Hello!

텍스트 입력 기반으로 감성 분석을 활용한 음악 플레이리스트 추천 서비스

2022 빅데이터분석가양성과정-MINI PROJECT

김소희,이소민

CONTENTS.

Project Introduce.

[주제] 텍스트 입력 기반으로 감성 분석을 활용한 음악 플레이리스트 추천 서비스

- 단순 플레이리스트 추천이 아닌 사용자의 감정에 기반한 플레이리스트를 추천해줌으로써 감정해소를 이룰 수 있음.

- 감정을 긍정/부정이 아닌 6가지로 세밀하게 분석해줌.
  → 공포 / 놀람 / 분노 / 혐오 / 슬픔 / 행복

Data List.

AI Hub

- 한국어 감정 정보가 포함된 단발성 대화 데이터 셋

DATA 1

AI Hub

- 한국어 감정 정보가 포함된 연속적 대화 데이터 셋

DATA 2

KAKAO ARENA

- Melon Platlist Dataset
→ Train.json file
(115,071개 플레이리스트 원본 데이터가 포함된 파일 사용)

DATA 3

Pre-Processing

STEP 1

Chat dataset pre-processing

- 감정 분류 중 '중립' 제거
- 단발성 & 연속적 데이터 결합
- 감정 라벨링

STEP 2

Melon dataset pre-processing

- 상위 노출 태그 추출
- 단일태그(장르) 제거, 감정태그만 추출
- 멜론태그를 6가지감정으로 재분류
- 플레이리스트 좋아요 0~1개 목록 제거

01. 대화데이터셋 전처리 - 단발성 대화 데이터셋

단발성 대화 원본 데이터 : 총 38,593 문장

	Sentence	Emotion
0	언니 동생으로 부르는게 맞는 일인가요..??	공포
1	그냥 내 느낌일뿐겠지?	공포
2	아직너무초기라서 그런거죠?	공포
3	유치원버스 사고 낫다던데	공포
4	근데 원래이런거맞나요	공포
...	...	...
38589	솔직히 예보 제대로 못하는 데 세금이라도 아끼게 그냥 폐지해라..	혐오
38590	재미가 없으니 망하지	혐오
38591	공장 도시락 비우생적임 아르바이트했는데 화장실가성 손도 않씯고 재료 담고 바닥 떨어...	혐오
38592	코딱지 만한 나라에서 지들끼리 피터지게 싸우는 센징 클래스 ㅉㅉㅉ	혐오
38593	와이프도 그렇고 댓글 다 볼텐데 이휘재 좀 하차 하라고 전해주세요	혐오

38594 rows × 2 columns

01. 단발성대화데이터셋 전처리 - 중립 감정 제거

```
df.groupby('Emotion').count()
```

Emotion	Sentence
공포	5468
놀람	5898
분노	5665
슬픔	5267
중립	4830
행복	6037
혐오	5429


```
df = df.drop(df[df['Emotion'] == '중립'].index)

df.groupby('Emotion').count()
```

Emotion	Sentence
공포	5468
놀람	5898
분노	5665
슬픔	5267
행복	6037
혐오	5429

01.대화데이터셋 전처리- 연속적 대화 데이터셋

연속적 대화 원본 데이터 : 총 55,628 문장

	Sentence	Emotion
0	발화	감정
1	아 진짜! 사무실에서 피지 말라니깐! 간접흡연이 얼마나 안좋은데!	분노
2	그럼 직접흡연하는 난 얼마나 안좋겠니? 안그래? 보면 꼭... 지 생각만 하고.	혐오
3	손님 왔어요.	중립
4	손님? 누구?	중립
...	...	...
55624	얘긴 다 끝났냐? 원예부	중립
55625	예. 그거 때문에, 부탁이 있.....는.....데요.	중립
55626	여자 숨겨달라는거면 사절이다.	중립
55627	아무래도 안되나요?	중립
55628	그 여자랑 내가 무슨 상관인데? 아까는 탐정님이 부탁하기에 너 구하는 김에 주워왔지...	중립

