텍스트 기반 감정분석을 통한 음악 추천

- 캡스톤종합프로젝트 1분반
- 지도교수: 김수형 교수님
- 171772 류교서
- 154171 김용정

목차

1.프로젝트의 배경

- 프로젝트 배경 및 문제: 자신의 상황이나 기분에 맞는 음악을 사용자 가 입력한 텍스트로부터 감성 값을 도출해 기분이나 상태를 객관적으로 알 수 있다.

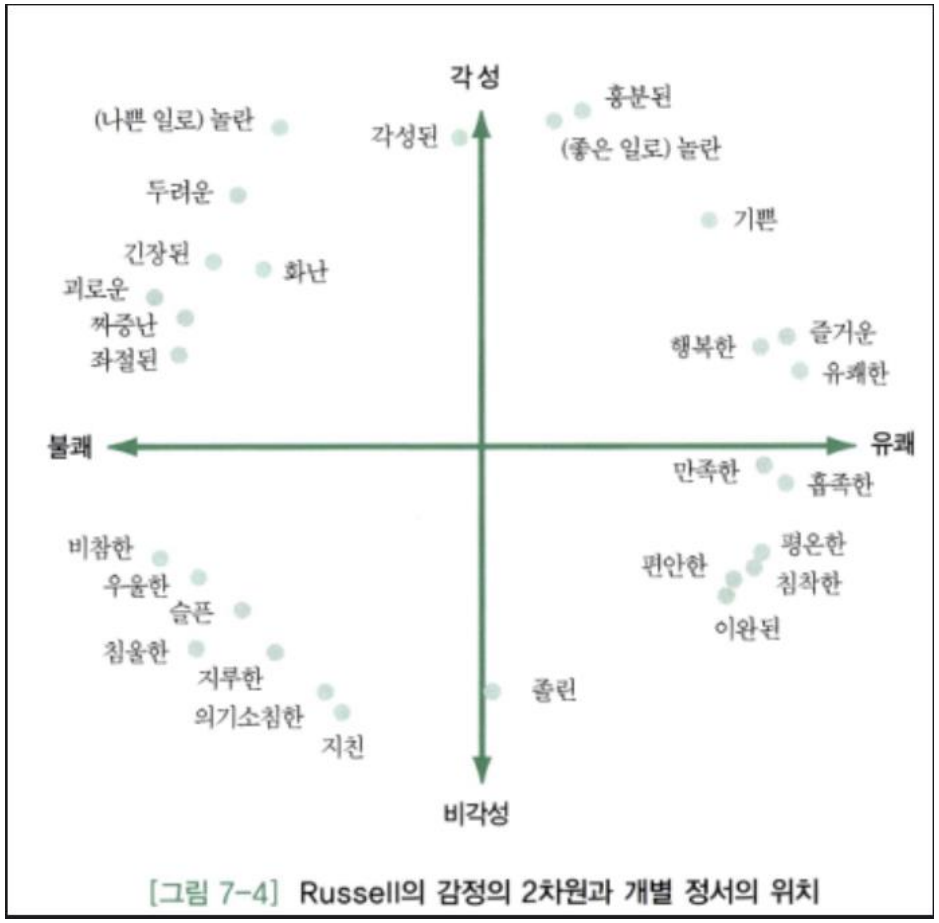
- 프로젝트 목표: 사용자로부터 텍스트를 입력을 받아 Russell감성 모델 기반으로 감성 값을 도출하여 음악에서 추출된 값과 가장 가까운 음악 을 추천.

1-1.러셀의 감정 모델

러셀의 감정 모델

- Arousal(각성,흥분)
  - 위쪽일 수록 각성 또는 흥분 상태가 높음
  - 아래일 수록 각성 또는 흥분 상태가 낮음

- Valence(불쾌 또는 유쾌, 긍정 또는 부정)
  - 왼쪽일 수록 불쾌 또는 부정 ↑
  - 오른쪽일 수록 유쾌 또는 긍정 ↑



[그림 7-4] Russell의 감정의 2차원과 개별 정서의 위치

2.요구사항

1. 텍스트 분석 및 감정 특성 값 도출(류교서)

