텍스트데이터분석 프로젝트

- 주제 : 영화가 사람들의 생각에 영향을 미치는가?

- 주제 선정 배경

최근에 사람들이 영화를 보면 인터넷을 활용하여 감상평을 남기는 일이 많아졌다. 또한 댓글 시스템처럼 공감이나 비공감을 받아 그 순으로 정리해둔 플랫폼들도 있다. 내가 봤던 영화 중 상당히 감명받았던 영화인 '내부자들'을 생각해보니 혹시 이러한 정치에 관한 내용이 들어가 있는 영화들이 사람들의 생각에 영향을 주지는 않나? 라는 생각이 들었다.

- 가설설정

본인은 이를 위해 가설을 하나 세웠는데, 영화평 중 공감이 높은 댓글이 사람들의 여론을 반영한다고 보고 만약 긍정적이라면 정치에 관해 관심을 갖게 된다고 가설을 세웠다. 그리고 이 가설을 검증하기 위한 수단을 하나 마련했다. 바로 투표율 데이터를 통해 사람들이 관심이 증가하면 투표율도 자연스럽게 높아지게 될 것이라고 생각하였다.

- 데이터 수집

일단 데이터 수집은 영화평은 다음 영화[1]에서, 투표율 데이터는 중앙선거관리위원회 선거통계시스템[2]에서 가져왔다. 또한 공감이 높은 댓글은 네이버에서 지원하기에 네이버 영화[3]에서 가져왔다.

영화 데이터 수집하는 방법을 보자.

```
In [1]: import requests
        import lxml.html
        import re

In [2]: from selenium.webdriver import Chrome

In [3]: browser = Chrome()
```

셀레늄을 이용해 크롤링 하였는데 셀레늄은 서버와 직접 통신하는 대신 웹 브라우저를 이용하여 간편하게 스크랩을 할 수 있기에 사용하였다.

---

[1] https://movie.daum.net/main/new#slide-1-0

[2] http://info.nec.go.kr/electioninfo/electionInfo_report.xhtml

[3] https://movie.naver.com/

```
In [6]: data = []
for page in range(1,469) :
    res = requests.get(url.format(page))
    root = lxml.html.fromstring(res.text)

    reviews = root.cssselect('p.desc_review')
    scores = root.cssselect('em.emph_grade')

    for review, score in zip(reviews, scores):
        content = review.text_content()
        content = content.strip()
        content = re.sub('₩r',' ', content)
        content = re.sub('r₩s+',' ', content)
        data.append((content, score.text))
```

Lxml.html을 이용해 html처리를 하고 cssselect를 사용해 리뷰와 평점을 긁어왔다. 또한 텍스트 추출 및 불필요한 공백들 삭제까지 진행하였다. 이 과정들을 모든 페이지에 대하여 진행하였다.

```
In [9]: df = pd.DataFrame(data, columns=['review', 'score'])
df
```

```
In [11]: df.to_excel('범죄와의전쟁.xlsx')
```

이것들을 dataframe으로 만들고 감성분석 실시를 위해 sentiment 열을 만들어 댓글 하나하나 분석하여 긍정이면 1, 부정이면 0을 넣어 xlsx문서로 저장하였다. 여기서 어려움을 겪게 되었는데 원래는 긍정/부정을 score, 즉 평점으로 실시하였으나 평점만으로 긍정인지 부정인지는 판단하기 힘들다고 보인다. 평점은 10점을 줬 어도 부정적인 반응이 나올 수 있기 때문에 이 부분들을 체크하기 위하여 직접 모든 댓글에 대하여 반응을 추가하였다. 조금 아쉬운 점은 일부만 직접 체크한 후 머신 러닝으로 학습하여 자동으로 반응을 추가해줄 수 있었다면 좋은 모델이 되었을 것 같다.

이 데이터 수집을 범죄와의 전쟁(2012 개봉), 광해(2013 개봉), 내부자들(2015 개봉), 더 킹(2017 개봉)에 대해 진행하였다. 4개의 영화에 대해 진행한 것은 대선, 총선, 지선이 각각 5년, 4년, 4년마다 열리기 때문에 그 사이의 간격을 두기 위해 해당 년도 중 유명세를 탄 정치 영화를 선정한 것이다.

