유튜브 조회수 예측을 위한 데이터 분석

이경민(930408)
kmlee0408@naver.com
이경민(930408)/정진형(931014)

"초등학생 장래희망에 유튜버가 공무원과 함께 1~2위를 다툰다. 진입 장벽도 없다.
취미를 직업으로 만들 수 있다. 인기와 돈을 거머쥘 수 있는 새로운 생태계로 돈과 사람이 몰리고 있다."
-조선일보 기사 中- (https://goo.gl/N2q4rR)

전 세계 최대 무료 동영상 공유 사이트 Youtube에 대한 관심은 초등학생에 까지 영향을 미치고 있다. 유튜브에 영상을 업로드하여 수익을 창출하는 직업을 '유튜버'라고 한다. 유튜버가 돈을 버는 원리는 조회수에 비례하여 제공되는 광고 수익과 인기로 인해 얻는 부수적인 수입이다.

- JTBC '랜선 라이프 1화 中- '

유튜버 '대도서관'의 경우 연간 수입이 17억원이다. 왜 초등학생들이 유튜버를 장래희망 1순위로 뽑는지 이해가 된다. 이러한 수입은 위에 언급한대로 결국 영상의 조회수와 밀접한 관련이 있다.

그렇다면 대중들이 좋아하는 영상은 무엇일까?

영상의 어떤 특징으로 조회수가 많은 지, 영상을 올리기 전 영상의 조회수를 예측 하는 것이 가능한지 궁금하여 분석을 시작하게 되었다.
조회수에 영향을 주는 특징을 발견한다면, 조회수가 높은 영상을 만들어 내는데 유용할 것이다. 또한 영상을 올리기 전에 영상의 조회수가 얼마인지 알 수 있다면, 유튜버는 영상을 더 수정해서 올릴지 말지 결정 할 수 있을 것이다. 물론 팀원 모두 유튜버가 될 생각은 없지만, 흥미로운 주제이므로 분석을 시작해보겠다.

분석에는 'Kaggle'에서 제공하는 'Trending YouTube Video Statistics ' 과 구글 API 통해 추가적으로 유튜브 영상 길이 데이터를 확보했다.

데이터 출처	Trending YouTube Video Statistics (https://www.kaggle.com/datasnaek/youtube-new)
데이터 설명	• 데이터 세트에는 일일 인기 급상승 YouTube 동영상에 대한 몇 개월 (및 계산) 데이터가 포함된다. • 데이터는 미국, GB, CA 지역 (각각 미국, 영국, 캐나다)에 포함되며 최대 200 개의 트렌드 동영상이 하루에 나열된다. • 확보한 영상 데이터는 총 118266개 다.

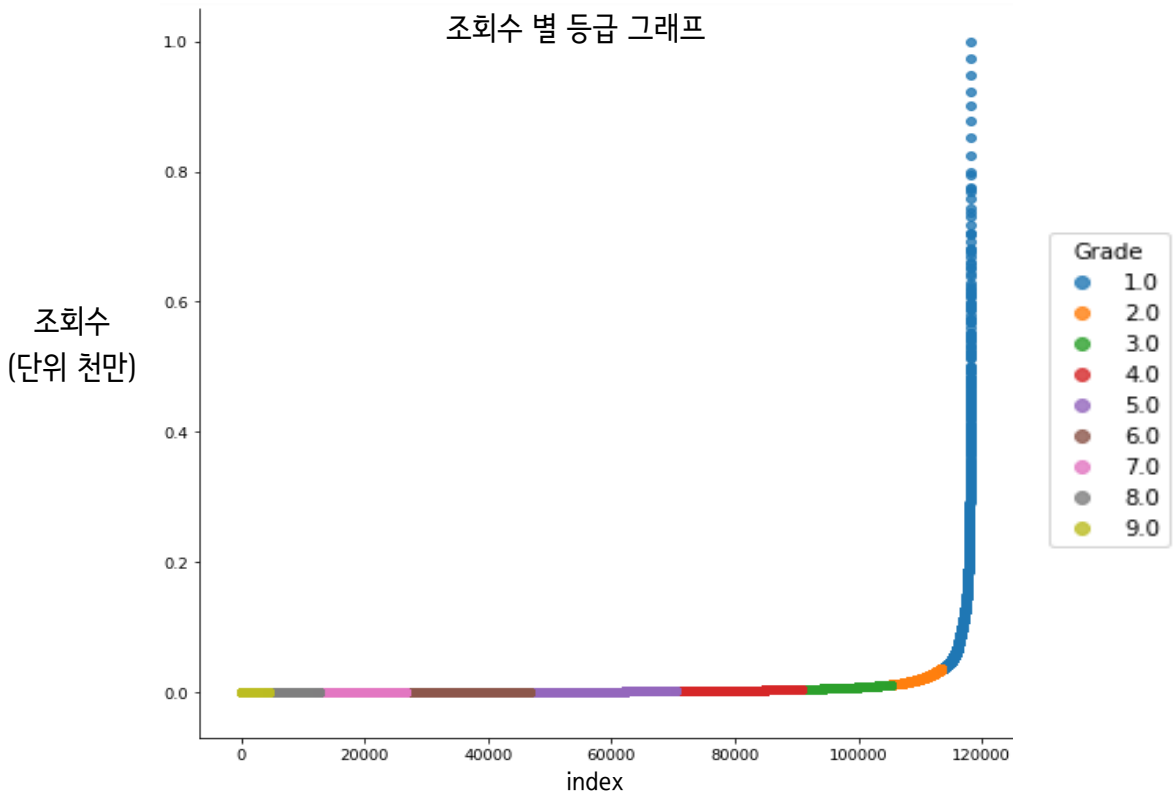
데이터의 자세한 변수 설명과 변수로 분석할 수 있는 방향은 다음과 같다.

