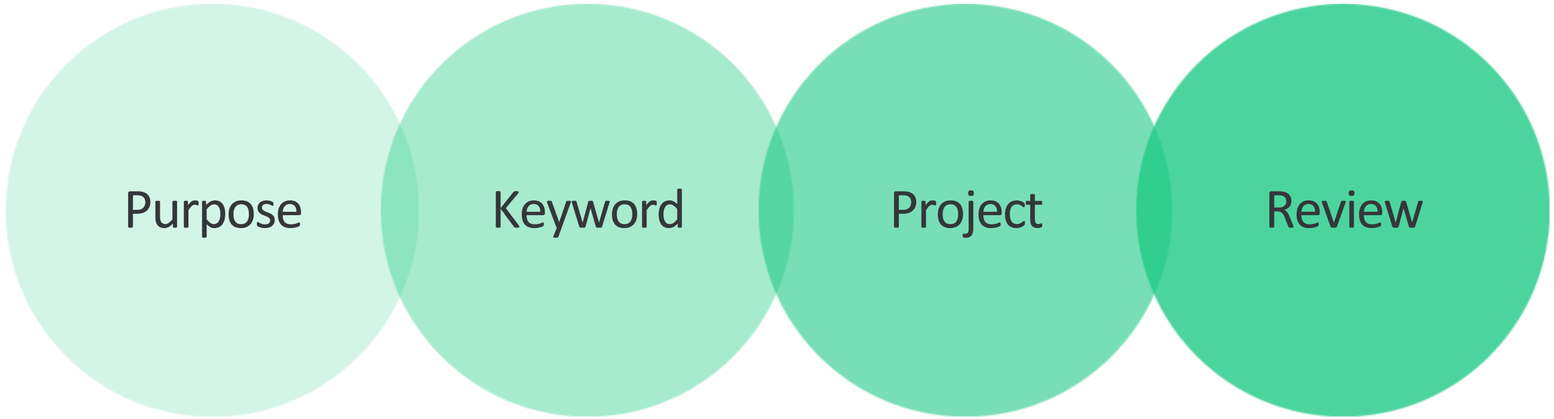
NLP Mini Project

네이버 영화 추천 시스템

김성민 이이삭 주하영

INDEX



Purpose

네이버 영화 추천 시스템

TF-IDF와 코사인 유사도로 영화의 줄거리에 기반해서 영화를 추천한다.

TF-IDF

TF(단어 빈도) 특정 문서 d에서의 특정 단어 t의 등장 횟수

DF(문서 빈도)특정 단어 t가 등장한 문서의 수

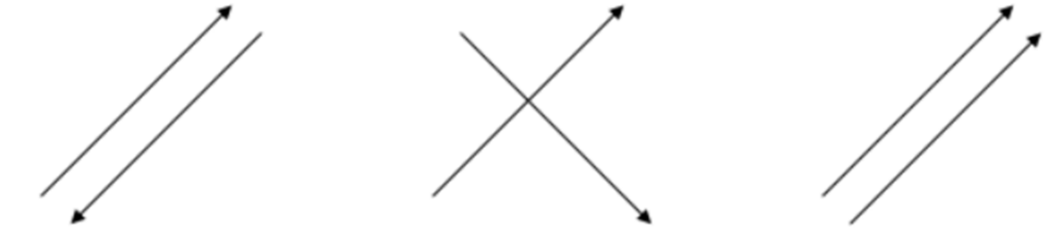
IDF(역문서 빈도)는 DF의 역수

TF-IDF는 TF와 IDF를 곱한 값으로 다른 문서에 자주 언급되지 않고
해당 문서에는 자주 언급되는 token에 대해 점수를 높게 부여한다.