- **주제분석**

먼저 이 영화 감상평에 영향을 미치는 잠재변수, 즉 '주제'를 파악하기 위해 주제분석을 진행하였다. LSA를 이용해 주제분석을 실시하였는데 문서들을 의미상으로 재배치시키기 위하여 차원축소를 진행하였고 다른 해석을 보기 위해 회전까지 진행하였다. 일단 범죄와의 전쟁 데이터로 설명하겠다.

```
In [2]: df1 = pd.read_excel('범죄와의전쟁.xlsx', index_col=0)
df2 = pd.read_excel('광해.xlsx', index_col=0)
df3 = pd.read_excel('내부자들.xlsx', index_col=0)
df4 = pd.read_excel('더킹.xlsx', index_col=0)
df1
```

```
In [3]: df1 = df1.dropna(axis=0)
        df2 = df2.dropna(axis=0)
        df3 = df3.dropna(axis=0)
        df4 = df4.dropna(axis=0)
        df1
```

먼저 데이터를 불러와 review에 아무것도 없는 행들을 제거하였는데, 이는 분석에 도움되지 않기에 제거하였다. 그 결과 4680개의 데이터에서 1829개가 되었다.

	review	score	sentiment
0	영화 어제 봤습니다. 80~90년대 부산에 산 사람만이 알 수 있는 표현들이 너무 ...	10	1
1	ㅂㅈㅇㅇㅈㅈ 아~ 부산냄새~ 나는 서울 출신 이라서 무슨말하는지 이해안감	0	0
2	조폭영화중 최고 최민식이 다 살렸다	6	1
4	프리비에르 돌았네~~	9	1
5	살아있네~~	9	1
...	...	...	...
4675	미치겠다.. 도대체 언제 개봉이냐?? ㅎ	10	1
4676	캐스팅 쩐다 진짜....ㄷㄷㄷㄷ	10	1
4677	이영화 대박날듯!!!!!!!!!!!!!!!!!!!!!!100	10	1
4678	부산 촬영현장을 봤는데....두배우의 카리스마..대단합니다.	10	1
4679	진짜 재밌을 것 같아요!	10	1

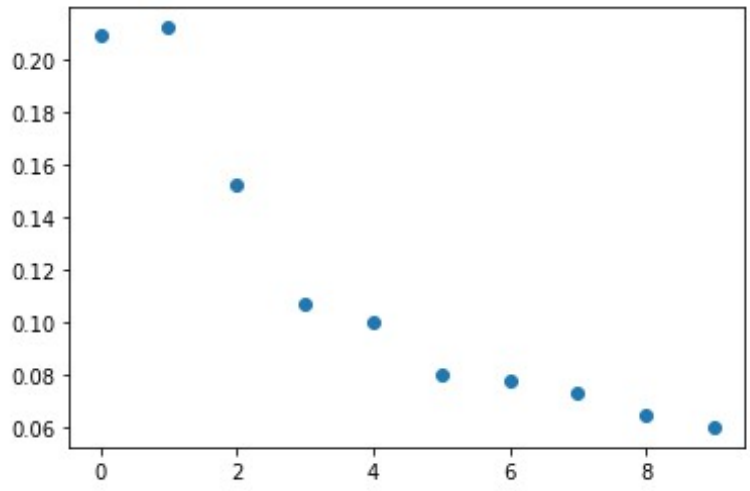
1829 rows × 3 columns

단어 문서행렬을 만들고 데이터를 저장한다.

```
In [7]: from sklearn.feature_extraction.text import CountVectorizer

In [8]: cv1 = CountVectorizer(stop_words='english', max_features=2000)

In [9]: x1_train = cv1.fit_transform(review1_train)

In [10]: x1_test = cv1.fit_transform(review1_test)

In [12]: import joblib

         data1 = {
             'x1_train': x1_train,
             'x1_test': x1_test,
             'cv1': cv1
         }
         joblib.dump(data1, 'review1.pkl')

Out[12]: ['review1.pkl']
```

일단 이 데이터를 10차원으로 줄이고 적절한 주제를 찾기 위해 scree plot을 그려 보았다. 이는 차원의 크기에 따라 설명하는 분산의 양이 달라지는 것이 보이고, 이 그래프가 꺾이는 점에서 차원을 결정하므로 8차원에서 결정하였다.

```
[<matplotlib.lines.Line2D at 0x262a620f048>]
```