<table>
<tr><td colspan="2">유튜브에 영상 업로드 전
알 수 있는 데이터</td><td colspan="2">유튜브에 영상 업로드 후
생성되는 데이터</td></tr>
<tr><td>변수 이름</td><td>설명</td><td>변수 이름</td><td>설명</td></tr>
<tr><td>title</td><td>동영상의 제목</td><td>video_id</td><td>유튜브 영상을 구분 짓는 고유한 id</td></tr>
<tr><td>channel_title</td><td>해당 동영상의 유튜브 채널 제목</td><td>Trending_date</td><td>인기 급상승 동영상으로 선정된 날짜</td></tr>
<tr><td>publish_time</td><td>동영상을 업로드 한 날짜</td><td>views</td><td>동영상의 조회수</td></tr>
<tr><td>tags</td><td>동영상을 설명하는 짧은 단어, 게시자가 작성한다.</td><td>likes</td><td>동영상의 좋아요 수</td></tr>
<tr><td>description</td><td>영상을 설명하는 글</td><td>dislikes</td><td>동영상의 싫어요 수</td></tr>
<tr><td>duration
(구글 API)</td><td>영상의 길이(시간)</td><td>comment_count</td><td>영상의 댓글 수</td></tr>
<tr><td>category_id</td><td>동영상의 장르 카테고리를 코드로 표현한 변수
예) 10: 음악 / 17: 스포츠</td><td colspan="2">+ 업로드 전에
알 수 있는 모든 변수</td></tr>
<tr><td colspan="2">영상을 올리기 전,
영상의 조회수를 예측 할 수 있을까?</td><td colspan="2">조회수에 영향을 주는 변수는 뭘 까?</td></tr>
</table>

전체 얻을 수 있는 변수를 크게 영상 업로드 전과 후로 나누어 각각 다른 방향의 분석을 하고자 한다. 그 이유는 영상 조회수를 예측하기 위해, 예측 후 발생하는 변수를 사용할 수 없기 때문이다. 예를 들어 '동영상의 좋아요 수' 변수는 조회수가 발생해야 부수적으로 생기는 변수이므로 예측에 사용할 수 없다.

그렇다면 대중들이 좋아한다는 것의 기준은 무엇 일까?

대중들이 좋아하는 영상의 기준을 잡기 위해 영상 별 조회수 데이터에 '수능 등급제'를 도입하였다. 아래와 같이 'Python'을 이용하여 조회수('views')별 9등급부터 1등급을 부여하였다.



- 조회수 별 등급 그래프

그래프에서 갑자기 조회수가 급등하는 지점을 발견 하였다. 그리고 그 지점의 시작(15,399,127회)이 1등급과 2등급 사이 인 것으로 보아, 대다수의 영상은 조회수가 유사하고, 약 4%의 영상만이 특출 난 조회수를 가진 것을 알 수 있었다. 쉽게 말해, 조회수의 빈부격차가 심한 것을 확인 했다.

등급을 나누는 특징에는 어떤 것이 있을까?

특징을 분석하기 위해 먼저 데이터에서 제공하는 변수들의 속성과 등급별 어떻게 분포되어 있는지 확인해보자. 요리를 하기 전 재료의 맛을 먼저 보는 것처럼.

변수 명	속성	변수 명	속성
video_id	object	tags	object
trending_date	object	views	int64
title	object	likes	int64
channel_title	object	dislikes	int64
category_id	int64	comment_count	int64
publish_time	object	description	object
duration	int64		