1) https://wikidocs.net/31698

코사인 유사도

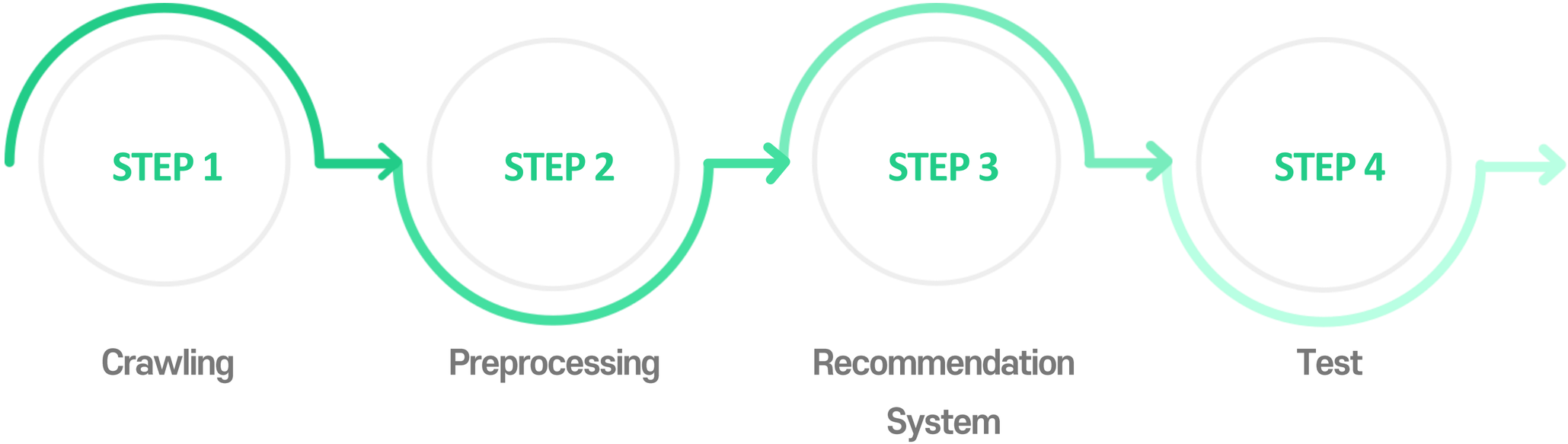
코사인 유사도는 두 벡터 간의 코사인 각도를 이용하여 구할 수 있는 **두 벡터의 유사도**를 의미한다.
두 벡터의 방향이 완전히 동일한 경우에는 1의 값을 가지며, 90°의 각을 이루면 0,
180°로 반대의 방향을 가지면 -1의 값을 갖게 된다.
코사인 유사도는 -1 이상 1 이하의 값을 가지며 **값이 1에 가까울수록 유사도가 높다고** 판단할 수 있다.



코사인 유사도 : -1　　코사인 유사도 : 0　　코사인 유사도 : 1

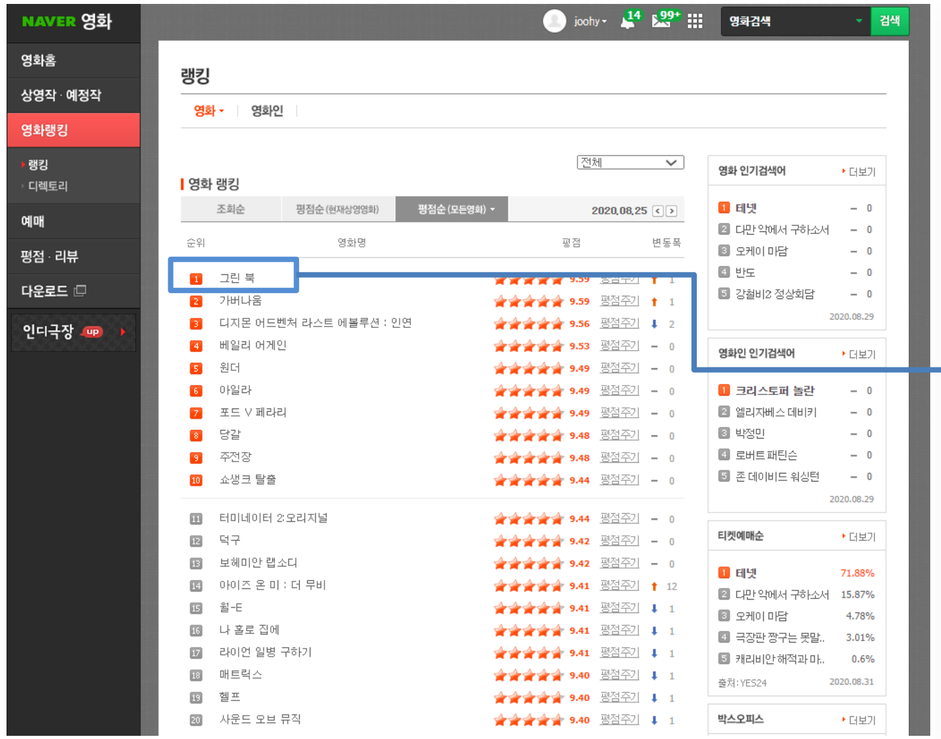
1) https://ko.wikipedia.org/wiki/코사인_유사도

