감성 분석 모델 리포트

2019311195 김지유

1. 과제 소개

서울대학교 언어학과 연구진이 구축한 KOSAC(KOrean Sentiment Analysis Corpus)내의 polarity 사전을 활용하여 총 991개의 영화 '라라랜드' 리뷰에 대한 감성 분석을 진행하였다.

2. 과제 수행 과정

- 1st 영화 리뷰 데이터 가져오기

```
#영화 리뷰 데이터 가져오기
import pandas as pd
lala_review = pd.read_csv('naver_preprocess.csv',encoding='utf-8')
lala_review = lala_review.dropna()
lala_review.head()
```

이미 전처리 과정을 거친 영화 리뷰 데이터를 가져온 후, 결측값이 있는 행을 제거해 주었더니 리뷰 개수가 총 1000개에서 991개로 줄었다.

- 2nd 영화 리뷰 데이터에 라벨 달기

```
#라벨 달기
lala_review['sentiment'] = [0 if score<=5 else 1 for score in lala_review['score']]
lala_review.head(5)
```

영화 리뷰 데이터의 'score' 열을 기준으로 하여 각 리뷰에 대한 라벨(긍정/부정) 라벨을 달았다. 'score' 열의 값이 5점 이하면 부정으로, 6점 이상이면 긍정으로 라벨을 달았다. 긍정 데이터, 부정 데이터의 개수를 세어 보니 각각 914개, 77개였다. 데이터의 불균형이 큼을 알 수 있었다.

- 3rd KOSAC polarity 감성사전 데이터 가져오기

```
#감성사전 데이터 가져오기
polarity_df = pd.read_csv('polarity.csv')
polarity_df.head(10)
```

KOSAC 내의 polarity 감성사전 데이터를 가져왔다.

- 4th KOSAC polarity 감성사전 데이터를 이용해 딕셔너리 만들기

```
#감성사전 데이터를 이용해 딕셔너리 만들기
polarity_dict = {}
for i in range(len(polarity_df)):
  key = str()

  ngram = polarity_df.iloc[i][0].split(';')
  for data in ngram:

    word, tag = data.split('/')
    if word[-1] == '*':
      word = word[:-1]

    key += word + '/' + tag

  neg_score = polarity_df.iloc[i][3]
  pos_score = polarity_df.iloc[i][6]

  polarity_dict[key] = {'neg_score': neg_score, 'pos_score': pos_score}
```

KOSAC 내의 polarity 감성사전을 다듬어 사용하기 쉽게 딕셔너리 형태로 바꿔주었다. 아래 첨부한 감성사전을 살펴보면 'ngram'열의 값에 '*'와 같이 알 수 없는 기호가 붙은 경우가 존재하여, 모두 제거해주었다. 또한 polarity 감성사전이 최소 1-gram부터 최대 3-gram까지 고려하기 때문에 2-gram, 3-gram의 각 형태소 사이사이에 ';'과 같은 기호가 삽입되어 있어, 모두 제거해주었다.

	ngram	freq	COMP	NEG	NEUT	None	POS	max.value	max.prop
0	가*/JKS	1	0.0	0.000000	0.000000	0.0	1.000000	POS	1.000000
1	가*/JKS;있/VV	1	0.0	0.000000	0.000000	0.0	1.000000	POS	1.000000
2	가*/JKS;있/VV;었/EP	1	0.0	0.000000	0.000000	0.0	1.000000	POS	1.000000
3	가*/VV	3	0.0	0.000000	0.000000	0.0	1.000000	POS	1.000000
4	가*/VV;ㄴ 다*/EF	1	0.0	0.000000	0.000000	0.0	1.000000	POS	1.000000

이렇게 다듬은 'ngram'열의 값을 key로 하고, 'NEG' 열의 값과, 'POS'열의 값을 갖는 딕셔너리를 value로 갖는 아래와 같은 딕셔너리를 만들었다.

```
'가/JKS': {'neg_score': 0.46428571399999996, 'pos_score': 0.41071428600000004},
'가/JKS있/VV': {'neg_score': 0.272727273, 'pos_score': 0.545454545},
'가/JKS있/VV었/EP': {'neg_score': 0.0, 'pos_score': 1.0},
'가/VV': {'neg_score': 0.545454545, 'pos_score': 0.181818182},
'가/VVㄴ 다/EF': {'neg_score': 0.0, 'pos_score': 1.0},
```