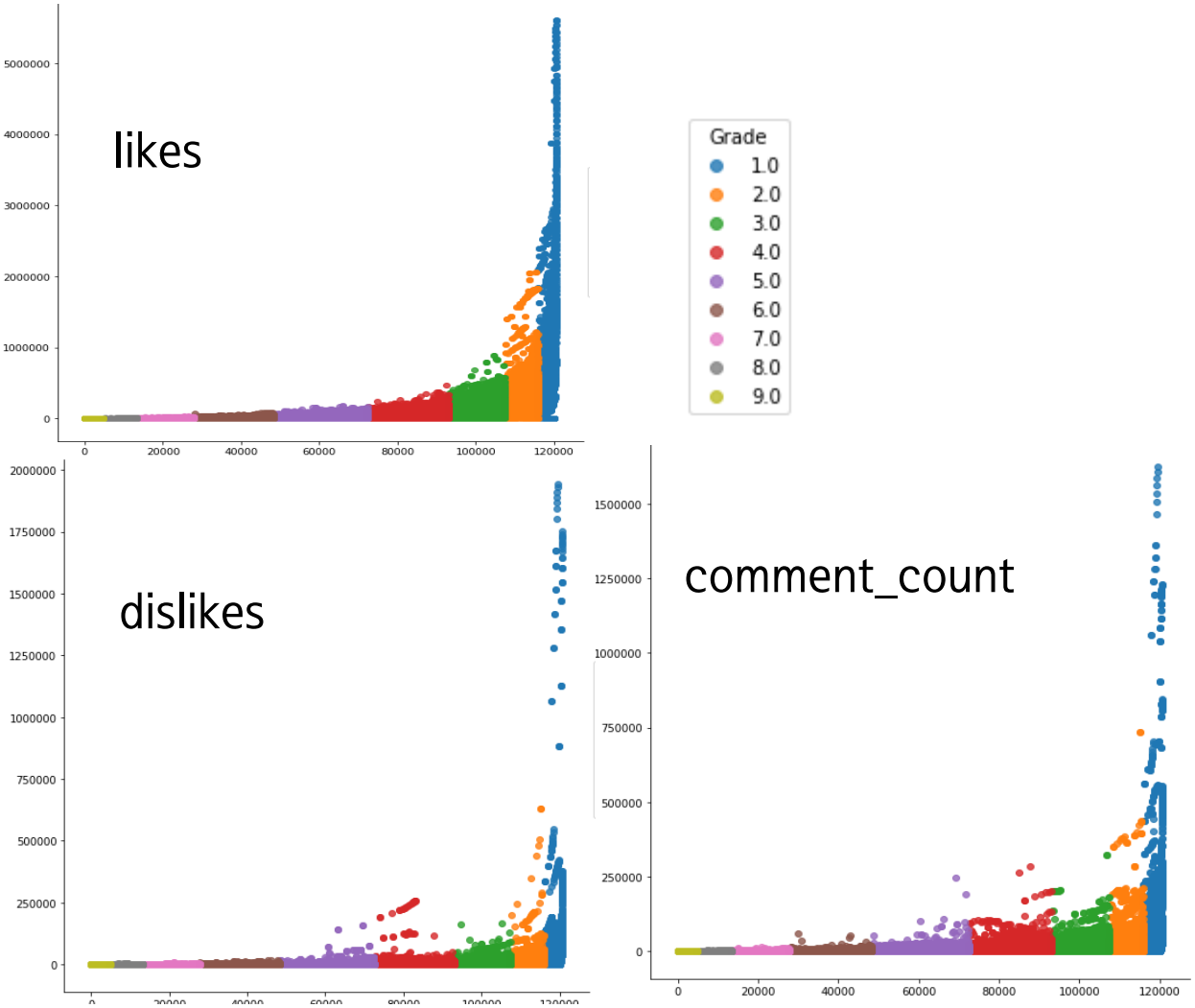
- 변수 속성 표

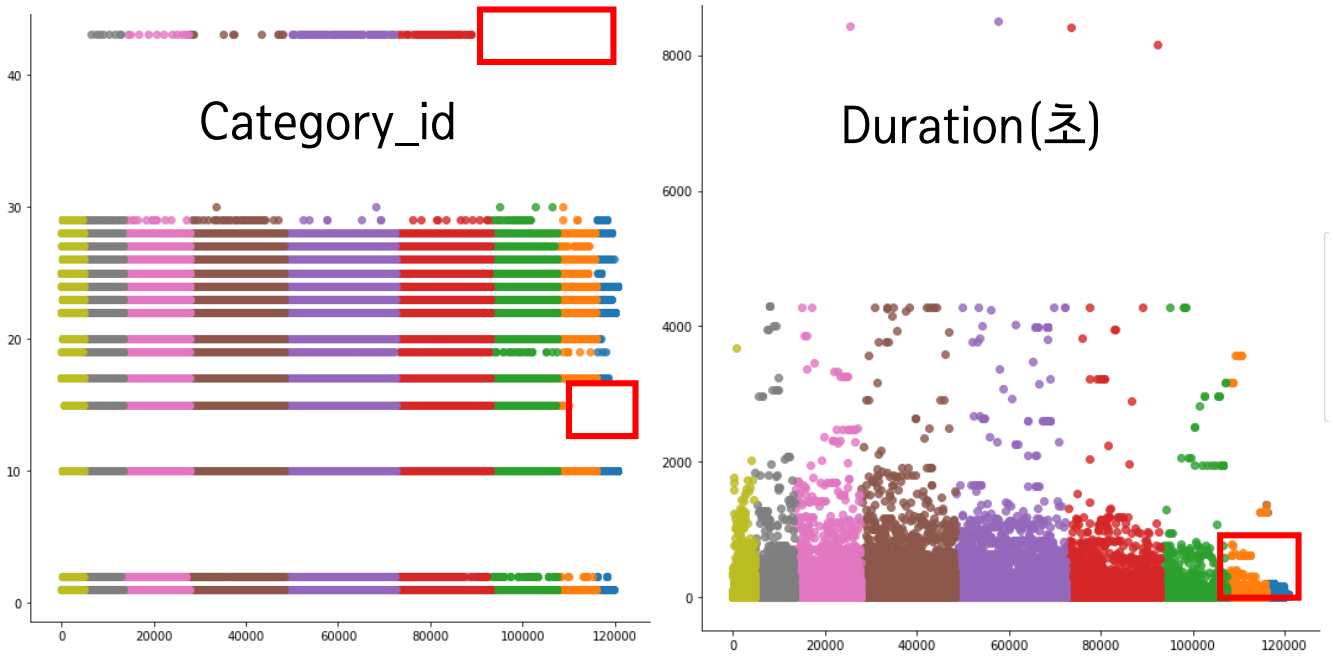
Object로 표기된 값은 문자로 된 데이터이고, int64로 표기된 값은 숫자로 된 데이터이다. 수치형 데이터(int64)의 경우 데이터를 분석하는데 무리가 없지만, 문자로 된 데이터는 모델링에 넣기가 어렵다.