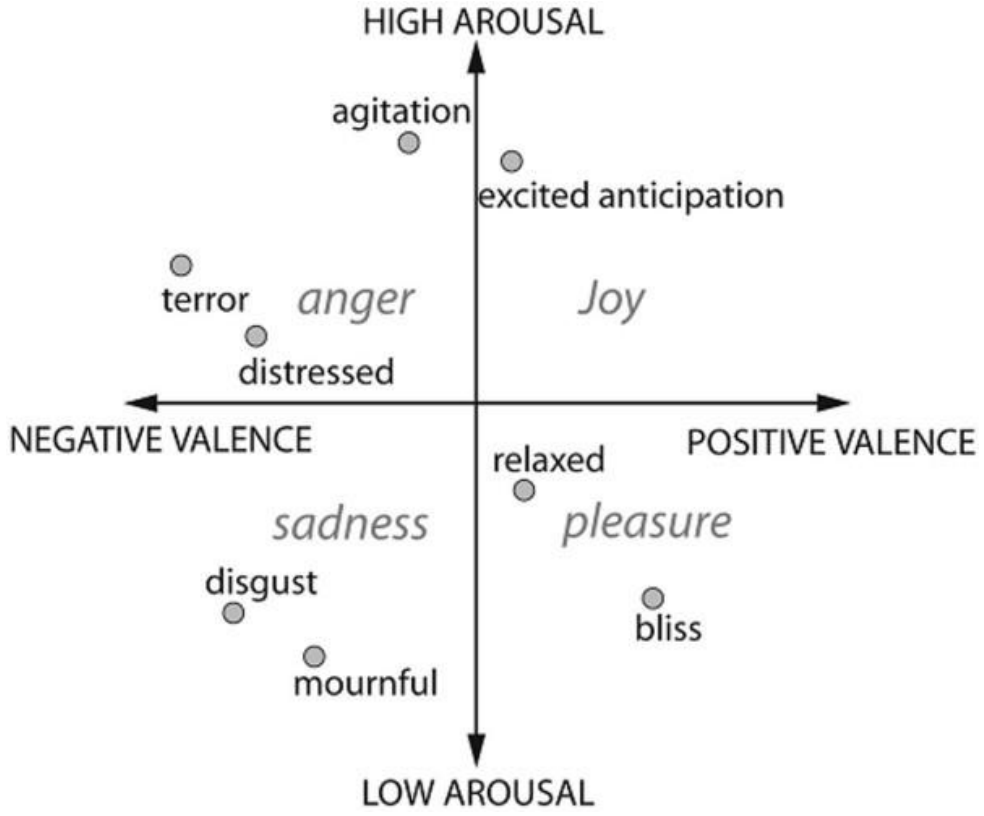
2. 음악 분석 및 감정 특성 값 도출(김용정)

3. 텍스트 특성 값과 가까운 음악매칭(김용정, 류교서)

3.텍스트 분석 및 감정 특성 값 도출

3-1.텍스트 감정 값 추출 개요

- 테일러~설명
- 목표 추출값
- 추출할 감성값에는 정도(Valence) 각성 상태(Arousal) 존재
- 정도(Valence): 긍/부정의 세기를 수치화
- 각성 상태(Arousal): Valence 값의 정도
- 추출 방식
- 파이썬의 NLTK(Natural Language Toolkit, 자연어 처리) 패키지중 워드넷레머타이저(WordNetLemmatizer) 라이브러리와, 감정 단어 사전 ANEW를 사용해 감정값을 추출



3-2.텍스트 감정 값 추출 방식

- 감정 값을 추출할 데이터: 소셜 네트워크 서비스(**SNS**: **S**ocial **N**etworking **S**ervice) 중 Twitter의 데이터를 사용
- twitter api로 데이터 추출
- 데이터에는 target/ID/data/플래그/사용자 /텍스트 형식
- 데이터에서 '텍스트 형식'를 사용

3-3. 입력 데이터

```
if __name__ == '__main__':
    # input Data with main fun parameter
    input_file = (r'C:\Users\user\originalData\originalData4.txt')
    input_dir = (r'C:\Users\user\originalData')
    mode = 'mika'
    output_dir = r'C:\Users\user\SentimentAnalysis-master\SentimentAnalysis-master\anew_' + mode
    if not os.path.exists(output_dir):
        os.makedirs(output_dir)

        # run main
main(input_file, input_dir, output_dir, mode)
```