55629 rows × 2 columns

01.연속적대화데이터셋 전처리- 중립 및 오타감정제거

연속적 대화 데이터에서 중립 감정, 오타 감정을 제거함

```
df2.groupby('Emotion').count()
```

Emotion	Sentence
ㄴ중립	1
ㅈ중립	1
ㅍ	12
감정	1
공포	98
놀람	4866
분	4
분ㄴ	1
분노	3628
슬픔	1972
줄	1
중림	1
중립	43786
행복	1030
혐오	220

```
df2 = df2.drop(index=df2.loc[df2.Emotion == 'ㄴ중립'].index)
df2 = df2.drop(index=df2.loc[df2.Emotion == 'ㅈ중립'].index)
df2 = df2.drop(index=df2.loc[df2.Emotion == 'ㅍ'].index)
df2 = df2.drop(index=df2.loc[df2.Emotion == '감정'].index)
df2 = df2.drop(index=df2.loc[df2.Emotion == '공포'].index)
df2 = df2.drop(index=df2.loc[df2.Emotion == '분'].index)
df2 = df2.drop(index=df2.loc[df2.Emotion == '분ㄴ'].index)
df2 = df2.drop(index=df2.loc[df2.Emotion == '줄'].index)
df2 = df2.drop(index=df2.loc[df2.Emotion == '중림'].index)
df2 = df2.drop(index=df2.loc[df2.Emotion == '중립'].index)
```

```
df2.groupby('Emotion').count()
```

Emotion	Sentence
놀람	4866
분노	3628
슬픔	1972
행복	1030
혐오	220

01.대화데이터셋 전처리- 단발성& 연속적데이터셋 결합

```
chat = pd.concat([schat, lchat], ignore_index=True)
chat
```

	Sentence	Emotion
0	언니 동생으로 부르는게 맞는 일인가요..??	공포
1	그냥 내 느낌일뿐겠지?	공포
2	아직너무초기라서 그런거죠?	공포
3	유치원버스 사고 낫다던데	공포
4	근데 원래이런거맞나요	공포
...	...	...
45475	뭐? 다시 한 번 말해봐.	분노
45476	어? 정말요?	놀람
45477	혹시, 다들 은행 계좌없는 거예요?	놀람
45478	자네는 대체 뭘 하러 왔나! 젖은 생쥐 꼴이 된 나를 보면서 비웃으러 왔나?	분노
45479	정말?	놀람

45480 rows × 2 columns

최종 대화 데이터 : 총 45,479 문장

모델링 수행 시 전체 데이터의 30%를 테스트 세트로 분류

01.대화데이터셋전처리- 감정라벨링

6가지로 라벨링 > 0 : 공포 / 1 : 놀람 / 2: 분노 / 3 : 혐오 / 4 : 슬픔 / 5: 행복

```
df['Emotion'] = df['Emotion'].str.replace('공포', '0')
df['Emotion'] = df['Emotion'].str.replace('놀람', '1')
df['Emotion'] = df['Emotion'].str.replace('분노', '2')
df['Emotion'] = df['Emotion'].str.replace('혐오', '3')
df['Emotion'] = df['Emotion'].str.replace('슬픔', '4')
df['Emotion'] = df['Emotion'].str.replace('행복', '5')
```

	Sentence	Label
0	언니 동생으로 부르는게 맞는 일인가요..??	0
1	그냥 내 느낌일뿐겠지?	0
2	아직너무초기라서 그런거죠?	0
3	유치원버스 사고 낫다던데	0
4	근데 원래이런거맞나요	0
...	...	...
45475	뭐? 다시 한 번 말해봐.	2
45476	어? 정말요?	1
45477	혹시, 다들 은행 계좌없는 거예요?	1
45478	자네는 대체 뭘 하러 왔나! 젖은 생쥐 꼴이 된 나를 보면서 비웃으러 왔나?	2
45479	정말?	1