그렇다고 해서 텍스트 데이터를 버리기엔 너무 아깝다는 생각이 든다. 수치형 데이터로 활용할 방안을 생각해봐야 겠다.

또한 'category_id'처럼 숫자로 되어있지만, 범주형 데이터인 경우 숫자의 의미가 연속형이 아닌 이산형이다. 'Music'이라는 카테고리와 'Food'라는 카테고리 사이에 어떤 연관도 없지만, 그것이 연관이 있는 것처럼 표기가 되어 있다. 따라서 각 카테고리를 따로 분류하여 발현되면 1, 발현이 되지 않으면 0으로 바꿔주는 'One-hot-encoding'이 필요해 보인다.

Numeric 변수의 등급별 분포를 확인해 보자.





- 변수 별 등급 그래프

Numeric 변수들의 전체 분포를 보면 'likes', 'dislikes', 'comment_count'의 그래프가 'views'의 그래프 모양과 유사하므로 조회수와 좋아요, 싫어요, 댓글 수는 선형관계가 있을 것으로 보인다.
'category_id' 그래프에선 신기하게도 상위 조회수 영상에 나오지 않는 카테고리가 발견되었다.(빨간 네모) 바로 'Pet & Animals'와 'Shows' 였다. 또한 'duration' 그래프를 보아 상위 조회수 영상의 길이가 약 13분(1000초) 이하 인 것을 알 수 있다.

1500만 이상의 조회수를 올리고 싶다면, 동물, 쇼 주제를 피한 13분 내외의 영상을 올리는 것이 좋겠다.

변수의 특징을 확인해 보았다. 그 중 텍스트 변수는 새롭게 가공하여 사용하여야 할 것을 알 수 있었다.

텍스트 변수로 새로운 수치형 변수를 만들어 낼 수 있지 않을까?

다시 목표를 생각해보자. 우리의 목표는 **동영상의 조회수 예측과, 조회수에 영향을 주는 변수**를 찾는 것이다. 목표에 맞춰 관련이 될 만한 변수를 생각해보았다.

기존 변수	새로운 변수	내용/근거
title,tags	word_count	제목과 태그에 자주 포함된 단어를 추출한다/ 조회수가 높은 영상 제목에 자주 등장하는 단어 패턴이 있을 것이다.
descripton	senti_mark	영상을 설명하는 문장의 긍정,부정을 파악/ 영상을 설명 글엔, 영상자체의 내용이 담겨있기 때문에 내용을 긍정(1)과 부정(-1)로 구분한다면 특징이 있을 것이다.

기존 변수	새로운 변수	내용/근거
publish_time	publish_divide	영상 게시 시간을 6개로 구분/ 24시간을 6개로 나눈 이유는 아침,점심,저녁이 다른 특징을 갖는 것 처럼 시간대별로 특징이 있기 때문에 연속형에서 이산형으로 바꾸어 주었다.
trending_time publish_time	gap_time	게시일로부터 인기동영상으로 선정된 기간을 추출/ 얼마 만에 인기동영상으로 됐는지가 조회수에 영향을 줄 것이다. 아마 단기간에 선정되면 더 높지 않을까

- 새로운 변수 설명

publish_divide나 gap_time은 단순한 계산이므로 자세한 설명은 하지 않겠다. 그 외의 새로운 변수를 어떻게 생성했는지는 간략하게 설명하겠다.

- word_count

유튜브 제목에 사용된 최빈도 단어

```
Out[15]: [(31880, '|'),
          (22314, 'the'),
          (10767, 'official'),
          (10472, 'to'),
          (10048, 'a'),
          (9133, 'in'),
          (8530, 'of'),
          (8050, 'video'),
```

title 선정 단어: |,official, video,2018

유튜브 태그에 사용된 최빈도 단어

```
Out[95]: [(24687, 'the'),
          (17043, 'to'),
          (16680, 'of'),
          (15393, 'and'),
          (10458, 'a'),
          (10220, 'in'),
          (8622, 'vs'),
          (7034, 'new'),
```

tags 선정 단어: new, vs, music, you

'the'나 'a'같은 의미 없는 단어를 제외한 주요 명사 4개를 각각 선정하여, 영상 제목 혹은 태그에 포함이 되는지 1,0으로 표기 한 변수를 생성하였다. (one hot encoding)

- senti_mark

```
for i in range(len(df_)):

    text = df_.loc[i,col]
    try:
        wiki = TextBlob(text)
    except:
        continue

    mark = wiki.sentiment.polarity
    df_.loc[i,'senti_mark'] = mark
```

TextBlob library를 이용하여 해당 문장을 1(긍정) ~ -1(부정)사이 값으로 표현해준다.
TextBlob.sentiment.polarity

새롭게 만들어 낸 변수들까지 합쳐서, 카테고리별 어떤 특징이 있는지 확인해보았다.

