<< 네이버 영화 데이터 긍정 부정 판단 모델링 >>

네이버영화 메인 페이지 1~10위 영화 리뷰, 평점 크롤링 후 감성분석 모델링 만들어서 리뷰 입력하면 긍정,부정 판단하기

충북대학교 김진용 박지수 이주언 장희주

https://movie.naver.com/movie/running/current.nhn (https://movie.naver.com/movie/running/current.nhn)
https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=174903&type=after&onlyActualPointYn=N&order=sympathyScore&page=2 (https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=174903&type=after&onlyActualPointYn=N&order=sympathyScore&page=2)

전체 과정

- step1. 네이버 영화 메인 페이지(현재 상영작) 에서 1~10위 영화의 상세페이지 주소 크롤링
- step2. 상세페이지에서 평점의 더보기를 클릭했을 때 보여지는 페이지 주소 크롤링
- step3. 140자 평을 1페이지~끝 페이지 순회하면서 평점과 리뷰를 크롤링
- step4. 학습을 위한 데이터 전처리
- step5. 학습. 학습된 모델을 파일로 저장
- step6. 저장된 모델을 불러와 사용

비고 : 사전을 만들고 보다 정확한 분석을 하고자 했지만 시간적인 제약등 여러 문제점때문에 포기하고 train 데이터로 학습을 시키기로 했습니다. 평점과 리뷰를 매치하여 "평점을 잘 준사람은 리뷰도 긍정적으로 썼을 것이다" 라는 전제 하에 분석을 진행했습니다.

step0. import

In [1]:
```
import requests # http에 접속 요청 처리
from bs4 import BeautifulSoup #웹 크롤링에 필수
from time import sleep #운영 체제의 시간 관련 기능을 다룸
import pandas as pd #dataframe 생성
import os #directory 관련
from urllib import parse #웹상의 문서나 파일을 가져올 수 있게함
```

step1. 네이버 영화 메인 페이지(현재 상영작) 에서 1~10위 영화의 상세페이지 주소 크롤링,저장

In [2]:
```
def step1_get_detail_url() :
    # 접속할 페이지의 주소 네이버 영화 메인 페이지
    site = 'https://movie.naver.com/movie/running/current.nhn?order=reserve'

    # requests를 이용해 해당 URL에 접속한다
    response = requests.get(site)

    # 영화 페이지를 크롤링한다
    bs = BeautifulSoup(response.content, 'html.parser')

    # a 태그들을 가져온다.
    a_list = bs.select('.top_thumb_lst a')

    # href 속성을 가져온다.
    df = pd.DataFrame()
    for idx in range(10) :       # 상위 10개만 가져오기
        href = a_list[idx].get('href')

        # 가져온 href 속성의 주소를 분석한 객체를 생성한다.
        a1 = parse.urlparse(href)

        # 주소를 분석한 객체서 쿼리 스트링을 가져온다(? 이후)
        query_str = parse.parse_qs(a1.query)

        # 추출한 쿼리스트링 데이터에서 원하는 파라미터 데이터를 추출한다.
        code = query_str['code'][0]
        print(code)

        df = df.append([[code]], ignore_index=True)

    df.columns = ['code'] #추출한 10개 영화 코드를 저장한다.
    df.to_csv('movie_code_list.csv', index=False, encoding='utf-8-sig') #코드를 CSV
로 저장
    print('주소 저장 완료')
step1_get_detail_url()
```

```
174903
182355
180351
178526
167653
140656
181694
180374
154672
163788
주소 저장 완료
```

step2. 상세페이지에서 평점의 더보기를 클릭했을 때 보여지는 페이지 주소 크롤링,저장

In [3]:
```
def step2_get_reple_href() :
    # 스크랩한 영화 코드를 불러온다.
    code_frame = pd.read_csv('movie_code_list.csv')
    code_list = code_frame['code'].tolist()

    #테스트용
    #code_list = ['174903', '182355']

    # 영화코드와 주소를 합쳐서 요청할 주소를 만든다.
    url_list = pd.DataFrame()
    for code in code_list: #코드가 들어갈 부분을 %s로 놓고 10개의 코드를 반복하여 u
rl을 만든다
        site = 'https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=%s&
type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMil
eageSubscriptionReject=false' % code
        url_list = url_list.append([[site]], ignore_index=True)

    url_list.columns = ['url']
    url_list.to_csv('movie_url_list.csv', index=False, encoding='utf-8-sig') #평점
과 리뷰가 있는 영화의 상세 페이지 URL을 저장한다
    print('저장완료')
step2_get_reple_href()
```