45480 rows × 2 columns

02멜론데이터셋전처리- 상위 노출 태그추출

- 태그 노출 빈도수가 1,000회 이상인 태그만 추출
- WordCloud 활용하여 결과 시각화

02.멜론데이터셋전처리- 단일 태그제거

대화 데이터와 멜론 태그의 감정 분류 일치를 위해 단독으로 쓰인 장르 태그 제거

```
# 장르를 제거한 감정 태그만 추출하기
genre = ['발라드', '팝', '락', '랩', '힙합', '일렉', '뉴에이지', '소울', '알앤비', '클래식', '인디',
'월드뮤직', '가요', '록', '댄스', '매장음악', 'Pop', '공연', '셋리스트']
only_genre = []

for t in list_tag:
    for g in genre:
        if g in t:
            only_genre.append(t)
emo_tag = set(list_tag) - set(only_genre)
emo_tag_list = list(emo_tag)
print(emo_tag_list), len(emo_tag_list)
```

	tag_list	plylst_cnt
51670	['발라드']	5497
58918	['팝']	3147
48836	['락']	2977
48990	['랩', '힙합']	2550
45955	['댄스']	1558
...	...	...
57210	['운동']	97
17921	['가요']	94
41115	['까페', '비오는날']	94
56499	['여름', '휴식', '힐링']	94
49306	['록']	92

02.멜론데이터셋전처리- 멜론태그 6가지감정으로 재분류

	id	plylst_title	songs	like_cnt	tag	Emotion	plylst_cnt
1	111944	신화를 추억하며	[23743, 698749, 402171, 421925, 388924, 102519...	12	추억, 회상	None	372
3	46611	아버지 어머니께 바치는 노래	[138932, 176304, 473514, 78625, 251980, 212909...	17	추억, 회상	None	372
4	20864	추억에 젖는 25살의 학창시절	[391924, 629705, 198747, 506139, 333925, 38356...	5	추억, 회상	None	372
6	12697	추억의ost	[636163, 354666, 106249, 320601, 386522, 41898...	3	추억, 회상	None	372
7	15920	토토가 덕분에 다시듣는 명곡들!	[371691, 326088, 438424, 539090, 275179, 82293...	19	추억, 회상	None	372
...	...	...	...	...	...	...	...
6667	90220	스트레스 저리 비켜!	[700090, 433453, 125822, 440239, 193571, 27358...	30	기분전환, 스트레스	None	622
6669	101698	스트레스엔 스타리그BGM!!	[596414, 494891, 524265, 211325, 652662, 55421...	11	기분전환, 스트레스	None	622
6670	4504	80년생들은 한번쯤 흥얼거렸을 법한 스트레스 해소 가요	[689057, 123294, 263272, 14156, 615586, 80656,...	277	기분전환, 스트레스	None	622
6671	38264	선정성 안무로 방송금지된 걸그룹 노래모음	[104087, 233074, 527255, 34363, 284159, 447424...	49	기분전환, 스트레스	None	622
6672	2574	올한해의 아이돌 히트곡들!	[492163, 699874, 360197, 355939, 498935, 32282...	36	드라이브, 스트레스	None	110

감정 분류 모델링을 하기 위해

'Emotion' column 추가

02.멜론데이터셋 전처리- 멜론태그 6가지감정으로 재분류

공포 : 555개 / 놀람 : 693개 / 분노 : 771개

슬픔 : 1,298개 / 행복 : 1339개 / 혐오 : 580개

Emotion	id	plylst_title	songs	like_cnt	tag	plylst_cnt
공포	556	556	556	556	556	556
놀람	693	693	693	693	693	693
분노	771	771	771	771	771	771
슬픔	1298	1298	1298	1298	1298	1298
행복	1339	1339	1339	1339	1339	1339
혐오	580	580	580	580	580	580

<행복> 감정 태그

행복	감성	72
	겨울	136
	겨울, 까페	106
	드라이브	182
	드라이브, 운동	146
	봄	95
	사랑, 설렘, 잔잔한	14
	산책, 여행	16
	여름	80
	운동	86
	잔잔한, 추억, 회상	308
	클럽	98