	Most_Frequent_grade	Most_Frequent_publish_time_devide	Mean_duration	Mean_sentiment
Movies	3	4.0	7726.000000	0.200000
Music	3	4.0	311.809242	0.159727
Film & Animation	4	5.0	483.836437	0.173816
Comedy	4	5.0	440.159086	0.162853
Sports	5	1.0	549.795137	0.214433
Autos & Vehicles	5	5.0	958.250851	0.196376
Entertainment	5	5.0	828.242639	0.174249
Howto & Style	5	5.0	762.677639	0.202971
Science & Technology	5	5.0	704.897644	0.164781
Pets & Animals	5	4.0	438.544158	0.219256
Shows	5	6.0	879.218905	0.430448
Travel & Events	6	4.0	834.742697	0.325585
Gaming	6	5.0	1843.603697	0.176979
People & Blogs	6	5.0	817.403027	0.170613
Education	6	4.0	664.970039	0.161116
News & Politics	7	4.0	1196.937492	0.137883
Nonprofits & Activism	9	5.0	3132.149321	0.084688

- 카테고리별 동영상의 '최빈 등급 수', 최빈 게시 시간대', '평균 동영상 길이', '평균 감성 수치'

추출한 특성(동영상 카테고리, 동영상 길이, 감성 수치, 동영상 게시 시간대)와 등급 간의 상관관계를 파악하기 전에, 기술적 통계분석을 통해 카테고리별로 동영상의 등급 분포에 따른 정보를 확인해보도록 하자

위 테이블을 통해 알 수 있는 정보는 다음과 같다.

- 동영상의 가장 많은 카테고리는 영화와 음악이다

- 주로 오후 12~4시에 게시되었다
- 0.15~0.2의 감성 수치를 보였다 (밝은 장르)

- 영화 동영상은 평균적으로 128분이다.
- 음악 동영상은 평균적으로 5분이다.

그렇다면, 1등급 동영상은 어떤 특징을 갖는지 수치를 통해 파악해보자.

마찬가지로, 1등급 변수들의 기본 통계량을 확인해보자.

	category_id	duration	views	likes	dislikes	comment_count	gap_time	senti_mark	publish_time_devide
count	4801.00	4801.00	4801.00	4801.00	4801.00	4801.00	4801.00	4801.00	4801.00
mean	12.29	277.91	42209012.89	865429.52	52529.63	83786.36	1480011.66	0.14	3.50
std	6.22	470.60	43084586.81	848853.51	130433.87	138467.20	854696.25	0.24	1.55
min	1.00	0.00	15188497.00	0.00	0.00	0.00	86400.00	-0.80	1.00
25%	10.00	192.00	19001424.00	368155.00	12104.00	21811.00	777600.00	0.00	2.00
50%	10.00	226.00	26448434.00	571803.00	22375.00	39665.00	1382400.00	0.11	4.00
75%	10.00	263.00	47775206.00	1049132.00	46172.00	79828.00	2160000.00	0.29	5.00
max	29.00	13632.00	424538912.00	5613827.00	1753274.00	1228655.00	3283200.00	0.80	6.00

- 1등급 변수 통계량

변수 통계량의 평균을 보았을 때, 최상위 조회수 영상은 평균 오후 12시~3시30분(3.5) 사이에 올리고, 길이가 약 5분 이내의 영상이며, 인기동영상으로 선정되기까지 약 17일(1480011초) 정도 소요 된다는 사실을 알 수 있었다. 통계량엔 없지만, title과 tags로 추출된 변수까지 합쳐 최종 변수가 선정 되었다.

최종 결정된 변수들 중 등급에 가장 영향을 주는 변수는 무엇 일까?

등급을 가장 잘 구분하는 변수가 등급에 가장 영향을 주는 변수 일 것이다. 등급별 상이한 값을 갖는지 확인하기 위해 가장 극명한 1등급과 9등급에서 변수의 평균값을 비교해보자. 만약 평균 값이 비슷하다면, 등급에 관계 없이 변수 값이 일정하므로 영향을 덜 주는 변수라 생각해도 무방하다.

그러나 각 변수 별 값의 범위 다르기 때문에 변수별로 비교하는 것이 의미가 없을 것이다. 따라서 변수의 가장 큰 값으로 나누어 변수 값이 0~1에 있도록 정규화 한다.

다음은 정규화를 거쳐, 1등급과 9등급에서 갖는 평균 값의 차이 이다.

Out[5]:

	1	9	1등급_9등급 차이
likes	0.15503	0.00009	-0.15494
dislikes	0.03023	0.00002	-0.03021
comment_count	0.06867	0.00009	-0.06858
senti_mark	0.06748	0.07822	0.01074
duration	0.00321	0.00816	0.00495

- 1등급-9등급 평균 차이

연속형 변수 중에 'likes'가 등급간 최대 격차를 보였다. 하지만 1등급과 9등급 간의 차이로만 변수의 영향을 판단하기엔 무리가 있어 보인다.

등급이 변할 때, 변수 값도 변한다면 그 변수는 등급에 영향을 준다고 할 수 있을 것이다. 즉 흐름이 유사한 변수를 찾아 보는 것을 생각해보았다.