저장완료

step3. 140자 평을 1페이지~끝 페이지 순회하면서 평점과 리뷰를 크롤링,저장

In [4]:
```
def step3_get_reply_data() :
    # csv에 저장되어 있는 url 데이터를 가져온다.
    df = pd.read_csv('movie_url_list.csv')
    url_list = df['url'].tolist()

    for url in url_list :
        print(url)
        # 해당 영화의 첫 페이지 html 데이터를 가져온다. (총 몇건의 리뷰가 있는지 확인해서 "페이지수"를 계산하기 위함)
        response = requests.get(url)
        bs = BeautifulSoup(response.content, 'html.parser') #10개의 영화의 평점,리뷰 페이지를 크롤링한다

        # 총 페이지 수를 구한다.
        strong = bs.select('.total em')
        score_total = int(strong[1].text.replace(',', ''))    # 쉼표 없애기 / int(정수형)로 만들기

        pageCnt = score_total // 10     # 한페이지당 10개의 리뷰가 있어서
        if score_total % 10 > 0 :
            pageCnt += 1

        # 전체 페이지를 돌면서 140평 데이터를 가져온다.
        # 현재 페이지
        now_page = 1

        pageCnt = 50        # 일단 테스트로 10페이지까지만 했습니다. 전부 하면 2000개가 넘는 페이지가 있어서 잘 안돌아감
        while now_page <= pageCnt :
            sleep(1)
            # 요청할 페이지의 주소
            url2 = url + '&page=' + str(now_page)

            # 140자평 데이터를 추출한다.
            response2 = requests.get(url2)
            bs2 = BeautifulSoup(response2.content, 'html.parser')

            result_df = pd.DataFrame()

            # li 태그들을 가져온다.(score_reple 태그-리뷰-를 포함하고 있는)
            lis = bs2.select('.score_result li')

            for obj in lis :
                # 평점
                star_score = obj.select('.star_score em')[0].text
                # 140자평
                score_reple = obj.select('.score_reple p')[0].text
                # 저장한다.
                result_df = result_df.append([[score_reple, star_score]], ignore_index=True)

            if os.path.exists('star_score.csv') == False :     # 아직 파일이 없으면 파일을 만든다. 평점과 평가를 feature로 하는 dataframe을 저장
                result_df.columns = ['text', 'score']
                result_df.to_csv('star_score.csv', index=False, encoding='utf-8-sig')
```

```
            else :                                                  # 이미 파일이 있으
면 결과를 더한다.
                result_df.to_csv('star_score.csv', index=False, encoding='utf-8-si
g', mode='a', header=False)

            print("%d / %d" % (now_page, pageCnt))                  # 진행경과를 보여
준다. n번째 중 몇 번째 진행중인지.
            now_page += 1

    print('저장완료')
step3_get_reply_data()
```

https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=174903&type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMileageSubscriptionReject=false
1 / 50
2 / 50
3 / 50
4 / 50
5 / 50
6 / 50
7 / 50
8 / 50
9 / 50
10 / 50
11 / 50
12 / 50
13 / 50
14 / 50
15 / 50
16 / 50
17 / 50
18 / 50
19 / 50
20 / 50
21 / 50
22 / 50
23 / 50
24 / 50
25 / 50
26 / 50
27 / 50
28 / 50
29 / 50
30 / 50
31 / 50
32 / 50
33 / 50
34 / 50
35 / 50
36 / 50
37 / 50
38 / 50
39 / 50
40 / 50
41 / 50
42 / 50
43 / 50
44 / 50
45 / 50
46 / 50
47 / 50
48 / 50
49 / 50
50 / 50
https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=182355&type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMileageSubscriptionReject=false
1 / 50