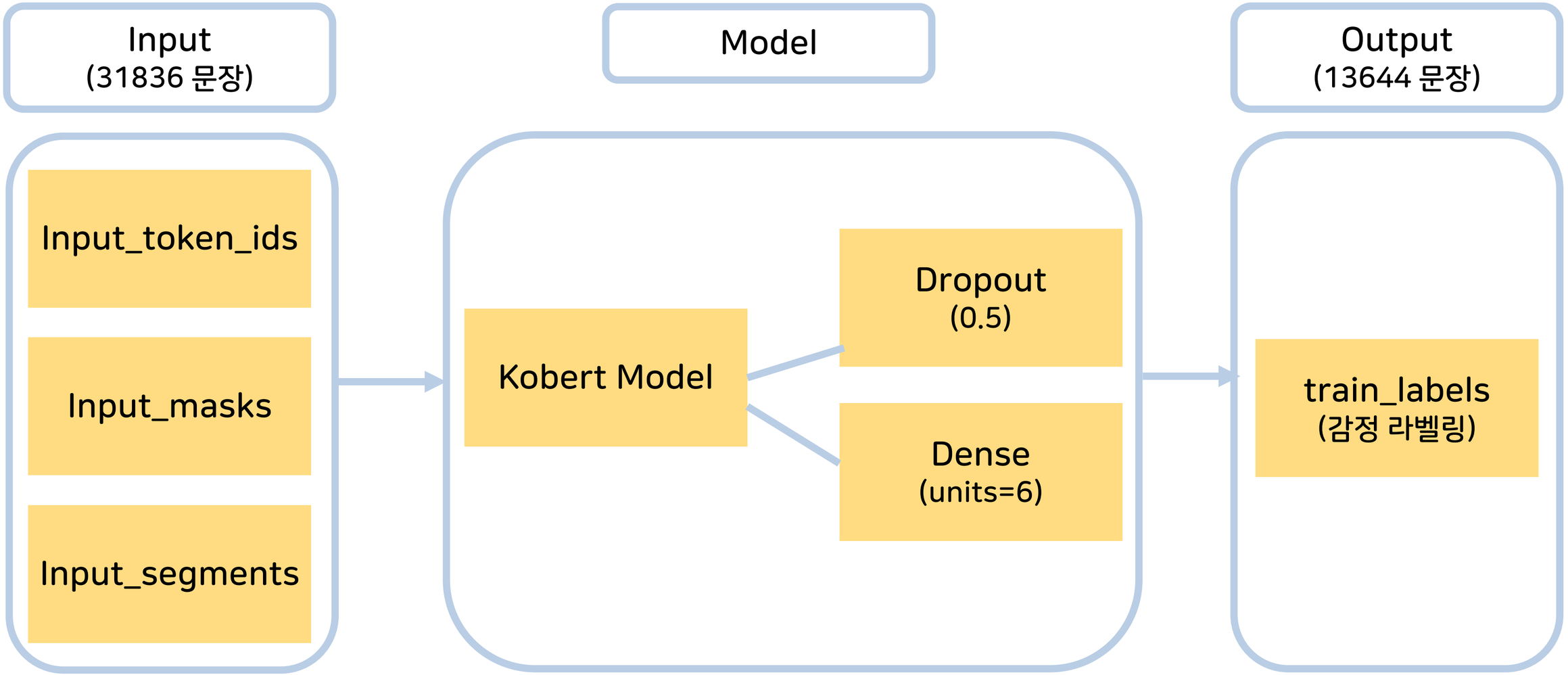
02 멜론데이터셋 전처리 - 좋아요 1개 이하 플레이리스트 제거

좋아요 1개 이하는 제목이 태그와 연관되지 않거나 형식이 올바르지 않기 때문에 플레이리스트 제거

```
emo_tag_train = emo_tag_train[emo_tag_train['like_cnt'] > 1]
emo_tag_train
```

57718	['힙합', '붐뱁', '한	한국힙합	1
57719	['까페', '잔잔한']	욱욱욱욱	1
57751	['운동']	창훈창훈	1
57761	['그루브', '일렉트	서정적 감동 타게 되는 일렉트로니카	1
57762	['변화하다', '씨잼	C jamm	1