- 5th 각 리뷰에 대한 감성 분석하기

```
kkma = Kkma()
tagged_text = kkma.pos(text)
```

조사 결과, KOSAC 감성사전과 꼬꼬마 한글 형태소 분석기의 합이 가장 좋다는 글이 많아서 위와 같이 꼬꼬마 한글 형태소 분석기를 이용해 각 리뷰(text)를 분석하였다.

```
#1-gram
for word, tag in tagged_text:
  unigram = word + '/' + tag

  score_data = polarity_dict.get(unigram, None)
  if score_data == None:
    continue

  else:
    sentiment += score_data['pos_score'] - score_data['neg_score']
    tokens_count += 1
```

```
#2-gram
for i in range(len(tagged_text)-1):
  word1, tag1 = tagged_text[i]
  word2, tag2 = tagged_text[i+1]
  bigram = word1 + '/' + tag1 + word2 + '/' + tag2

  score_data = polarity_dict.get(bigram, None)
  if score_data == None:
    continue

  else:
    sentiment += score_data['pos_score'] - score_data['neg_score']
    tokens_count += 1
```

```
#3-gram
for i in range(len(tagged_text)-2):
  word1, tag1 = tagged_text[i]
  word2, tag2 = tagged_text[i+1]
  word3, tag3 = tagged_text[i+2]
  trigram = word1 + '/' + tag1 + word2 + '/' + tag2 + word3 + '/' + tag3

  score_data = polarity_dict.get(trigram, None)
  if score_data == None:
    continue

  else:
    sentiment += score_data['pos_score'] - score_data['neg_score']
    tokens_count += 1
```

각 리뷰(text)에 대한 형태소 분석 후, 위와 같이 1-gram, 2-gram, 3-gram을 모두 구하였다. 구한 n-gram이 감성사전에 있을 경우 긍정 점수에서 부정 점수를 뺀 값을 감성 점수(sentiment)에 더하는 방식으로 각 리뷰에 대한 감성점수를 산출했다. 산출된 감성 점수가 0점 이상인 리뷰를 긍정으로 분류하고, 산출된 감성 점수가 0 미만인 리뷰를 부정으로 분류하였다. 만약, 리뷰를 이루고 있는 형태소가 감성사전에 없어서 감성 점수가 계산되지 않았다면 그 리뷰는 긍정으로 분류하였다.

- 6th 감성 분석(분류) 결과 평가하기

	precision	recall	f1-score	support
0	0.11	0.36	0.17	77
1	0.93	0.75	0.83	914
accuracy			0.72	991
macro avg	0.52	0.56	0.50	991
weighted avg	0.87	0.72	0.78	991

sklearn에서 제공하고 있는 classification_report를 이용해 감성 분석 결과를 확인한 결과, accuracy는 72%가 나왔다. 위에서 살펴보았듯이 해당 리뷰 데이터의 라벨이 굉장히 불균형하기 때문에 accuracy는 정확한 평가 지표가 될 수 없다. 긍정으로 분류된 리뷰 중 실제로 긍정인 리뷰의 비율은 93%로 굉장히 높았다. 반면, 실제로 긍정인 리뷰 중에 긍정으로 분류된 리뷰의 비율은 75%로 상대적으로 낮았다. precision과 recall의 비율은 어느 정도 tradeoff이기 때문에 precision은 상대적으로 높은 반면, recall은 상대적으로 낮게 나온 것 같다. precision과 recall의 조화 평균인 f1-score는 83%로 과제를 수행하기 전에 예상했던 것보다 괜찮은 수치를 보였다.

3. 한계점

- 첫 번째 한계점

  KOSAC내의 polarity 감성사전에서 사용하고 있는 품사 태그와 꼬꼬마 한글 형태소 분석기에서 사용하고 있는 품사 태그가 몇몇 품사에 대해서 다르다. 예를 들어 관형사형 전성 어미에 대한 태그로 감성사전은 'ETM'을 사용하고 있는 반면, 꼬꼬마 한글 형태소 분석기는 'ETD'를 사용하고 있다.

- 두 번째 한계점

  KOSAC 내의 polarity 감성사전에서 분류한 형태소와 꼬꼬마 한글 형태소 분석기에서 분류한 형태소가 몇몇 단어에 대해서 다르다. 예를 들어, '재밌습니다'에 대한 형태소 분석을 감성사전은 '재미있/VA'으로 하는 반면, 꼬꼬마 한글 형태소 분석기는 '재밌/VA'으로 한다.

  위의 두 한계점으로 인해 감성사전을 이용하여 감성 점수가 분석되지 않는 경우가 종종 존재해 모델의 성능을 저하시켰다.