2 / 50
3 / 50
4 / 50
5 / 50
6 / 50
7 / 50
8 / 50
9 / 50
10 / 50
11 / 50
12 / 50
13 / 50
14 / 50
15 / 50
16 / 50
17 / 50
18 / 50
19 / 50
20 / 50
21 / 50
22 / 50
23 / 50
24 / 50
25 / 50
26 / 50
27 / 50
28 / 50
29 / 50
30 / 50
31 / 50
32 / 50
33 / 50
34 / 50
35 / 50
36 / 50
37 / 50
38 / 50
39 / 50
40 / 50
41 / 50
42 / 50
43 / 50
44 / 50
45 / 50
46 / 50
47 / 50
48 / 50
49 / 50
50 / 50
https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=180351&type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMileageSubscriptionReject=false
1 / 50
2 / 50
3 / 50
4 / 50
5 / 50

6 / 50
7 / 50
8 / 50
9 / 50
10 / 50
11 / 50
12 / 50
13 / 50
14 / 50
15 / 50
16 / 50
17 / 50
18 / 50
19 / 50
20 / 50
21 / 50
22 / 50
23 / 50
24 / 50
25 / 50
26 / 50
27 / 50
28 / 50
29 / 50
30 / 50
31 / 50
32 / 50
33 / 50
34 / 50
35 / 50
36 / 50
37 / 50
38 / 50
39 / 50
40 / 50
41 / 50
42 / 50
43 / 50
44 / 50
45 / 50
46 / 50
47 / 50
48 / 50
49 / 50
50 / 50
https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=178526&type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMileageSubscriptionReject=false
1 / 50
2 / 50
3 / 50
4 / 50
5 / 50
6 / 50
7 / 50
8 / 50
9 / 50

10 / 50
11 / 50
12 / 50
13 / 50
14 / 50
15 / 50
16 / 50
17 / 50
18 / 50
19 / 50
20 / 50
21 / 50
22 / 50
23 / 50
24 / 50
25 / 50
26 / 50
27 / 50
28 / 50
29 / 50
30 / 50
31 / 50
32 / 50
33 / 50
34 / 50
35 / 50
36 / 50
37 / 50
38 / 50
39 / 50
40 / 50
41 / 50
42 / 50
43 / 50
44 / 50
45 / 50
46 / 50
47 / 50
48 / 50
49 / 50
50 / 50
https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=167653&type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMileageSubscriptionReject=false
1 / 50
2 / 50
3 / 50
4 / 50
5 / 50
6 / 50
7 / 50
8 / 50
9 / 50
10 / 50
11 / 50
12 / 50
13 / 50

14 / 50
15 / 50
16 / 50
17 / 50
18 / 50
19 / 50
20 / 50
21 / 50
22 / 50
23 / 50
24 / 50
25 / 50
26 / 50
27 / 50
28 / 50
29 / 50
30 / 50
31 / 50
32 / 50
33 / 50
34 / 50
35 / 50
36 / 50
37 / 50
38 / 50
39 / 50
40 / 50
41 / 50
42 / 50
43 / 50
44 / 50
45 / 50
46 / 50
47 / 50
48 / 50
49 / 50
50 / 50
https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=140656&type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMileageSubscriptionReject=false
1 / 50
2 / 50
3 / 50
4 / 50
5 / 50
6 / 50
7 / 50
8 / 50
9 / 50
10 / 50
11 / 50
12 / 50
13 / 50
14 / 50
15 / 50
16 / 50
17 / 50

18 / 50
19 / 50
20 / 50
21 / 50
22 / 50
23 / 50
24 / 50
25 / 50
26 / 50
27 / 50
28 / 50
29 / 50
30 / 50
31 / 50
32 / 50
33 / 50
34 / 50
35 / 50
36 / 50
37 / 50
38 / 50
39 / 50
40 / 50
41 / 50
42 / 50
43 / 50
44 / 50
45 / 50
46 / 50
47 / 50
48 / 50
49 / 50
50 / 50
https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=181694&type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMileageSubscriptionReject=false
1 / 50
2 / 50
3 / 50
4 / 50
5 / 50
6 / 50
7 / 50
8 / 50
9 / 50
10 / 50
11 / 50
12 / 50
13 / 50
14 / 50
15 / 50
16 / 50
17 / 50
18 / 50
19 / 50
20 / 50
21 / 50