더 진행시켜 0번 주제를 보았더니

	0	word
1136	0.584772	은근
1556	0.363587	중금
1876	0.356936	하지만
1724	0.333759	칭할
1087	0.256570	요소도

다른 해석을 보기 위하여 회전시켜 보았다. 그 결과,

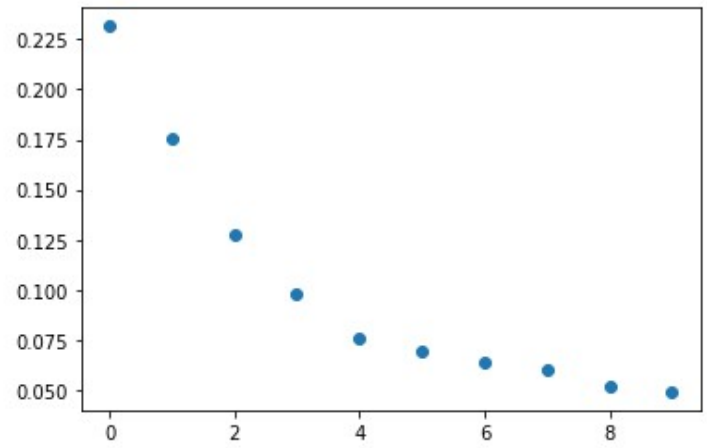
	0	word
1876	0.699419	하지만
1724	0.624395	칭할
1092	0.107425	욕심이
1081	0.099306	왜이러지
1093	0.097674	욕이

가 나왔다. 이러한 단어들을 보고는 무언가를 판단하기가 힘들다고 생각하였다.

나머지 영화들에 대해서도 scree plot을 그려 차원을 판단하고 주제 분석, 회전까지 실시하였다.

광해 :

```
[<matplotlib.lines.Line2D at 0x262a6bd8f08>]
```



6차원으로 결정.

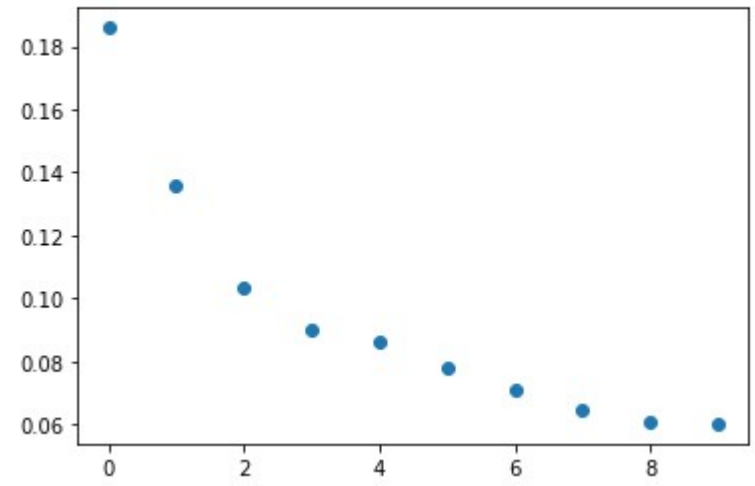
	0	word
1220	0.824525	이병헌
1677	0.292502	코믹하면서
1400	0.204753	재미도
1183	0.134213	이래야한다는
327	0.111095	무엇보다

	0	word
1220	0.956345	이병헌
756	0.059654	연기자네요
1736	0.059519	틀에서는
1491	0.057741	지루하고
615	0.032118	언제개봉하냐규ㅠㅠㅠㅠㅠ

(좌) 6차원 (우) 회전

내부자들 :

```
[<matplotlib.lines.Line2D at 0x262a6d3edc8>]
```



8차원으로 결정.