변수의 분산과 등급의 분산 간에 차이가 적다면, 등급에 영향을 주는 변수일 것이다. 다른 변수까지 합하여 등급과의 ANOVA(분산분석)을 한 뒤, F-검정의 p-value 이 큰 순서(차이가 없는) 순서대로 순위를 매기면 어떤 변수가 등급과 흐름이 유사한지를 알 수 있을 것이다.

```
X_new = SelectKBest(f_classif, k=i).fit(X, y) # ANOVA를 이용하여 주요 Feature 선정
SFcolumns=X_new.get_support(indices=True) # index를 interger로 추출
```

```
1  등 Index(['likes'], dtype='object')
2  등 Index(['likes', 'comment_count'], dtype='object')
3  등 Index(['likes', 'comment_count', 'category_id_10'], dtype='object')
```

- ANOVA 결과

sklearn library 의 SelectkBest를 이용하여 분산분석을 해보았다. 그 결과 1,2,3등 순서대로 좋아요 수, 댓글의 수, 카테고리가 'music'인 영상이 선정되었다. 음악은 만국 공통어라는 것이 증명이 된 셈이다.
등급의 평균 간의 차이로만 영향을 판단한 것보다 더욱 정교한 결과가 나왔다. 하지만 분산분석의 경우 변수 하나하나를 독립적으로 생각하였다는 가정이 필요하다. 만약 독립적이지 못하다면, 위의 결과는 무용지물인 것이다.

변수의 교호작용을 고려하여 영향도를 보려면 어떻게 해야 할까?

독립성을 증명하는 것도 방법이 될 수 있지만, 좀 더 간단한 방법이 있다. 바로 교호작용을 고려한 모델에서 각 변수의 가중치를 산출하는 것이다. 우리는 의사결정나무를 통해 모델링 한 후 모델로 부터 변수의 영향도를 보기로 하였다.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

tree=DecisionTreeClassifier(criterion='entropy',max_depth=19,random_state=0)
tree.fit(X_train, y_train)
y_predtree=tree.predict(X_test)
```

```
print('변수 영향도')
for k,v in zip(df_ver1.columns,tree.feature_importances_):
    print(str(k)+' : ' +"{:.2f}".format(v))
```

```
변수 영향도
duration : 0.08
likes : 0.33
dislikes : 0.28
comment_count : 0.07
gap_time : 0.06
senti_mark : 0.06
```

- 의사결정나무를 통한 변수 중요도 결과

결과는 흥미로웠다. '댓글 수'보다 영향이 없다고 판단된 '싫어요 수'가 더욱 영향을 끼치는 변수로 밝혀졌다. 이러한 순위 변동의 이유는 변수들 간의 교호작용이 있다는 의미이다. 교호작용이 없다면, ANOVA와 같은 결과여야 할 것이다. 따라서 ANOVA의 결과는 의미가 없다.

조회수에 영향을 주는 변수를 찾기 위한 여행은 평균, 분산, 모델이라는 나라를 거쳤다. 최종적으로 각 방법마다 1등을 했던 'likes'가 조회수에 가장 영향을 미치는 변수였다. 그에 못 지 않게 'dislikes'도 중요한 변수였다.

여기서 잘 생각해봐야 하는 것은 인과관계인데, 조회수가 높으면 당연히 좋아요 수와 싫어요 수도 높지 않을까 하는 생각이 든다. 인과관계를 제외하면, '영상의 길이'가 조회수에 영향을 미치는 진정한 변수라 할 수 있다.

영상의 길이가 짧은 영상이 인기가 좋은 이유는 뭐든 빠르게 결과를 봐야 하고, 느린 것에 금방 질리는 현대사회를 반영한 이유가 아닐까..?

조회수에 큰 영향을 주는 변수는 대부분 조회수로부터 발생되는 변수였다.

영상을 업로드 하기 전,
조회수와 무관한 상태로 얻을 수 있는 데이터로
조회수를 예측할 수 있을까?

변별력 있는 변수도 없는 상황에서 정확한 조회수를 예측하기란 컴퓨터도 힘들 것 이라는 생각이 든다. 아까 사용했던 등급제를 도입하여 컴퓨터에게 어느 정도의 선택지를 주었다. 주관식에서 객관식으로 바뀐 것이다. 어떤 모양의 모델이 가장 정확한지 모르기 때문에 가장 기초적인 선형모델로 컴퓨터를 학습시켜보자. 모델은 대략 다음과 같은 모양 일 것이다. (x는 변수)

$$Y(\text{등급}) = ax_1 + bx_2 + cx_3 \cdots$$

여러 선형 모델 중 단일퍼셉트론을 사용하여 모델을 학습시켜보았다. 그 결과는 다음과 같다.

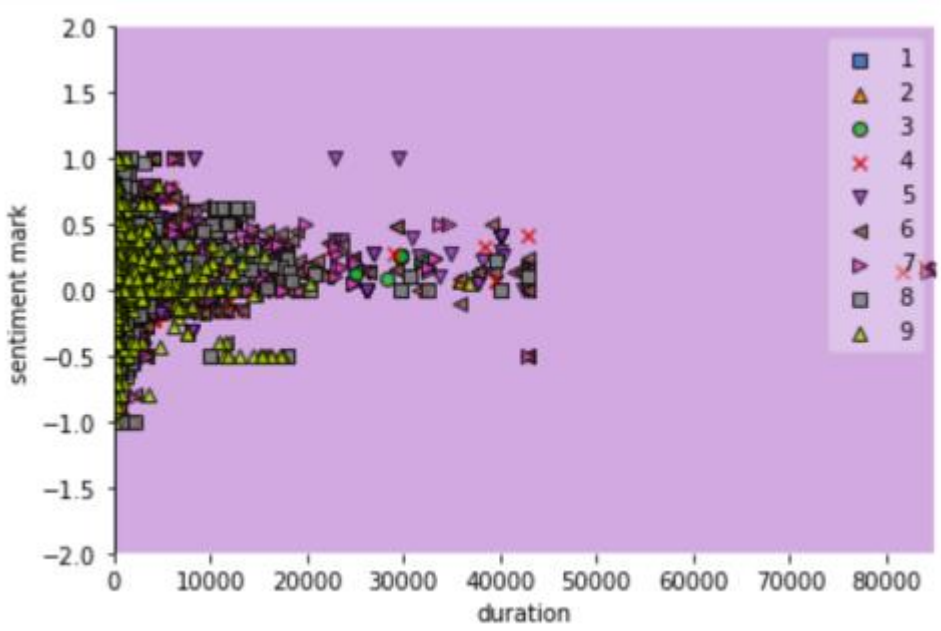
이때, 학습데이터와 실험데이터는 7:3의 비율이다.

