect(
    database=DBNAME,
    user= USER,
    password= PASSWORD,
    host= ENDPOINT,
    port= PORT
)
con = db.cursor()
```

```python
def db_random(emo):
    command = "select i.song, ar.artist, i.url " \
        "from song_info i, song_emo e, artist ar where e.emotion = \'"+emo+"\' "\
        "and e.song_id = i.song_id and i.artist_id = ar.artist_id ORDER BY RANDOM() LIMIT 1"
    con.execute(command)
    query_results = con.fetchall()
```

Note: 操作過程並無對database進行更動，若要新增歌單/歌，可一樣使用上述程式擷取資料得到csv後，用psql指令將csv加到all_data中處理

Emotion dataset :

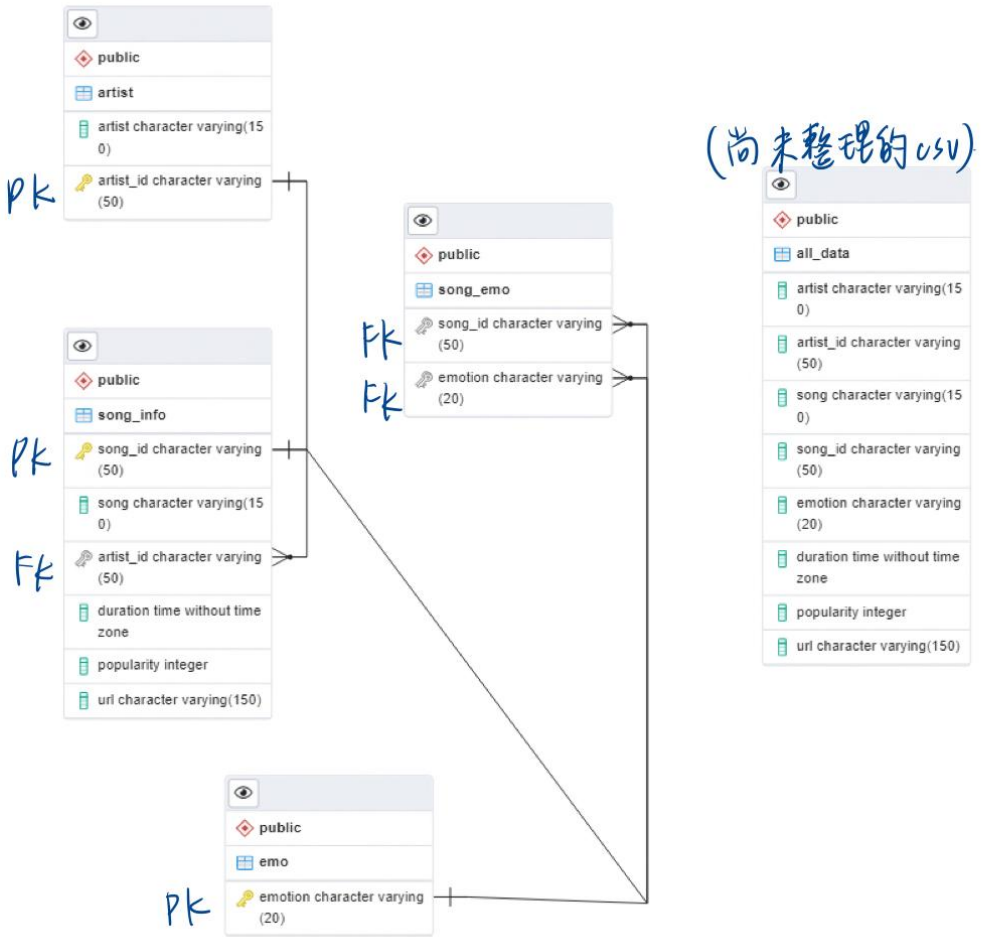
Decision tree 的train/test data則是來自kaggle的 “Emotions dataset for NLP classification tasks” dataset。Dataset中共有20000筆資料，而句子個別標示情緒，有sadness, joy, anger, fear, love and surprise等六種分類。因為主要為training需要資料

故並無再使用DBMS處理，而是直接在python中讀取csv檔。

```
im feeling rather rotten so im not very ambitious right now;sadness
im updating my blog because i feel shitty;sadness
i never make her separate from me because i don t ever want her to feel like i m ashamed with her;sadness
i left with my bouquet of red and yellow tulips under my arm feeling slightly more optimistic than when i arrived;joy
i was feeling a little vain when i did this one;sadness
i cant walk into a shop anywhere where i do not feel uncomfortable;fear
i felt anger when at the end of a telephone call;anger
```

Link of dataset : https://www.kaggle.com/datasets/praveengovi/emotions-dataset-for-nlp

Database schema



原先的資料全部都在同一個表格內，因為

1. 歌手可能對應到多首歌 (1-n)
2. emotion對應多首歌、有些歌會對應到大於一個emotion (n-n)

故將資料拆成emo、song_info、artist_info、song_emo這四個表格:
emo : 存emotion
song_info : 存song_id所對應的歌名、url、artist_id、時長等資訊
artist_info : 存artist_id對應的歌手名字
song_emo : 存song_id和emotion tag的對應關係

all_data : 未整理的資料，直接輸入自python生成的歌單csv

Functional dependency :
song_info : song_id → song、url、artist_id、duration、popularity
artist_info : artist_id → artist_id

SQL queries

emo:

```
create table emo as
select distinct emotion from all_data

alter table emo
ADD PRIMARY KEY (emotion);
```

artist:

```
create table artist as
select distinct artist, artist_id from all_data

alter table artist
ADD PRIMARY KEY (artist_id);
```

song_info:

```
create table song_info as
select distinct song_id,song,artist_id,duration,popularity,url  from all_data

alter table song_info
ADD PRIMARY KEY (song_id);

alter table song_info
ADD FOREIGN KEY (artist_id) REFERENCES artist(artist_id)
```

song_emo:

```
create table song_emo as
select distinct song_id, emotion from all_data

alter table song_emo
ADD FOREIGN KEY (song_id) REFERENCES song_info(song_id)

alter table song_emo
ADD FOREIGN KEY (emotion) REFERENCES emo(emotion)
```

Search for a random song with specific emotion tag:

```
select i.song, ar.artist, i.url
from song_info i, song_emo e, artist ar
where e.emotion = \'"+emo+"\' and e.song_id = i.song_id and i.artist_id = ar.artist_id
ORDER BY RANDOM() LIMIT 1
```