- Input_file : 감정 값을 추출하기 위한 입력 데이터
- Input_dir : 입력 데이터의 디렉터리
- mode: 감정 값을 추출하는 모드(Valence, Arousal 값 추출 목적으로 지정
- Output_dir: 추출한 감정 값을 저장하는 디렉터리
- 위의 4가지 parameter로 main 함수 실행

3-4. main

```
def main(input_file, input_dir, output_dir, mode):
    if len(output_dir) < 0 or not os.path.exists(output_dir):  # empty output
        print('No output directory specified, or path does not exist')
        sys.exit(0)
    elif len(input_file) == 0 and len(input_dir)  == 0:  # empty input
        print('No input specified. Please give either a single file or a directory of files to analyze.')
        sys.exit(1)
    elif len(input_file) > 0:  # handle single file
        if os.path.exists(input_file):
            analyzefile(input_file, output_dir, mode)
```

3-4.추출 방식과 함수-(main 함수)

```
    else:
        print('Input file "' + input_file + '" is invalid.')
        sys.exit(0)
elif len(input_dir) > 0:  # handle directory
    if os.path.isdir(input_dir):
        directory = os.fsencode(input_dir)
        for file in os.listdir(directory): #주어진 디렉터리에 있는 항목들의 이름을 담고 있는 리스트 반환
            filename = os.path.join(input_dir, os.frowssdecode(file)) #fsdecode 파일 경로 리턴
            print("filename is : ",filename)
            if filename.endswith(".txt"):
                start_time = time.time()
                print("Starting sentiment analysis of " + filename + "...")
                analyzefile(filename, output_dir, mode)
                print("Finished analyzing " + filename + " in " + str((time.time() - start_time)) + " seconds")
    else:
        print('Input directory "' + input_dir + '" is invalid.')
        sys.exit(0)
```

3-4.analyzefile (1/5)- 변수 설정

```
# use anew lexicon, get sentiment values(Valencd, Arousal) and get .csv output file
def analyzefile(input_file, output_dir, mode):

    output_file = os.path.join(output_dir, os.path.basename(input_file).rstrip('.txt') + ".csv") +
    utterances = []
    with open(input_file, 'r',encoding='latin-1') as myfile:
        i = 1
        with open(output_file, 'w', -1, 'utf-8') as csvfile:
            fieldnames = ['Sentence ID', 'Sentence', 'Valence', 'Arousal', 'Dominance', 'Sentiment Label',
                          'Average VAD', '# Words Found', 'Found Words', 'All Words']
            writer = csv.DictWriter(csvfile, delimiter=';', fieldnames=fieldnames)
            writer.writeheader()
            # analyze sentence per line
            for line in myfile.readlines():
                s = tokenize.word_tokenize(line.lower())
                all_words = []
                found_words = []
                total_words = 0
                v_list = []  # hold Valence, Arousal, Dominace value
                a_list = []
                d_list = []
```

- output file 설정 및 input file open

3-4. analyzefile (2/5)-품사 태그화

```
# search words in ANEW
words = nltk.pos_tag(s) #tokenized words to tagging(by identifying part of speech)
for index, p in enumerate(words): #순서가 있는 자료형을 입력으로 받아 인덱스 값을 포함하는 enumerate 객체 리턴
    w = p[0] #p의 words
    pos = p[1]
    if w in stops or not w.isalpha():
        continue
    j = index-1
    neg = False
    while j >= 0 and j >= index-3:
        if words[j][0] == 'not' or words[j][0] == 'no' or words[j][0] == 'n\'t':
            neg = True
            break
        j -= 1
```