KoBert기반다중감성분류모델



모델 학습=>과대적합발생

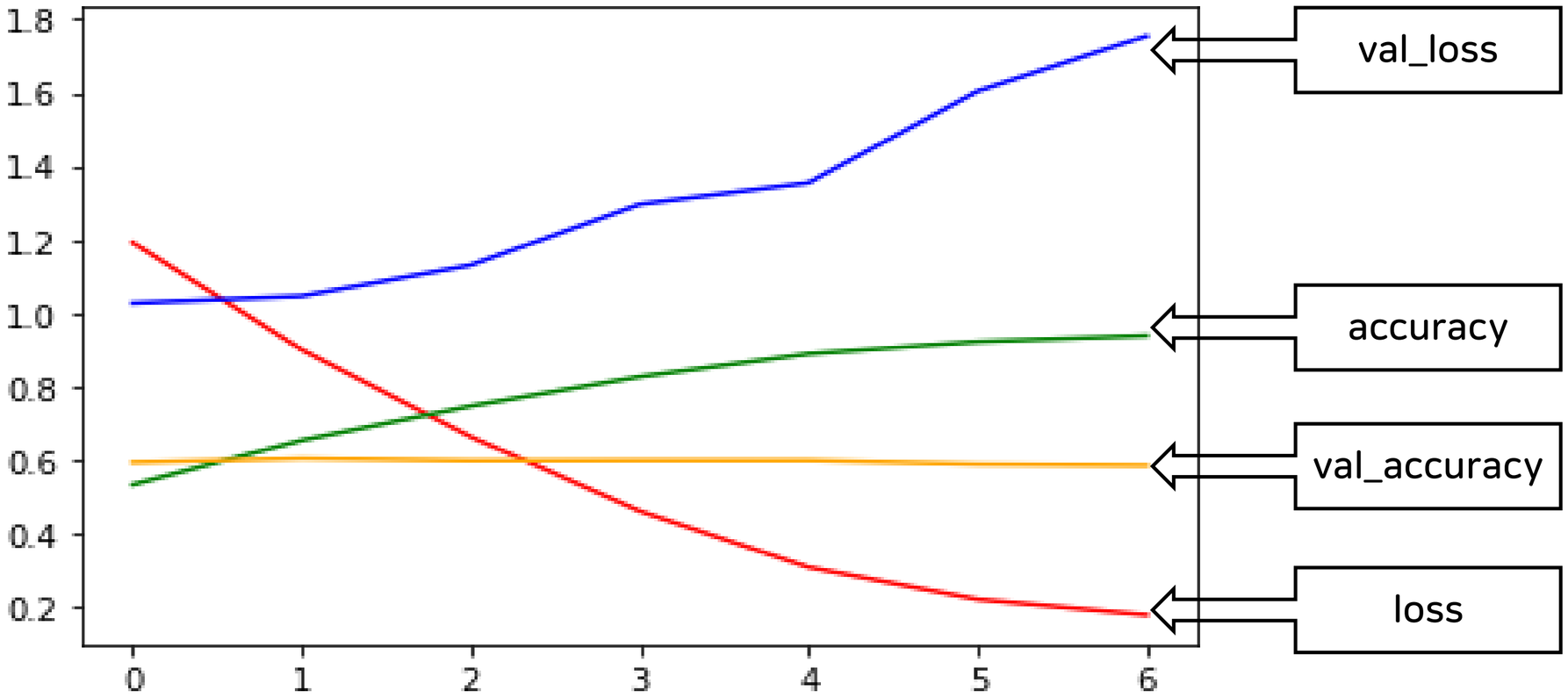
훈련세트 정확도는 93%인 반면, 검증세트 정확도는 59%로 낮은 성능을 보임

```
history = model.fit(train_inputs, train_labels, validation_split=0.2,
                    epochs=10, batch_size=64,
                    verbose=1,
                    callbacks=[checkpoint, earlystopping])
```

```
Epoch 5: val_sparse_categorical_accuracy did not improve from 0.60474
398/398 [==============================] - 78s 197ms/step - loss: 0.3255 - sparse_categorical_accuracy: 0.8854 - val_loss: 1.4481 - val_sparse_categorical_accuracy: 0.5974
Epoch 6/10
398/398 [==============================] - ETA: 0s - loss: 0.2270 - sparse_categorical_accuracy: 0.9211
Epoch 6: val_sparse_categorical_accuracy did not improve from 0.60474
398/398 [==============================] - 78s 197ms/step - loss: 0.2270 - sparse_categorical_accuracy: 0.9211 - val_loss: 1.6476 - val_sparse_categorical_accuracy: 0.5787
Epoch 7/10
398/398 [==============================] - ETA: 0s - loss: 0.1861 - sparse_categorical_accuracy: 0.9354
Epoch 7: val_sparse_categorical_accuracy did not improve from 0.60474
398/398 [==============================] - 79s 198ms/step - loss: 0.1861 - sparse_categorical_accuracy: 0.9354 - val_loss: 1.7802 - val_sparse_categorical_accuracy: 0.5878
```