NLP Project Process



Project

STEP 1 Crawling



네이버 영화 크롤링

영화랭킹 페이지의 평점순(모든 영화)
1000개의 영화 Data

Title Genre Story

Project

STEP 1 Crawling

```
import requests
import csv
from bs4 import BeautifulSoup

base_url = 'https://movie.naver.com/movie/sdb/rank/rmovie.nhn?sel=pnt&date=20200825&tg=0&page='

movie_url_list = []
name_list = []
genre_list = []
story_list = []

for num in range(1, 21) :
    page_num = num
    url = base_url + str(page_num)

    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')

    name_div = soup.select('.tit5')

    for name in name_div :
        title = name.select_one('a')['title']
        href = name.select_one('a')['href']

        name_list.append(title)
        movie_url_list.append(href)

print('done')
```

Project

STEP 1 Crawling

```
i = 0
for movie_url in movie_url_list :
    url = "https://movie.naver.com/" + movie_url
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')

    if soup.find('dl') == None :
        i += 1
        genre = ''
        print(genre, i)
    else :
        genre_a_tag = soup.find('dl', class_="info_spec").find('span').find('a')
        genre = genre_a_tag.text
        i += 1
        print(genre, i)

    if soup.find('p',class_="con_tx") == None :
        story = ''
    else :
        story_p_tag = soup.find('p',class_="con_tx")
        story = story_p_tag.text



    genre_list.append(genre)
    story_list.append(story)

print('done')
```

STEP 1 Crawling

```
with open('./movie.csv', 'a', encoding='utf-8') as csvfile:
        fieldnames = ['title', 'genre', 'story']
        writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
        writer.writerow({
            'title' : 'title',
            'genre' : 'genre',
            'story' : 'story'
        })

for i in range(len(name_list)) :
    with open('./movie.csv', 'a', encoding='utf-8') as csvfile:
        fieldnames = ['title', 'genre', 'story']
        writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
        writer.writerow({
            'title' : name_list[i],
            'genre' : genre_list[i],
            'story' : story_list[i]
        })
```

STEP 2 Preprocessing

```
movies = data.loc[:,['title','story']]
print(pd.DataFrame(movies))
```

```
                            title                                                story
0                            그린 북  1962년 미국, 입담과 주먹만 믿고 살아가던 토니 발레롱가(비고 모텐슨)는 교양과...
1                           가버나움  나를 세상에 태어나게 한 ₩r "부모님을 고소하고 싶어요..."₩r -출생기록조차 ...
2          디지몬 어드벤처 라스트 에볼루션 : 인연  디지몬 어드벤처 20주년 기념 극장판!₩r "우리들의 끈끈한 '인연'을 보여주자."...
3                         베일리 어게인  귀여운 소년  '이든'의 단짝 반려견  '베일리'는 행복한 생을 마감한다.₩r 하지만 ...
4                            원더  누구보다 위트 있고 호기심 많은 매력 부자  '어기'(제이콥 트렘블레이). 하지만 남...
..                            ...                                                ...
995                          히노키오  원격조종 로봇을 통한 대리등교를 허용하는 법안이 통과된 후, 교통사고로 어머니를 잃...
996                         언노운 우먼  젊고 부유한 보석 세공사인 아다처 부인의 집에 들어가기 위해 기존의 가정부를 사고로...
997                         카사블랑카  중동에 위치한 요지, 모로코의 카사브랑카는 전란을 피하여 미국으로 가려는 사람들의 ...
998                        죽여주는 여자  종로 일대에서 노인들을 상대하며 근근이 살아가는 65세의  '박카스 할머니'  소영. ...
999  인피니트 콘서트 세컨드 인베이전 에볼루션 더 무비 3D  4월 1일, 서울 올림픽공원 체조경기장에서 열린 인피니트의 앵콜 콘서트  'SECON...

[1000 rows x 2 columns]
```