- ANEW 어휘집에서 words 검색
- 문장의 단어마다 품사 식별해 태그 설정
- 튜플 형태('단어','태그') 형태 로 설정

3-4. analyzefile (3/5)- 표제어 추출

```
# use anew lexicon, get sentiment values(Valencd, Arousal) and get .csv output file
def analyzefile(input_file, output_dir, mode):

    output_file = os.path.join(output_dir, os.path.basename(input_file).rstrip('.txt') + ".csv") +
    utterances = []
    with open(input_file, 'r',encoding='latin-1') as myfile:
        i = 1
        with open(output_file, 'w', -1, 'utf-8') as csvfile:
            fieldnames = ['Sentence ID', 'Sentence', 'Valence', 'Arousal', 'Dominance', 'Sentiment Label',
                          'Average VAD', '# Words Found', 'Found Words', 'All Words']
            writer = csv.DictWriter(csvfile, delimiter=';', fieldnames=fieldnames)
            writer.writeheader()
            # analyze sentence per line
            for line in myfile.readlines():
                s = tokenize.word_tokenize(line.lower())
                all_words = []
                found_words = []
                total_words = 0
                v_list = []  # hold Valence, Arousal, Dominace value
                a_list = []
                d_list = []
```

- 품사에 기초해 표제어 추출
- am, are, is 의 표제어는 be

3-4. analyzefile (4/5)- 감정 값 계산

```
#valence cal
if statistics.mean(v_list) < avg_V: #v_list 의 평균값이 avg_V 보다 작으면 최대값-평균값
    sentiment = max(v_list) - avg_V
elif max(v_list) < avg_V:
    sentiment = avg_V - min(v_list)
else:
    sentiment = max(v_list) - min(v_list)
#arousal cal
if statistics.mean(a_list) < avg_A:
    arousal = max(a_list) - avg_A
    print(arousal)
elif max(a_list) < avg_A:
    arousal = avg_A - min(a_list)
else:
    arousal = max(a_list) - min(a_list)
```

- Valence 감정 값 계산
- v_list 의 각 평균, 최대 값과 avg_V 값의 크기 차이로 결정
- Arousal 감정 값 계산
- a_list 의 각 평균, 최대 값과 avg_A 값의 크기 차이로 결정

3-4. analyzefile (5/5)- 감정 값 입력

```
label = 'neutral'
if sentiment > 6:
    label = 'positive'
elif sentiment < 4:
    label = 'negative'
writer.writerow({'Sentence': line,
                 'Valence': sentiment,
                 'Arousal': arousal,
                 'Dominance': dominance,
                })
i += 1
```

- 추출한 감정 값을 output file에 입력

3-5.감정 값 추출 결과

In [19]:
```
import pandas as pd
csv_test = pd.read_csv('C:/Users/user/SentimentAnalysis-master/SentimentAnalysis-master/anew_mika/originalData4.csv',  sep=';' ,error_bad_line
csv_test
```

Out[19]:

	Sentence ID	Sentence	Valence	Arousal	Dominance	Sentiment Label	Average VAD	# Words Found	Found Words	All Words
0	NaN	0\t1822926805\tSat May 16 20:42:30 PDT 2009\tN...	4.80	0.41	2.90	NaN	NaN	NaN	NaN	NaN
1	NaN	0\t1822926890\tSat May 16 20:42:31 PDT 2009\tN...	0.85	0.60	2.71	NaN	NaN	NaN	NaN	NaN
2	NaN	0\t1822926956\tSat May 16 20:42:32 PDT 2009\tN...	4.26	1.84	2.91	NaN	NaN	NaN	NaN	NaN
3	NaN	0\t1822927034\tSat May 16 20:42:32 PDT 2009\tN...	1.50	0.84	0.91	NaN	NaN	NaN	NaN	NaN
4	NaN	0\t1822927136\tSat May 16 20:42:33 PDT 2009\tN...	1.52	3.52	0.76	NaN	NaN	NaN	NaN	NaN
5	NaN	0\t1822927637\tSat May 16 20:42:37 PDT 2009\tN...	4.62	2.86	3.55	NaN	NaN	NaN	NaN	NaN
6	NaN	0\t1822927802\tSat May 16 20:42:39 PDT 2009\tN...	1.30	0.93	0.78	NaN	NaN	NaN	NaN	NaN
7	NaN	0\t1822927855\tSat May 16 20:42:39 PDT 2009\tN...	4.26	0.17	3.94	NaN	NaN	NaN	NaN	NaN

4. 음악 분석 및 감정 특성 값 도출

4-1.입력 데이터

- DEAM dataset라는 openSMILE을 통해 특징이 추출된 csv파일 제공
- 각 특징의 표준 편차를 제외한 133개의 평균 특징이 있음
- 출처:http://cvml.unige.ch/databases/DEAM/