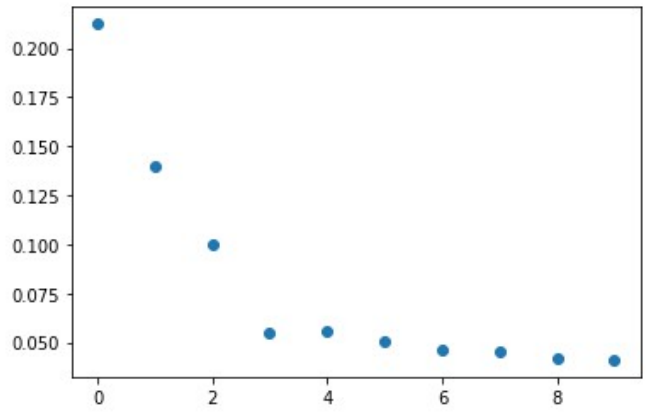
	0	word
867	0.807323	없네
1289	0.275999	짖어대다가
1005	0.158278	웃고
194	0.133932	그런시대에
785	0.129271	아닐까

	0	word
867	0.953013	없네
403	0.082683	돼지말자
1482	0.051581	확실
32	0.051020	3시간이란
1386	0.044396	포기한

(좌) 8차원 (우) 회전

더 킹 :

```
[<matplotlib.lines.Line2D at 0x262a69d9448>]
```



6차원으로 결정.

	0	word
1253	0.878654	재미있다
342	0.227309	남자의
1625	0.183281	초딩나레이션과
1674	0.138466	친절해서
1219	0.114310	잡탕을

	0	word
1253	0.959842	재미있다
588	0.040358	봤지만
443	0.037591	매력을
783	0.032657	영구히
823	0.032433	영화속

(좌) 6차원 (우) 회전

이렇게 4가지 영화 모두 보았는데 무언가를 판단하기에는 부족한 정보인 것 같다. 그래서 주로 나오는 단어들을 보아 영화의 전반적 내용을 알기 위하여 단어구름을 그려보았다.

- **단어문서행렬 + 단어구름**

```
# nan값 있는 항 모두제거
df1 = df1.dropna(axis=0)
df2 = df2.dropna(axis=0)
df3 = df3.dropna(axis=0)
df4 = df4.dropna(axis=0)
```

일단 데이터들에 review가 없는 항들부터 제거하고 시작하였다.

```
nlp = stanza.Pipeline('ko')
```

```
def extract_nouns(text):
    doc = nlp(text)
    for sentence in doc.sentences:
        for word in sentence.words:
            lemma = word.lemma.split('+')
            xpos = word.xpos.split('+')
            for w,p in zip(lemma,xpos):
                if p.startswith('n'):
                    yield w
```

Stanza를 사용하여 명사만 추출하는 함수를 만들어 주고,

```
cv1 = CountVectorizer(max_features=2000, tokenizer=extract_nouns)
```

```
tdm1 = cv1.fit_transform(df1['review'])
```

단어 문서행렬을 만들어 주었다.

```
word_count1 = pd.DataFrame({
    '단어': cv1.get_feature_names(),
    '빈도': tdm1.sum(axis=0).flat
})
```

단어 별 빈도를 표로 정리하였고

```
word_count1.to_excel('범죄와의전쟁-count.xlsx')
```

xlsx파일로 저장해 주었다. 나머지 3개의 영화에 대해서도 똑같이 적용하였다.

그리고 하나씩 단어구름을 그려보았는데, 그 결과로

범죄와의 전쟁 :

대충 보았을 때 영화배우와 연기에 관한 내용뿐만 아니라 조폭, 나라, 시대, 검사 등 정치와 얽힌 얘기가 나온다.

광해 :

대충 보았을 때 영화배우와 연기에 관한 내용뿐만 아니라 정치, 나라, 시대, 대통령 등 정치와 얽힌 얘기가 나온다.

내부자들 :

대충 보았을 때 영화배우와 연기에 관한 내용뿐만 아니라 검사, 권력, 내부자, 정치 등 정치와 얽힌 얘기가 나온다.

더 킹 :

대충 보았을 때 영화배우와 연기에 관한 내용뿐만 아니라 권력, 정치, 국민, 검찰 등 정치와 얽힌 얘기가 나온다.

아까 주제 분석과 다르게 4가지 영화 모두 정치에 관련된 영화라는 결론이 나왔다.