```
ppn=Perceptron(max_iter=40, eta0=0.1,random_state=0) #학습 반복수 40, 학습 률 0.1 로 설정하여 모델 생성
ppn.fit(train_X, train_Y) # 모델에 데이터 학습
```

GridSearchCV 라는 library을 사용하여 퍼셉트론의 parameter (학습 반복수, 학습 률)의 최적 값을 산출한 뒤, 학습데이터로 학습시켰다. 그 결과는 아래와 같다.

```
Out[14]: 0.17208374200967294
```

모델이 등급을 맞출 확률은 17%로 아주 낮은 수치가 나왔다. 경황을 알기 위해 그래프를 확인 하였다.



- 단일 퍼셉트론의 그래프(duration-senti_mark)

경황이 없는 그래프가 나왔다. 컴퓨터가 포기한 것일까 코드대로라면 그래프에 모델이 예측한 선이 있어야 한다. 이해심을 발휘하여 선이 없는 이유를 생각해보면 등급의 분포가 한눈에 봐도 단순히 선으로 표현될 수 없다는 것을 알 수 있다. 선형모델로 적합이 되지 않았다고 주저앉을 수 없다. 이제는 비선형모델을 통해서 그 관계를 다시 확인해보도록 하자.

Support Vector Machine(SVM)으로 모델을 적합 시켜 보자. 커널을 이용해 차원을 늘리면 2차원에서 겹쳐 보이는 등급들이 3차원에서 구분되지 않을까 생각한다.

마찬가지 방법으로 GridSearchCV를 이용해 최적의 parameter를 찾아낸다.

```
clf_SVM=SVC()
SVM_SVC=GridSearchCV(clf_SVM, param_grid=
                     {"kernel": ['linear','rbf','poly','sigmoid'],
                      "C":[1,10],
                      "gamma":[0.1,0.5,1,5,10]})

SVM_SVC.fit(X_train_std, y_train.Fault)
```

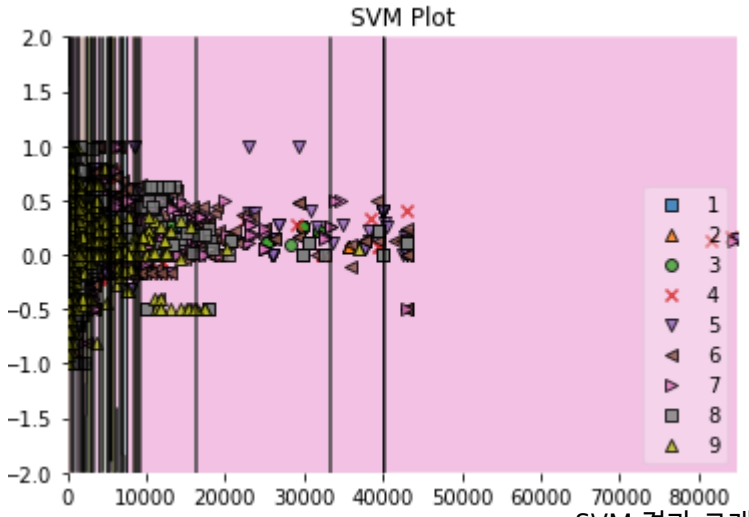
```
svm = SVC(kernel='rbf',C=10, gamma=0.1, verbose=True, random_state=0).fit(trainX, trainY)
```

- GridSearchCV 코드(SVM)

최적의 parameter를 찾는 GridSearchCV를 사용하면 물론 다른 parameter를 사용할 때보다 결과가 좋다. 하지만 큰 차이는 없기 때문에 만약 모델의 성능이 좋지 않다면, GridSearchCV로 성능을 올린다는 생각보단 데이터양이나 질을 다시 생각해볼 필요가 있다. 다시 본론으로 와서 SVM으로 학습시킨 결과 그 정확도는 약 58%였다. 자세한 경황을 알기 위해 마찬가지로 그래프를 그려보았다.