F0final_sma_amean	voicingFinalUnclipped_sma_amean	jitterLocal_sma_amean	jitterDDP_sma_amean
93.884056	0.742852	0.099609	0.095736
62.682589	0.754430	0.056241	0.054784
92.850316	0.753095	0.081527	0.095950
158.673853	0.757328	0.101659	0.108718
83.823484	0.787512	0.059757	0.060557

4-1출력 데이터

- DEAM dataset에서 1802곡의 Arousal, Valence 값을 csv파일로 제공
- 평균만 사용
- 출처:http://cvml.unige.ch/databases/DEAM/

	valence_mean	arousal_mean
0	3.10	3.00
1	3.50	3.30
2	5.70	5.50
3	4.40	5.30
4	5.80	6.40
...	...	...
53	5.40	3.60
54	5.00	5.20
55	5.00	4.60
56	3.17	6.83
57	3.80	5.80

4-2.XGBRegressor(arousal)

```
#XGBRegressor(arousal)
from xgboost import XGBRegressor
model_xgb = XGBRegressor(learning_rate=0.1,
                         max_depth=5,
                         n_estimators=100,n_jobs=12)
model_xgb.fit(x_train,y_train)
xgb_train_pred =model_xgb.predict(x_train).round(3)
xgb_pred = model_xgb.predict(x_test).round(3)
print(model_xgb.score(x_train,y_train))
print(model_xgb.score(x_test, y_test))
print(rmsle(y_train, xgb_train_pred))
print(rmsle(y_test, xgb_pred))
```

```
0.8642984285314614
-0.09307777407506213
0.4778655634475268 4
1.2713175792027964
```

4-2.랜덤포레스트(arousal)

```python
#랜덤포레스트(arousal)
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor(random_state=42, n_estimators=100)
model.fit(x_train, y_train)
rf_train_pred =model.predict(x_train).round(3)
rf_pred = model.predict(x_test).round(3)
print(model.score(x_train, y_train))
print(model.score(x_test, y_test))
print(rmsle(y_train, rf_train_pred))
print(rmsle(y_test, rf_pred))
```

```
0.856848389990899
-0.03978997544207119
0.490812534262355
1.2399511551008817
```

4-2. XGBRegressor(valence)

```
#XGBRegressor(valence)
from xgboost import XGBRegressor
model_xgb = XGBRegressor(learning_rate=0.1,
                         max_depth=5,
                         n_estimators=100,n_jobs=12)
model_xgb.fit(x_train_v,y_train_v)
xgb_train_pred_v =model_xgb.predict(x_train_v).round(3)
xgb_pred_v = model_xgb.predict(x_test_v).round(3)
print(rmsle(y_train_v, xgb_train_pred_v))
print(rmsle(y_test_v, xgb_pred_v))
print(model_xgb.score(x_train_v,y_train_v))
print(model_xgb.score(x_test_v, y_test_v))
```

```
0.4213838316252648
1.2158192851088991
0.8718340486373322
-0.1027834480664036
```

4-2. 랜덤포레스트(valence)

```
#랜덤포레스트(valence)
model_rf = RandomForestRegressor(random_state=42, n_estimators=100)
model_rf.fit(x_train_v, y_train_v)
train_pred_rf_v = model_rf.predict(x_train_v).round(3)
pred_rf_v = model_rf.predict(x_test_v).round(3)
print(rmsle(y_train_v, train_pred_rf_v))
print(rmsle(y_test_v, pred_rf_v))
print(model_rf.score(x_train_v,y_train_v))
print(model_rf.score(x_test_v, y_test_v))
```

```
0.4386813207554389
1.1653241735083337
0.8610996901069535
-0.013144919956646 24
```

4-3.문제점

- 문제점
  다른 알고리즘을 사용하거나 매개 변수를 변경하여도
  Score 점수가 음수가 나온다.
- 해결방법

 사이트에 제공된 음악 파일을 통해 Librosa 라이브러리 통해 특징들을 다시 뽑음

 제공된 Arousal과 Valence 값만 다시 사용

4-4.특징 추출

- 사용된 특징들
  - Tempo
  - Tonnez(6)
  - Mfcc(20)
  - Chroma_shft(12)