- **감성분석**

다음으로 영화평 중 공감이 높은 댓글이 사람들의 여론을 반영한다고 보고 만약 긍정적이라면 정치에 관해 관심을 갖게 된다는 가설을 검정하기 위해 감성분석을 실시하였다.

```
# nan값 있는 항 모두제거
df1 = df1.dropna(axis=0)
df2 = df2.dropna(axis=0)
df3 = df3.dropna(axis=0)
df4 = df4.dropna(axis=0)
```

일단 데이터들에 review가 없는 항들을 제거하고,

```
with open('nsmc1.txt', 'w', encoding='utf8') as f:
    f.write('₩n'.join(df1['review']))
```

```
from sentencepiece import SentencePieceTrainer
SentencePieceTrainer.Train('--input=nsmc1.txt --model_prefix=nsmc1 --vocab_size=3000')
```

```
from sentencepiece import SentencePieceProcessor
sp1 = SentencePieceProcessor()
sp1.Load("nsmc1.model")
```

인터넷 글들은 맞춤법이나 띄어쓰기를 잘 지키지 않아 형태소 분석 대신에 sentencepiece를 이용해 준단어 토큰화를 진행하였다. 토큰의 종류는 3000개로 하였다.

```
cv1 = CountVectorizer(lowercase=False, tokenizer=sp1.encode_as_pieces)
```

```
tdm1 = cv1.fit_transform(df1['review'])
```

단어문서행렬을 만들어 주는데 영어가 섞여 있으면, 영어 대문자 그대로 사용하기 위해 지정해주었다.

```
x1 = tdm1
y1 = df1['sentiment']
```

```
x1_train, x1_test, y1_train, y1_test = train_test_split(x1, y1, test_size=0.2, random_
```

전체데이터의 20%는 테스트용 데이터로 분할하였다.

```
cv1.tokenizer = None
```

저장을 하게 되면 sp.encode_as_pieces 메소드가 c++언어로 작성된 함수를 python으로 변환시켜 놓은 것이기 때문에 joblib저장이 안되므로 지정해준다.

```
model1 = tf.keras.models.Sequential()
```

```
model1.add(
    tf.keras.layers.Dense(
        1,
        input_shape=(3038,),
        activation='sigmoid',
        kernel_regularizer=tf.keras.regularizers.l2(0.001)
    ))
```

출력을 긍정(1)/부정(0)으로 설정하였고 sigmoid를 사용하였으며, 가중치를 전반적으로 작게 만드는 L2정규화를 실시하였다.

```
model1.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

학습알고리즘을 경사하강법인 adam을 사용, 손실함수는 교차 엔트로피, 보조 지표로 정확도를 사용하였다.

```
model1.fit(x1_train.toarray(), y1_train, epochs=100, validation_split=0.1,
          callbacks=[tf.keras.callbacks.EarlyStopping()])
```

모든 데이터를 입력하는 횟수인 Epochs를 100으로 충분히 지정해주었고, 일부 데이터를

validation데이터로 사용하기 위해 0.1로 지정해주었다. Epochs가 끝날 때 마다 earlystopping 함수 실행으로 validation을 실시하여 중단할지 말지 결정하게 해주었다.

범죄와의 전쟁 모델은

```
Epoch 23/100
1316/1316 [==============================] - 0s 33us/sample - loss: 0.2620 - accurac
y: 0.9309 - val_loss: 0.2915 - val_accuracy: 0.9456
```

Epochs가 23일 때 중단되었다.

```
model1.evaluate(x1_test.toarray(), y1_test)
```

```
366/366 [==============================] - 0s 44us/sample - loss: 0.3559 - accuracy:
0.9071
```

최종성능을 test데이터로 평가해보면 accuracy가 0.9071만큼의 성능을 보여준다.

```
weights2,_ = model2.trainable_weights

token_weight2 = pd.DataFrame({'토큰': cv2.get_feature_names(), '가중치': weights2.nump

token_weight2.sort_values('가중치').head()
```

계수 확인을 위해 trainable_weights 함수를 사용하여 dataframe을 만들어 주었다.

	토큰	가중치
73	T	-0.318324
2609	조선	-0.303892
634	_별로	-0.268537
74	V	-0.267715
1139	_재미없	-0.255375

	토큰	가중치
1294	_최고	0.375433
950	_연기	0.379698
1151	_재밌게	0.381203
1	!!	0.392650
738	_살아있네	0.440150

(좌) 부정단어 (우) 긍정단어

나머지 데이터들도 살펴보자.

광해 :

```
Epoch 30/100
2466/2466 [==============================] - 0s 33us/sample - loss: 0.1986 - accurac
y: 0.9461 - val_loss: 0.2694 - val_accuracy: 0.9200
```

Epochs가 30일 때 중단되었다.