22 / 50
23 / 50
24 / 50
25 / 50
26 / 50
27 / 50
28 / 50
29 / 50
30 / 50
31 / 50
32 / 50
33 / 50
34 / 50
35 / 50
36 / 50
37 / 50
38 / 50
39 / 50
40 / 50
41 / 50
42 / 50
43 / 50
44 / 50
45 / 50
46 / 50
47 / 50
48 / 50
49 / 50
50 / 50
https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=180374&type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMileageSubscriptionReject=false
1 / 50
2 / 50
3 / 50
4 / 50
5 / 50
6 / 50
7 / 50
8 / 50
9 / 50
10 / 50
11 / 50
12 / 50
13 / 50
14 / 50
15 / 50
16 / 50
17 / 50
18 / 50
19 / 50
20 / 50
21 / 50
22 / 50
23 / 50
24 / 50
25 / 50

26 / 50
27 / 50
28 / 50
29 / 50
30 / 50
31 / 50
32 / 50
33 / 50
34 / 50
35 / 50
36 / 50
37 / 50
38 / 50
39 / 50
40 / 50
41 / 50
42 / 50
43 / 50
44 / 50
45 / 50
46 / 50
47 / 50
48 / 50
49 / 50
50 / 50
https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=154672&type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMileageSubscriptionReject=false
1 / 50
2 / 50
3 / 50
4 / 50
5 / 50
6 / 50
7 / 50
8 / 50
9 / 50
10 / 50
11 / 50
12 / 50
13 / 50
14 / 50
15 / 50
16 / 50
17 / 50
18 / 50
19 / 50
20 / 50
21 / 50
22 / 50
23 / 50
24 / 50
25 / 50
26 / 50
27 / 50
28 / 50
29 / 50

30 / 50
31 / 50
32 / 50
33 / 50
34 / 50
35 / 50
36 / 50
37 / 50
38 / 50
39 / 50
40 / 50
41 / 50
42 / 50
43 / 50
44 / 50
45 / 50
46 / 50
47 / 50
48 / 50
49 / 50
50 / 50
https://movie.naver.com/movie/bi/mi/pointWriteFormList.nhn?code=163788&type=after&isActualPointWriteExecute=false&isMileageSubscriptionAlready=false&isMileageSubscriptionReject=false
1 / 50
2 / 50
3 / 50
4 / 50
5 / 50
6 / 50
7 / 50
8 / 50
9 / 50
10 / 50
11 / 50
12 / 50
13 / 50
14 / 50
15 / 50
16 / 50
17 / 50
18 / 50
19 / 50
20 / 50
21 / 50
22 / 50
23 / 50
24 / 50
25 / 50
26 / 50
27 / 50
28 / 50
29 / 50
30 / 50
31 / 50
32 / 50
33 / 50

```
34 / 50
35 / 50
36 / 50
37 / 50
38 / 50
39 / 50
40 / 50
41 / 50
42 / 50
43 / 50
44 / 50
45 / 50
46 / 50
47 / 50
48 / 50
49 / 50
50 / 50
저장완료
```

In [5]:
```python
# 140자평 데이터 전처리 함수
def text_preprocessing(text) :
    if text.startswith('관람객') :
        return text[3:]
    else :
        return text

# 평점 전처리 함수
def star_preprocessing(text) :
    value = int(text)

    if value <= 7 :
        return '0'
    else :
        return '1'
```

step4. 학습을 위해 데이터 전처리를 수행한다.

In [6]:
```
def step4_data_preprocessing() :
    # 수집한 데이터를 읽어온다.
    df = pd.read_csv('star_score.csv')

    # 전처리를 수행한다.
    df['text'] = df['text'].apply(text_preprocessing)
    df['score'] = df['score'].apply(star_preprocessing)

    # 학습데이터와 테스트 데이터로 나눈다.
    text_list = df['text'].tolist()
    star_list = df['score'].tolist()

    from sklearn.model_selection import train_test_split

    # 70%는 학습, 30%는 test
    text_train, text_test, star_train, star_test = train_test_split(text_list, star
_list, test_size=0.3, random_state=0)

    return text_train, text_test, star_train, star_test


from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn.metrics import accuracy_score
import pickle
from konlpy.tag import *
```

형태소 분석 함수

In [7]:
```
# 형태소 분석을 위한 함수
def tokenizer(text) :
    okt = Okt()
    return okt.morphs(text)
```

step5. 학습을 한다. 학습이 완료된 모델을 파일로 저장한다.