(Librosa 사용)

- Rmse
- Rolloff
- Zero_crossing_rate
- Spectral(centroid,bandwidth,contrast,flatness)

	tempo	chroma_shft_1	chroma_shft_2	chroma_shft_3	chroma_shft_4	chroma_shft_5	chroma_shft_6
0	143.554688	0.232202	0.201824	0.222168	0.307799	0.472035	0.495194
1	95.703125	0.437333	0.364443	0.396659	0.466947	0.577430	0.489825
2	172.265625	0.276094	0.208899	0.237324	0.251551	0.484929	0.253767
3	99.384014	0.252043	0.334039	0.275257	0.431722	0.289240	0.228196
4	117.453835	0.335063	0.277743	0.340410	0.380032	0.427186	0.445793

4-4. 랜덤포레스트(Arousal)-Librosa

- Score점수 변화
- -0.03->0.42

```
#랜덤포레스트(Arousal)
from sklearn.ensemble import RandomForestRegressor
model_rf = RandomForestRegressor(random_state=42,max_depth=10,n_estimators=100)
model_rf.fit(x_train, y_train)
train_pred_rf =model_rf.predict(x_train)
test_pred_rf = model_rf.predict(x_test)
print(model_rf.score(x_train, y_train))
print(model_rf.score(x_test, y_test))
print(rmsle(y_train, train_pred_rf))
print(rmsle(y_test, test_pred_rf))
```

```
C:₩anaconda₩lib₩site-packages₩ipykernel_launcher.py:3: DataConversionWarning: A
en a 1d array was expected. Please change the shape of y to (n_samples,), for ex
  This is separate from the ipykernel package so we can avoid doing imports unt
```

```
0.8242545867335772
0.42665622656753766
0.5391509254339317
0.9651381556731333
```

4-4. 랜덤포레스트(Valence)-Librosa

- Score점수 변화
- -0.013->0.44

```
#랜덤포레스트(Valence)
model_rf_2 = RandomForestRegressor(random_state=42, n_estimators=100)
model_rf_2.fit(x_train_2, y_train_2)
train_pred_rf_2 =model_rf_2.predict(x_train_2)
test_pred_rf_2 = model_rf_2.predict(x_test_2)
print(model_rf_2.score(x_train_2, y_train_2))
print(model_rf_2.score(x_test_2, y_test_2))
print(rmsle(y_train_2, train_pred_rf_2))
print(rmsle(y_test_2, test_pred_rf_2))
```

```
C:₩anaconda₩lib₩site-packages₩ipykernel_launcher.py:2: DataConversionW
en a 1d array was expected. Please change the shape of y to (n_samples
```

```
0.9187179742493498
0.4465508469555901
0.33764942347264 39
0.8417140375102 63
```

4-5.arousal 및 valence 값 예측

- Librosa를 통해 음악 특징을 추출함
- 랜덤포레스트(arousal,valence) 모델에 적용하여 arousal,valence값 예측함

	songname	arousal	valence
0	01. 가슴이 시린 게 (My Heartstore) - Lee Hyun 이현 (8e...	5.49	5.23
1	ABTB - Free Rider (무임승차)	5.80	6.79
2	AOA - 심쿵해 (Heart Attack) Music Video	5.91	6.86
3	Adele - Someone Like You (Official Music Video)	5.26	5.07
4	And Now The Day Is Done	5.36	4.07
...	...	...	...
156	하현우 (국카스텐) - DARKNESS [메이플스토리 MapleStory OST]	5.79	6.56
157	하현우 - 돌덩이 (이태원 클라쓰 OST PART.03) 가사 ITAEWON C...	5.96	5.89
158	한희정-더이상슬픔을노래하지않으리	4.53	4.89
159	황인욱-Phocha	5.82	5.33
160	휘성 (Whee Sung) - 결혼까지 생각했어 (Even thought of ma...	5.65	6.63

5.앞으로의 계획

- 텍스트 특성값과 가까운 음악매칭
- 텍스트 추출에서 영어만 사용할 수 있는 단점-> 영어 사용(google.api)해 단어의 다양화 목적

Q&A
감사합니다