```
model2.evaluate(x2_test.toarray(), y2_test)
```

```
686/686 [==============================] - 0s 25us/sample - loss: 0.2832 - accuracy:
0.9155
```

최종성능을 test데이터로 평가해보면 accuracy가 0.9155만큼의 성능을 보여준다.

	토큰	가중치
1694	냐	-0.508830
443	_데이브	-0.464805
1049	_이게	-0.447141
1377	_표절	-0.355221
854	_알바들	-0.348203

	토큰	가중치
1066	_이병헌의	0.531894
1329	_최고	0.531994
0	!	0.567077
1129	_잘	0.578303
915	_연기	0.591277

(좌) 부정단어 (우) 긍정단어

내부자들 :

```
Epoch 100/100
709/709 [==============================] - 0s 38us/sample - loss: 0.1293 - accuracy:
0.9803 - val_loss: 0.2448 - val_accuracy: 0.9494
```

Epochs가 100까지 다 돌았다.

```
model3.evaluate(x3_test.toarray(), y3_test)
```

```
197/197 [==============================] - 0s 36us/sample - loss: 0.1884 - accuracy:
0.9695
```

최종성능을 test데이터로 평가해보면 accuracy가 0.9695만큼의 성능을 보여준다.

	토큰	가중치
577	_무슨	-0.490435
1783	내용인지	-0.473362
1210	_재미없	-0.427505
2900	해서	-0.406046
398	_대	-0.381082

	토큰	가중치
90	~~	0.543954
134	_3	0.551459
0	!	0.576938
1392	_최고	0.729540
248	_굿	0.749861

(좌) 부정단어 (우) 긍정단어

더 킹 :

```
Epoch 26/100
1833/1833 [==============================] - 0s 32us/sample - loss: 0.2843 - accurac
y: 0.9242 - val_loss: 0.4636 - val_accuracy: 0.8333
```

Epochs가 26일 때 중단되었다.

```
model4.evaluate(x4_test.toarray(), y4_test)
```

```
510/510 [==============================] - 0s 29us/sample - loss: 0.4031 - accuracy: 0.8490
```

최종성능을 test데이터로 평가해보면 accuracy가 0.8490만큼의 성능을 보여준다.

	토큰	가중치
1365	_지루하고	-0.438611
606	_못	-0.427968
1404	_최악의	-0.391649
870	_실망	-0.362776
367	_나레이션	-0.347984

	토큰	가중치
1202	_잘	0.505837
1507	_현실	0.507013
69	^^	0.543683
1243	_재밌게	0.562631
355	_꼭	0.597373

(좌) 부정단어 (우) 긍정단어

이들 각각을 네이버 영화에서 best 댓글 상위 10개를 추출하여 그것들에 대한 감성분석을 실시하였다. 똑 같은 영화의 best 댓글을 적용시켰을 때,

```
model1.predict(x1_new.toarray())
```

```
array([[0.95040005],
       [0.70022285],
       [0.7016349 ],
       [0.67459166],
       [0.9607522 ],
       [0.84816766],
       [0.76829845],
       [0.74512774],
       [0.8576355 ],
       [0.73191404]], dtype=float32)
```

거의 모두가 긍정적 반응이었다.

```
array([[0.8044546 ],
       [0.96871567],
       [0.92740744],
       [0.9713144 ],
       [0.9750368 ],
       [0.9583796 ],
       [0.8792013 ],
       [0.79396695],
       [0.9498907 ],
       [0.8677815 ]], dtype=float32)
```

```
array([[0.86940473],
       [0.9292755 ],
       [0.9042327 ],
       [0.993309  ],
       [0.8398019 ],
       [0.8648984 ],
       [0.9436459 ],
       [0.9948738 ],
       [0.9672263 ],
       [0.9522758 ]], dtype=float32)
```

(좌) 광해 (우) 내부자들