크롤링 한 Raw Data

네이버 영화 1000개의 title과 story

STEP 2 Preprocessing

```
#정규표현식으로 필요없는 글자를 제거해준다
movies['keyword'] = movies['story']
for i in range(len(data.story)-1):
    movies['keyword'][i] =  re.sub(r'[^가-힣0-9a-zA-Z₩s]','', str(movies['story'][i]))
print(pd.DataFrame(movies))
```

```
                          title  ...                                            keyword
0                          그린 북  ...  1962년 미국 입담과 주먹만 믿고 살아가던 토니 발레롱가비고 모텐슨는 교양과 우아...
1                         가버나움  ...  나를 세상에 태어나게 한 ₩r 부모님을 고소하고 싶어요₩r 출생기록조차 없이 살아온...
2          디지몬 어드벤처 라스트 에볼루션 : 인연  ...  디지몬 어드벤처 20주년 기념 극장판₩r 우리들의 끈끈한 인연을 보여주자₩r ₩r ...
3                        베일리 어게인  ...  귀여운 소년 이든의 단짝 반려견 베일리는 행복한 생을 마감한다₩r 하지만 눈을 떠보...
4                           원더  ...  누구보다 위트 있고 호기심 많은 매력 부자 어기제이콥 트렘블레이 하지만 남들과 다른...
..                         ...  ...                                            ...
995                       히노키오  ...  원격조종 로봇을 통한 대리등교를 허용하는 법안이 통과된 후 교통사고로 어머니를 잃어...
996                      언노운 우먼  ...  젊고 부유한 보석 세공사인 아다처 부인의 집에 들어가기 위해 기존의 가정부를 사고로...
997                      카사블랑카  ...  중동에 위치한 요지 모로코의 카사브랑카는 전란을 피하여 미국으로 가려는 사람들의 기...
998                      죽여주는 여자  ...  종로 일대에서 노인들을 상대하며 근근이 살아가는 65세의 박카스 할머니 소영 노인들...
999  인피니트 콘서트 세컨드 인베이전 에볼루션 더 무비 3D  ...  4월 1일, 서울 올림픽공원 체조경기장에서 열린 인피니트의 앵콜 콘서트 'SECON...

[1000 rows x 3 columns]
```

정규표현식

필요 없는 글자(한글,숫자,영어 외) 제거

STEP 2 Preprocessing

```
# 스톱워드를 설정하여 해석에 필요없는 글자들을 제거해준다
token_ls = []
STOP_WORDS = ['의','가','이','은','들','는','좀','잘','걍','과','도','를','으로','자','에','와','한','하다',',',"'",'ㅠr']

for i in range(len(movies)):
  token_ls = []
  for j in okt.morphs(movies['keyword'][i], stem=True):
    if j not in STOP_WORDS:
      token_ls.append(j)
  movies['story'][i] = token_ls
```

STOPWORD 제거

의미 없는 STOPWORD를 정의하고 제거

STEP 2 Preprocessing

```
#영화 제목과 스토리를 정제한 단어들의 모음 re_story를 movie에 담는다
movie = movies.loc[:,['title','re_story']]
print(movie)
```

```
                          title                                           re_story
0                          그린 북  1962년 미국 입당 주먹 만 믿다 살아가다 토니 발레 롱 비고 모텐슨는 교양 우아...
1                         가버나움   나르다 세상 태어나다 고소하다 싶다 없이 살아오다 어쩌면 12 살 소년 자인 으로부터
2          디지몬 어드벤처 라스트 에볼루션 : 인연  디지몬 어드벤처 20 주년 기념 극장판 끈끈 인연 을 보여주다 타 이치 다른 선택 ...
3                       베일리 어게인  귀엽다 소년 든 단짝 반려견 베 일리 행복하다 생 을 마감 눈 을 떠보다 다시 시작...
4                          원더  누구 보다 위트 있다 호기심 많다 매력 부자 어기 제이콥 트렘블 레이   하지만 남...
..                          ...                                             ...
995                        히노키오  원 격조 종 로봇 을 통한 대리 등교 허용 법안 통과 되다 후 교통사고 로 어머니 ...
996                       언노운 우먼  젊다 부유하다 보석 세공 사인 아 다처 부인 집 들어가다 위해 기존 가정부 사고 로...
997                       카사블랑카  중동 위치 요지 모로코 카 사브 랑 카 전란 을 피하 여 미국 가다 사람 기항지 로...
998                      죽여주는 여자  종로 일대 에서 노인 을 상대 근 근 살아가다 65 세 박카스 할머니 소영 노 인들...
999  인피니트 콘서트 세컨드 인베이전 에볼루션 더 무비 3D  4월 1일 서울 올림픽 공원 체조 경기장 에서 열리다 인피니트 앵콜 콘서트  ' SE...
```