모델 학습=>과대적합발생

시각화를 통해 훈련세트 손실, 정확도와 비교했을 때 검증세트 손실, 정확도가 낮은 성능을 보임을 알 수 있음



모델 튜닝 ①Dropout: 0.2 -> 0.5 변경

Dropout을 조정한 결과 처음보다 검증세트 손실, 정확도가 약간 좋아졌지만
여전히 과대적합의 문제가 있음

```
bert_outputs = bert_outputs[1]
bert_outputs = layers.Dropout(0.2)(bert_outputs)
```

```
bert_outputs = bert_outputs[1]
bert_outputs = layers.Dropout(0.5)(bert_outputs)
```

```
history = model.fit(train_inputs, train_labels, validation_split=0.25,
                    epochs=10, batch_size=64,
                    verbose=1,
                    callbacks=[checkpoint, earlystopping])
```

```
Epoch 6: val_sparse_categorical_accuracy did not improve from 0.61163
374/374 [==============================] - 75s 201ms/step - loss: 0.2395 - sparse_categorical_accuracy: 0.9195 - val_loss: 1.5540 - val_sparse_categorical_accuracy: 0.5859
Epoch 7/10
374/374 [==============================] - ETA: 0s - loss: 0.1897 - sparse_categorical_accuracy: 0.9363
Epoch 7: val_sparse_categorical_accuracy did not improve from 0.61163
374/374 [==============================] - 75s 202ms/step - loss: 0.1897 - sparse_categorical_accuracy: 0.9363 - val_loss: 1.8008 - val_sparse_categorical_accuracy: 0.5910
```

모델 튜닝 ②random_state: 42 -> 2 변경

Random_state까지 조정한 결과 훈련세트 정확도 97%, 검증세트 정확도 91%로
높은 성과를 보이며 과대적합을 해결함

```
train_x, test_x, train_y, test_y = model_selection.train_test_split(df['Sentence'], df['Label'],
                                                    test_size=0.3,
                                                    random_state=42)
```

```
model_selection.train_test_split(df['Sentence'], df['Label'],
                                 test_size=0.3,
                                 random_state=2)
```

```
history = model.fit(train_inputs, train_labels, validation_split=0.25,
                    epochs=10, batch_size=64,
                    verbose=1,
                    callbacks=[checkpoint, earlystopping])
```

```
Epoch 5: val_sparse_categorical_accuracy did not improve from 0.94434
374/374 [==============================] - 76s 202ms/step - loss: 0.0702 - sparse_categorical_accuracy: 0.9773 - val_loss: 0.3119 - val_sparse_categorical_accuracy: 0.9188
Epoch 6/10
374/374 [==============================] - ETA: 0s - loss: 0.0744 - sparse_categorical_accuracy: 0.9763
Epoch 6: val_sparse_categorical_accuracy did not improve from 0.94434
374/374 [==============================] - 76s 203ms/step - loss: 0.0744 - sparse_categorical_accuracy: 0.9763 - val_loss: 0.3628 - val_sparse_categorical_accuracy: 0.9085
```

모델 결과및예측

각 감정별 평균 정확도

< Classification Report >

	precision	recall	f1-score	support
공포	0.95	0.96	0.95	1645
놀람	0.91	0.97	0.94	3211
분노	0.94	0.93	0.93	2708
혐오	0.97	0.95	0.96	1754
슬픔	0.98	0.89	0.93	2158
행복	0.95	0.97	0.96	2168
accuracy			0.94	13644
macro avg	0.95	0.94	0.95	13644
weighted avg	0.95	0.94	0.94	13644

문장 입력 후 감정 예측 결과

```
1  txt = '날이 좋지 않아서 우울해'
```

```
1  predict_sentiment(txt,tokenizer,model)
```

```
99.23% 확률로 슬픔 텍스트입니다.
```

```
1  txt = '오늘 하루도 화이팅'
```

```
1  predict_sentiment(txt,tokenizer,model)
```

```
99.9% 확률로 행복 텍스트입니다.
```

향후발전방향.

- 저장된 best-model을 활용하여 Flask로 웹 서버 구축
- 텍스트 입력 후 모델을 통해 감정을 분류하고 그에 맞는 멜론 플레이리스트들 추천

Thank U