```
array([[0.6073537],
       [0.8728601],
       [0.8398966],
       [0.9098527],
       [0.878911 ],
       [0.9556842],
       [0.8199466],
       [0.9635894],
       [0.9390126],
       [0.9537958]], dtype=float32)
```

(좌) 더 킹

나머지 3개의 영화들에 대해서도 모두가 긍정적 반응을 보였다. 영화평 중 공감이 높은 댓글이 사람들의 여론을 반영한다고 보고 만약 긍정적이라면 정치에 관해 관심을 갖게 된다는 가설이 해당된다고 보고 영화를 보고 정치에 대해 관심을 갖게 된다고 결론을 내렸다.

정치에 대해 관심을 갖게 되는 것을 검증하기 위해 투표율 데이터로 확인해 보았다. 사람들의 관심이 높아진다면 투표율이 높아진다고 생각하기 때문에 투표율이 실제로 증가한다면 검증된다고 보았다.

- **투표율과 검증**

투표율 데이터를 전처리 하여,

	시도명	5회 동시지방선거 \n(2010.06.02)	19대 국회의원선거 \n(2012.04.11)	18대 대통령선거 \n(2012.12.19)	6회 동시지방선거 \n(2014.06.04)	20대 국회의원선거 \n(2016.04.13)	19대 대통령선거 \n(2017.05.09)	7회 동시지방선거 \n(2018.06.13)
3	합계	54.5	54.2	75.8	56.8	58.0	77.2	60.2

이러한 결과물을 얻었다. 그리고 지선, 총선, 대선마다 투표율 기댓값이 다르기 때문에 각각의 선거마다 비교하기로 하였다.

대선부터 보자면,

	18대 대통령선거\n(2012.12.19)	19대 대통령선거\n(2017.05.09)
3	75.8	77.2

18대와 19대 사이의 영화가 광해, 내부자들, 더 킹이 있다. 투표율이 실제로 올라간 모습이다.

그 다음 총선,

	19대 국회의원선거\n(2012.04.11)	20대 국회의원선거\n(2016.04.13)
3	54.2	58.0

19대와 20대 사이의 영화가 광해, 내부자들이 있다. 투표율이 실제로 올라간 모습이다.

마지막으로 지선,

	5회 동시지방선거\n(2010.06.02)	6회 동시지방선거\n(2014.06.04)	7회 동시지방선거\n(2018.06.13)
3	54.5	56.8	60.2

5회와 6회 사이에는 범죄와의 전쟁, 광해가 있고, 6대와 7대 사이에는 내부자들, 더 킹이 있다. 투표율이 두 구간 모두 실제로 올라간 모습이다.

내가 세운 가설(영화평 중 공감이 높은 댓글이 사람들의 여론을 반영한다고 보고 만약 긍정적이라면 정치에 관해 관심을 갖게 된다)이 검증되었다.

- **프로젝트 진행 후 느낀 점**

이 프로젝트를 진행하면서 투표율로 정치에 대한 관심을 파악하는 것, 댓글로 여론을 파악하는 것 두개 모두가 실제로 투표율이 오르고 정치에 대한 관심이 있으며, 여론을 반영하는 것에 엄청난 영향을 끼치는 것이라고 생각이 되진 않는다. 오히려 뉴스, 사건 등이 투표율과 여론에 더 영향을 미친다고 생각한다. 그러나 내가 분석한 이 결과가 아예 의미가 없다고는 할 수 없다. 결과가 나름 정확하게 나왔고 단지 가설을 검정할 수단이 부족했을 뿐이라고 생각한다.

- **본인평가**

[수업반영] 4점. 전개상 불필요한 7주차의 과정인 NMF, LDA를 이용한 주제분석 및 평가를 제외하고는 수업내용을 모두 반영하였음.

[문제해결] 4점. 감성분석을 위해 데이터에 sentiment 행의 긍정 부정 반응을 넣는 것에 대해 본인은 비록 수작업을 하였으나 해결방안인 머신 러닝으로 학습하면 될 것이라고 제시함.

[내용구성] 4점. 주제와 관련하여 구성은 제대로 하였으나 약간의 의문이 드는 구성이 있음.

[결과해석] 3점. 표면적인 결과로만 보고 실제 이것이 영향을 미치는가? 에 대한 물음이 남아있음.