전처리 DATA

토큰화 한 data를 movie에 저장

STEP 3 Recommendation System

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer()
tfidf_matrix = tfidf.fit_transform(movie.re_story)
# 줄거리에 대해서 tf-idf 수행
print(tfidf_matrix.shape)
```

TF-IDF

각 단어에 대한 중요도를 계산하여 영화를
비교하기 위해 줄거리에 대해 TF-IDF를 수행

STEP 3 Recommendation System

```
"""코사인 유사도 구하기"""
from sklearn.metrics.pairwise import linear_kernel
cosine_sim = linear_kernel(tfidf_matrix, tfidf_matrix)
cosine_sim
```

```
array([[1.        , 0.04570757, 0.02268699, ..., 0.02911174, 0.0215305 ,
        0.01975203],
       [0.04570757, 1.        , 0.        , ..., 0.        , 0.04647326,
        0.        ],
       [0.02268699, 0.        , 1.        , ..., 0.0081705 , 0.02040378,
        0.01072493],
       ...,
       [0.02911174, 0.        , 0.0081705 , ..., 1.        , 0.01873337,
        0.01106293],
       [0.0215305 , 0.04647326, 0.02040378, ..., 0.01873337, 1.        ,
        0.00557773],
       [0.01975203, 0.        , 0.01072493, ..., 0.01106293, 0.00557773,
        1.        ]])
```

코사인 유사도

코사인 유사도로 영화 간의 유사도 구하기

STEP 3 Recommendation System

```
def movie_REC(title, cosine_sim=cosine_sim):
    #입력한 영화로 부터 인덱스 가져오기
    idx = indices[title]
    print(idx)

    # 모든 영화에 대해서 해당 영화와의 유사도를 구하기
    sim_scores = list(enumerate(cosine_sim[idx]))
    print(sim_scores)
    # 유사도에 따라 영화들을 정렬
    sim_scores = sorted(sim_scores, key=lambda x:x[1], reverse = True)
    print(sim_scores)
    # 가장 유사한 10개의 영화를 받아옴
    sim_scores = sim_scores[1:11]
    print(sim_scores)
    # 가장 유사한 10개 영화의 인덱스 받아옴
    movie_indices = [i[0] for i in sim_scores]
    print(movie_indices)
    #기존에 읽어들인 데이터에서 해당 인덱스의 값들을 가져온다. 그리고 스코어 열을 추가하여 코사인 유사도도 확인할 수 있게 한다.
    result_df = movie.iloc[movie_indices].copy()
    result_df['score'] = [i[1] for i in sim_scores]

    # 읽어들인 데이터에서 줄거리 부분만 제거, 제목과 스코어만 보이게 함
    del result_df['re_story']

    # 가장 유사한 10개의 영화의 제목을 리턴
    return result_df
```

Story가 유사한 영화를 찾는 함수

선택한 영화의 코사인 유사도를 이용하여

가장 story가 유사한 10개의 영화를 찾는 함수

STEP 4 TEST

```
movie_REC('아이언맨')
```

663

	title	score
757	아이언맨 3	0.286218
229	캘리토	0.149146
689	100일 동안 100가지로 100퍼센트 행복찾기	0.113311
0	그린 북	0.108672
649	스카페이스	0.094057
414	첩혈쌍웅	0.080852
934	본 아이덴티티	0.078587
360	빅 히어로	0.074293
162	스파이 지니어스	0.067581
551	미션 임파서블	0.062059

아이언맨과 Story가 유사한 영화 10개 추천 성공!! 

Review

김성민

데이터에 NULL값이 괜히 있는게 아니구나...

이이삭

데이터 전처리가 절반이다..!
그리고 컨텐츠 기반의 추천이 정확도가 낮지만 결과를 보니까 생각보다 ...쓸 만했다

주하영

전처리를 어떻게 하느냐에 따라 결과물이 다르게 나오는 것을 보고 전처리과정의 중요성을 크게 깨달았다.
자연어처리가 제일 어렵다고 생각했는데 이번 프로젝트를 진행하면서 진짜 어렵구나라고 느꼈고 머릿속에 있던 개념들을 정리하는 계기가 되었다.

THANK YOU!

네이버 영화 추천 시스템

김성민 이이삭 주하영