```
svm.score(test_X,test_Y)
```

```
0.5890012513951365
```



- SVM 결과 그래프

아까보단 컴퓨터가 노력한 흔적이 보인다. SVM을 이용해 등급을 예측해보면, 선형모델인 경우보다 41퍼센트 정확한 수치가 나왔다. 하지만 58%라는 정확도는 예측 모델이라고 하기엔 낮은 수치라 생각이 든다. 멈추지 않고 통계적 모델인 의사결정 나무로 데이터를 학습시켜보았다.

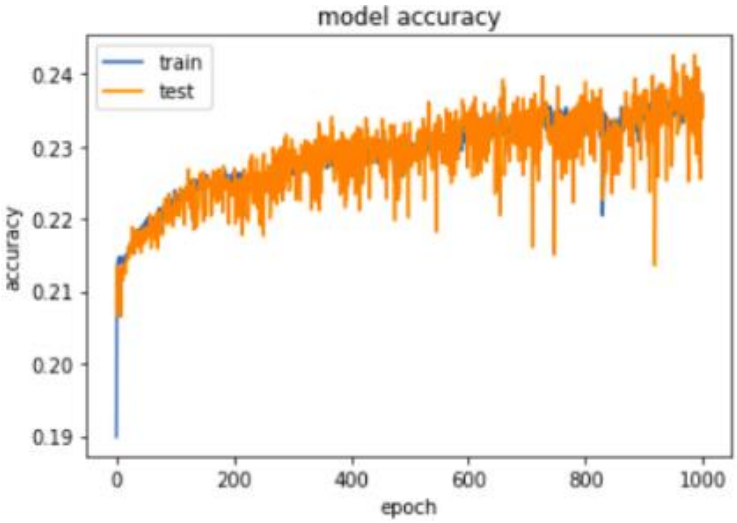
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

tree=DecisionTreeClassifier(criterion='entropy',max_depth=19,random_state=0)
tree.fit(X_train, y_train)
y_predtree=tree.predict(X_test)


print('Accuracy of tree: %.2f' % accuracy_score(y_test, y_predtree)) #전체 정확율
```

```
Accuracy of tree: 0.51
```

- 의사결정 나무 정확도 결과

의사결정나무의 정확도는 51%였다. SVM보다 7%가 낮은 결과이다. 의사결정 나무의 통계적 방식을 바탕으로 예측치를 추론하는 것은 적절하지 못하였다.

더 욕심을 내서 'keras'를 사용해, 은닉층의 수가 10개, 노드가 10개인 신경망 네트워크를 사용하여 딥러닝을 해보았다. (Epoch =1000)



- DNN 학습 수-정확도 그래프

기대했던 DNN(Deep Neural Network)은 23%라는 낮은 예측치를 보여주었다. 변별력이 없는 변수를 먹고 자라서 인지 비실비실하게 컸다. 결국 입력 값 변수의 변별력이 약하기 때문에 80%~90% 이상의 예측 성능을 보여준 모델은 없었다. 다른 모델로 학습시켜도 입력 값이 완벽하지 않기 때문에 높은 예측치를 기대하긴 어려울 것이다.

모델의 정확도를 높이기 위해 어떻게 해야 할까?

영상에 나오는 사람의 수, 남자인지 여자인지 등 영상자체에서 발생하는 데이터라면 더욱 좋은 변수가 될 것 같다는 생각이 들지만 영상처리 기술이 없고, 구글 API를 통해 변수를 늘리고 싶지만, 분석보고서 제출기간을 고려하면 현실적으로 무리이다. 예측 성능을 높이기 위한 다른 방향을 생각해 보았다.

동영상의 조회수를 9등급이 아닌 3등급 (20%,60%,20%)으로 나눈다면, 조회수의 세밀한 구분은 하지 못하더라도 모델의 정확도는 올라 갈 것이다. 쉽게 말해 9지 선다형 문제가 3지 선다형 문제로 되는 것이다. 모델을 통해 동영상의 조회수가 높음/보통/낮음 으로 구분할 수 있을 것이다.

DNN, 의사결정나무, SVM의 결과는 다음과 같다.

- DNN 결과 (67%)

```
HIDDEN_LAYERS_NEURONS = 10
VALIDATION_SPLIT = 0.2


model = Sequential()

model.add(Dense(HIDDEN_LAYERS_NEURONS, kernel_initializer=he_normal(seed=None),
          bias_initializer='zeros', input_shape=(33,)))
model.add(Activation('relu'))

#for layeradd in range(5):
#   model.add(Dense(HIDDEN_LAYERS_NEURONS))
#   model.add(Activation('relu'))
for l in range(9):
    model.add(Dense(HIDDEN_LAYERS_NEURONS))
    model.add(Activation('relu'))

model.add(Dense(3))
model.add(Activation('softmax'))
model.summary()
```

```
score = model.evaluate(test_X, test_Y, verbose=True)
```

```
29567/29567 [==============================] - 1s 20us/step
```

```
print(score[1])
```

```
0.6680082524254594
```

- 의사결정나무 결과 (64%)

```
tree=DecisionTreeClassifier(criterion='entropy',max_depth=20,random_state=0)
tree.fit(train_XX, train_YY)# 훈련데이터로 의사결정 나무 모델 구축
```

```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=20,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=0,
            splitter='best')
```

```
tree.score(test_XX,test_YY)
```

```
0.6410525247742416
```

- SVM 결과 (81%)

```
svm = SVC(kernel='rbf',C=10, gamma=0.1, verbose=True, random_state=0).fit(trainX, trainY)
```

```
[LibSVM]
```

```
svm.score(test_X,test_Y)
```

```
0.8129671593330402
```

등급을 3개로 나누어 학습시킨 결과 정확도를 81%까지 끌어 올릴 수 있었다. 이제 예측모델의 구색은 갖추었다.

유튜브에 영상을 올리기 전, 이 영상이 사람들에게 인기가 있을지, 없을지, 보통일지 정도는 81%확률로 구분할 수 있다. 더 많은 데이터와 변수, 시간에 대한 아쉬움을 남기며 분석을 마친다.

"미래의 유튜버에게 도움이 될 분석이길 바라며..."