In [8]:
```
def step5_learning(X_train, y_train, X_test, y_test):
    # 주어진 데이터를 단어 사전으로 만들고 각 단어의 빈도수를 계산한 후 벡터화 하는 객체 생성
    tfidf = TfidfVectorizer(lowercase=False, tokenizer=tokenizer)

    # 문장별 나오는 단어수 세서 수치화, 벡터화해서 학습을 시킨다. 회귀분석 이용
    logistic = LogisticRegression(C=10.0, penalty='l2', random_state=0)

    pipe = Pipeline([('vect', tfidf), ('clf', logistic)])

    # 학습한다.
    pipe.fit(X_train, y_train)

    # 학습 정확도 측정
    y_pred = pipe.predict(X_test)
    print(accuracy_score(y_test, y_pred))

    # 학습된 모델을 저장한다.
    with open('pipe.dat', 'wb') as fp :
        pickle.dump(pipe, fp)

    print('저장완료')
```

step6. 저장된 모델을 불러와 사용한다.

In [9]:
```
def step6_using_model() :
    # 객체를 복원한다.
    with open('pipe.dat', 'rb') as fp:
        pipe = pickle.load(fp)

    import numpy as np

    for num in [1,2,3,4,5,6] :
        text = input('리뷰를 작성해주세요 :')

        str = [text]
        # 예측 정확도
        r1 = np.max(pipe.predict_proba(str) * 100)
        # 예측 결과
        r2 = pipe.predict(str)[0]

        if r2 == '1' :
            print('긍정적인 리뷰')
        else :
            print('부정적인 리뷰')

        print('정확도 : %.3f' % r1)
```

In [10]:
```
# 스크래핑 함수
def scrapping() :

    # 각 영화 코드 데이터를 가져와 저장한다.
    # step1_get_detail_url()

    # 140자 평 데이터가 있는 페이지의 주소를 저장한다.
    step2_get_reple_href()

    # 140평 데이터를 가져온다.
    step3_get_reply_data()

# 학습 함수
def learing() :
    text_train, text_test, star_train, star_test = step4_data_preprocessing()
    step5_learning(text_train, star_train, text_test, star_test)

# 사용 함수
def using() :
    step6_using_model()
```

In [11]:
```
#scrapping()
learing()
using()
```

```
C:₩Users₩jisu₩Miniconda3₩envs₩emotion-analysis₩lib₩site-packages₩jpype₩_core.py:210:
UserWarning:
-------------------------------------------------------------------------------
Deprecated: convertStrings was not specified when starting the JVM. The default
behavior in JPype will be False starting in JPype 0.8. The recommended setting
for new code is convertStrings=False.  The legacy value of True was assumed for
this session. If you are a user of an application that reported this warning,
please file a ticket with the developer.
-------------------------------------------------------------------------------


  """)
C:₩Users₩jisu₩Miniconda3₩envs₩emotion-analysis₩lib₩site-packages₩sklearn₩linear_mode
l₩logistic.py:432: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22.
Specify a solver to silence this warning.
  FutureWarning)

0.9670555936856554
저장완료

긍정적인 리뷰
정확도 : 99.775

긍정적인 리뷰
정확도 : 99.342

부정적인 리뷰
정확도 : 95.986

부정적인 리뷰
정확도 : 96.471

긍정적인 리뷰
정확도 : 89.162

부정적인 리뷰
정확도 : 98.144
```

0.9670555936856554

저장완료

리뷰를 작성해주세요 : 완전 웃겨요 ㅋㅋㅋㅋ 최근에 본 것 중에 제일 재밌음 추천!

긍정적인 리뷰

정확도 : 99.775

리뷰를 작성해주세요 : ㅋㅋㅋㅋ 조정석 개 웃김 꿀잼

긍정적인 리뷰

정확도 : 99.342

리뷰를 작성해주세요 : 솔직히 왜 사람들 많이 보는지 모르겠다 난 노잼

부정적인 리뷰

정확도 : 95.986

리뷰를 작성해주세요 : 감독 수준 봐라 억지 눈물 쥐어짜내네

부정적인 리뷰

정확도 : 96.471

리뷰를 작성해주세요 : 공짜로 볼 수 있어서 봄 영화는 재밌긴한데 내 돈 내고는 안 볼 듯

긍정적인 리뷰

정확도 : 89.162

리뷰를 작성해주세요 : 배우들 연기는 괜찮은데 시나리오는 좀 억지스러움 개연성도 떨어지고..

부정적인 리뷰

정확도 : 98.144

In [ ]: