유튜브 영상 댓글 분석을 통한 장원영 평판 보고서

Text Mining 2022-2
담당 교수: 안정용
통계학과 201822037 한민주

목차

서론

본론

결론

붙임

서론

1. 주제 선정 이유

MZ세대 아이콘으로 불리는 장원영은 팬층이 두텁지만 최근에는 악성댓글이 늘어나고 있다. 19세의 어린 나이로 대중들에게 몸살을 앓고 있는 이유가 무엇인지 궁금했다. 따라서 장원영이 나오는 유튜브 영상 댓글을 분석하여 장원영의 인기 요인을 분석하고 어떤 논란이 있는지 알아보고자 한다.

2. 인물 소개

2004년생으로 현재 나이 19세이다. 2018년 서바이벌 오디션 프로듀스48에 참가해 최종 우승을 하였고 이후 걸그룹 IZ*ONE의 센터로 2021년 4월 29일까지 활동하였다. 현재는 스타쉽 엔터테인먼트 소속 걸그룹 IVE의 멤버이다. IVE는 2021년 12월 1일에 데뷔하여 앨범 'ELEVEN', 'LOVE DIVE', 'After Like'를 발매하였다. 발매하는 족족 히트를 쳐 최근에는 2022 MAMA 어워즈에서 신인상과 더불어 대상 중 하나인 '올해의 노래상'을 받았다.

그룹 활동뿐만 아니라 개인 활동이 활발하다. 단독 광고만 미우미우, 키르시, 고스피어, 이니스프리 등을 포함하여 9개이다. ELLE, VOGUE, BAZAAR 등 각종 화보를 섭렵하였고 현재 뮤직뱅크 MC를 맡고 있다. 'MZ세대 아이콘', '초통령', '갓기', '노벨 아이돌상' 등의 별명을 가지고 있으며, 바쁜 스케줄 와중에도 언제나 밝은 성격을 보여 큰 인기를 끌고 있다.

본론

1. 데이터 설명

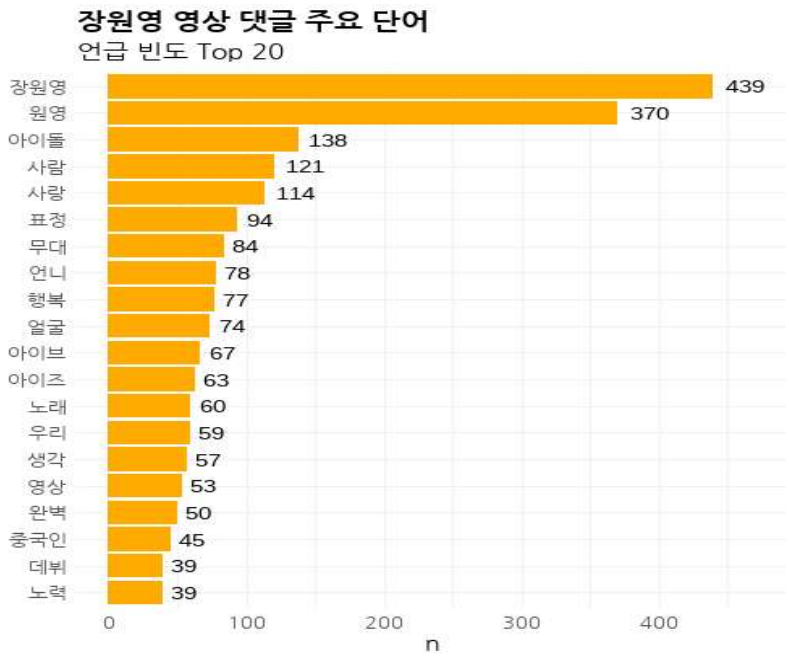
데이터는 유튜브에 장원영을 검색하면 나오는 7개 영상의 댓글을 R selenium으로 크롤링하여 생성되었다. 중립적인 3개의 영상에서 1,033개의 댓글을, 선의적인 2개의 영상에서 969개의 댓글을, 악의적인 2개의 영상에서 496개의 댓글을 크롤링하여 총 2,498개의 댓글을 분석에 사용하였다. 모든 분석은 R 프로그램을 사용하였고 R code는 붙임에 첨부하였다.

2. 데이터 분석 결과

01) WordCloud

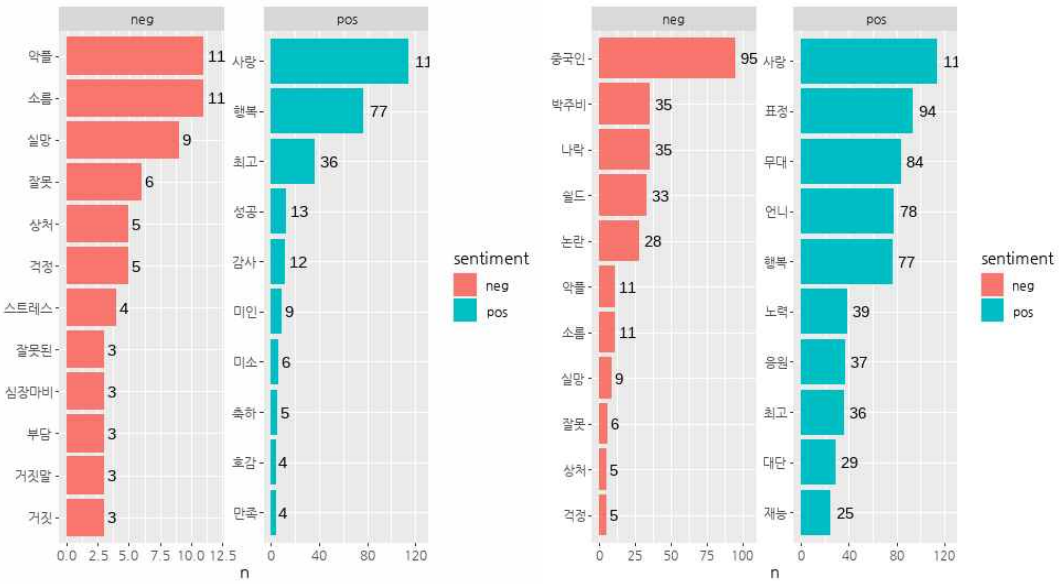
댓글 원문을 전처리 후, 명사 기준으로 토큰화하여 워드 클라우드를 만들었다. 장원영, 아이돌, 얼굴, 표정, 무대, 아이브, 아이즈원 등이 높은 빈도로 나타났고, 그 밖에 검정고시, 머리스타일, 화교, 입꼬리, 성형, 몸무게 등 장원영을 둘러싸고 있는 키워드들을 확인할 수 있다.

02) 댓글 주요 단어



장원영 관련 유튜브 영상 댓글에 달린 단어 중 빈도가 가장 높은 20개를 막대그래프로 만들었다. 아이돌, 사랑, 행복, 완벽 등 긍정적인 단어가 많았고 중국인 논란이 있어 중국인이 18번째로 빈도가 높은 단어를 차지했다.

03) 단어별 감정 분석

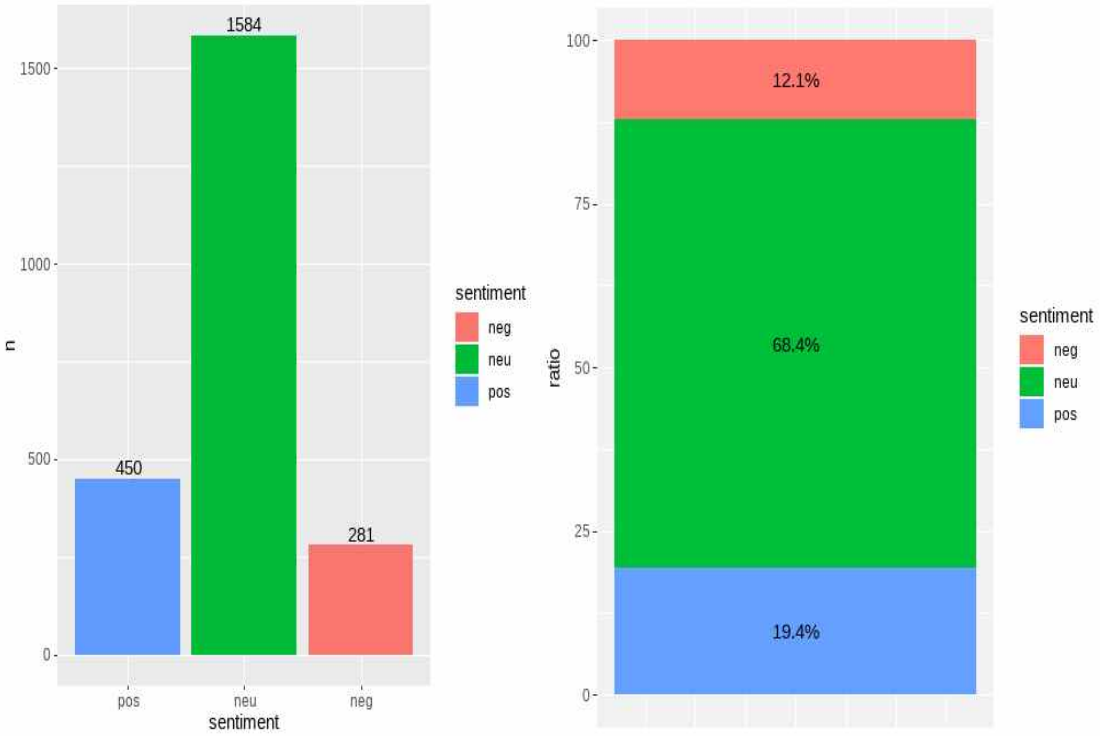


군산대학교 소프트웨어융합공학과에서 만든 'KNU 한국어 감성 사전'을 이용하여 댓글을 감

정 분석하였다. 댓글을 단어 기준으로 토큰화하고 감정 점수를 부여한 후, 긍정 단어와 부정 단어 중 가장 자주 사용된 단어를 10개씩 추출하여 막대그래프를 만들었다. 왼쪽은 감성 사전 원본을 사용하여 감정 점수를 부여한 막대그래프이고, 오른쪽은 중립 단어에 대해 수정한 감성 사전을 사용하여 감정 점수를 부여한 막대그래프이다.

부정적 단어를 보면 장원영이 중국인이라는 '중국인 논란'과 아이브 데뷔조였던 박주비를 괴롭혀 강제 탈퇴시켰다는 '박주비 논란' 그리고 이에 대해 극성팬들이 쉴드(방어)를 하며 논란을 묻히고 있다는 내용의 댓글에 사용된 단어임을 예상할 수 있다. 긍정적 단어를 보면 장원영이 사랑스럽고 무대를 잘하고 행복했으면 좋겠다는 칭찬과 응원 댓글에 사용된 단어임을 예상할 수 있다.

04) 문장별 감정 분류



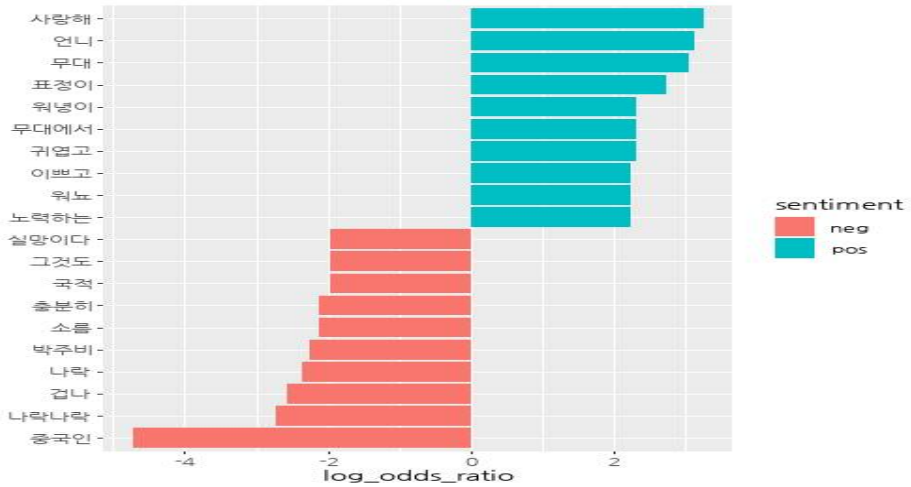
단어별 감정 점수를 합산하여 문장별 감정 점수를 구하였다. 장원영 관련 영상에는 중립적인 댓글이 68.4%, 긍정적인 댓글이 19.4%, 부정적인 댓글이 12.1%였다. 부정적 댓글보다는 긍정적 댓글이 많지만 그 차이는 7.3%로 크지 않았다.

감정	댓글 원문	감정점수
긍정	**"노력하며 즐기는 천재는 이길 수 없다의 표본 매 컴백 때마다 더 예뻐지고 더 늘어서 오는데 무대 위에서 말도 안되게 빛나 표정이며 춤선이며 너무 예술이고 무대하는 동안 정말 행복해보여 무대도 너무 잘하는 갓기가 끝나고 팬들이랑 소통하는데 저 사랑이 가득 담긴 눈을 보면 장원영을 사랑할 수밖에 없음"**	+16

부정	**“장원영 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락”**	-46

가장 긍정적인 댓글과 가장 부정적인 댓글의 점수는 각각 +16점, -46점이었다. 가장 긍정적인 댓글에는 장원영을 좋아하는 이유가 구체적으로 명시된 반면에 가장 부정적인 댓글에는 장원영을 싫어하는 이유가 없었다. 인터넷상에서 잘나가는 연예인을 시기 질투하는 전형적인 키보드 워리어가 단 댓글로 예상된다.

05) 로그 오즈비 막대그래프

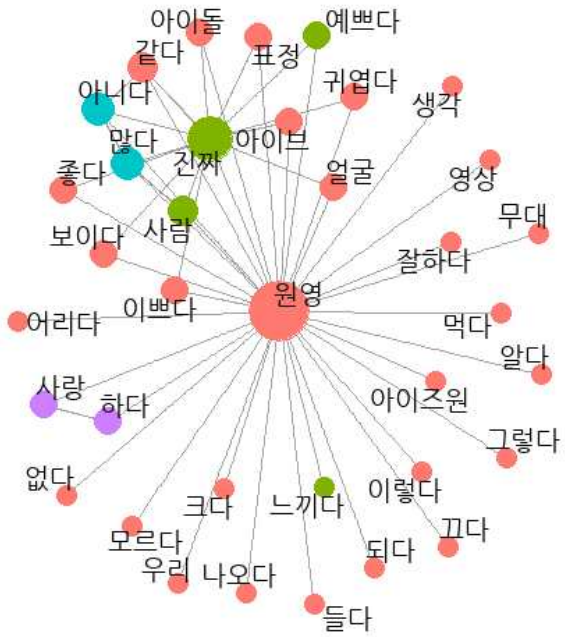


로그 오즈비를 이용해 긍정 댓글과 부정 댓글에 상대적으로 어떤 단어가 자주 사용되었는지 분석하였다. 긍정 댓글에는 언니, 워넹이, 워뇨 등 장원영의 애칭이 상대적으로 자주 사용되었고, 부정 댓글에는 국적과 박주비 논란에 관련된 단어, 소름, 실망 등의 단어가 상대적으로 자주 사용되었다. 밑의 표는 오즈비가 큰 단어의 원문을 6개 추출한 것이다.

단어	댓글 원문	로그 오즈비
사랑해	**“성적도 얼굴도 실력도 성격도 다 좋은데 진짜 자존감까지 높다는게 진짜 너무 부럽고 본받고싶다 언니 사랑해욯”**	+3.26
언니	**“정말 대단한게 초긍정 마인드에 대게 맘상할 악플에도 신경 안쓰는 방탄마인드가 어린 나이에도 존경스러워요 그리고 케이콘 위해 방문한 프프공항서 팬의 언니에게 보여준 잔망스런 애교와 무대서의 애교는 프듀 이후 더욱 발전한 팬서비스를 보여주어 정말”**	+3.13
무대	**“장원영 죽자고 미워하는 애들한테 편견있음 타고난것도 있지만 무대 올라가기 전에 사소한 표정이랑 손짓 하나하나까지 다 신경쓰고 연습하던데 그러니까 당연히 다른 애들보다 눈에 잘띄고 센터 역할도 할수 있는**	+3.04

	거라고 봄 난 얘가 무대하면서 한번도 센터로서 역할을 제대로 못하는건 본적이 없는거같다”	
중국인	“님들 장원영아니고 장위엔잉이에요 장원영으로 부르면 한국인같잖아요 그동안 찐중국인이 국적 숨기고 한국인행세하며 잘해먹었지”	-4.73
나락나락	“장원영 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락 나락나락”	-2.74
겁나	“와 장원영 진짜 실망이다 장원영 지금이라도 박주비한테 사과하던가 탈퇴하던가 하 장원영한테 쌓였던 논란 그냥 논란이 아니네 박주비 겁나 불쌍하다 장원영 진짜 자기 혼자 편하겠다고 내쫓는거 봐 드디어 인성을 들어내네 장원영이 최애였는데 이젠 장원영이 젤 싫고 가을 유진 레이 리즈 이서 주비가 젤 좋아 장원영 완전 실망이야”	-2.57

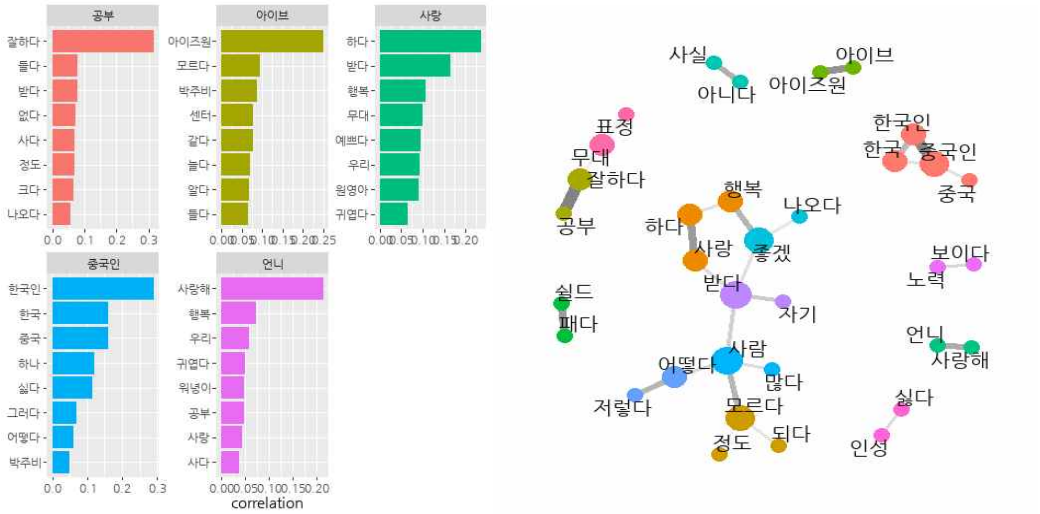
06) 동시 출현 네트워크 그래프



댓글을 품사를 기준으로 토큰화하여 동시 출현 빈도를 구한 후, 단어의 관계를 네트워크 형태로 만들었다. 네트워크가 너무 복잡하지 않도록 18회 이상 사용된 단어만 추출하였고, 연결중심성과 커뮤니티도 함께 표현하였다.

1번 커뮤니티로 분류된 ‘원영’은 ‘아이즈원’, ‘아이브’, ‘귀엽다’, ‘아이돌’ 등과 연결중심성이 높았고, 2번 커뮤니티에 분류된 단어는 ‘진짜’, ‘사람’, ‘예쁘다’로 외모 칭찬과 관련된 단어끼리 연결중심성이 높았다. 3번 커뮤니티는 ‘아니다’, ‘많다’로 ‘진짜’와 연결 중심성이 높았고, 4번 커뮤니티는 ‘사랑’, ‘하다’로 ‘원영’과 연결 중심성이 높았다.

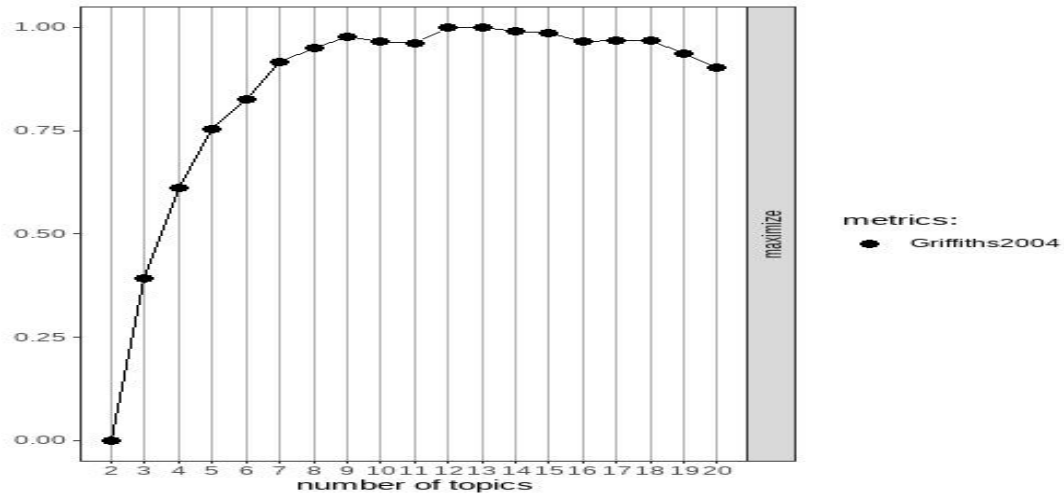
07) 파이계수를 이용한 네트워크 그래프



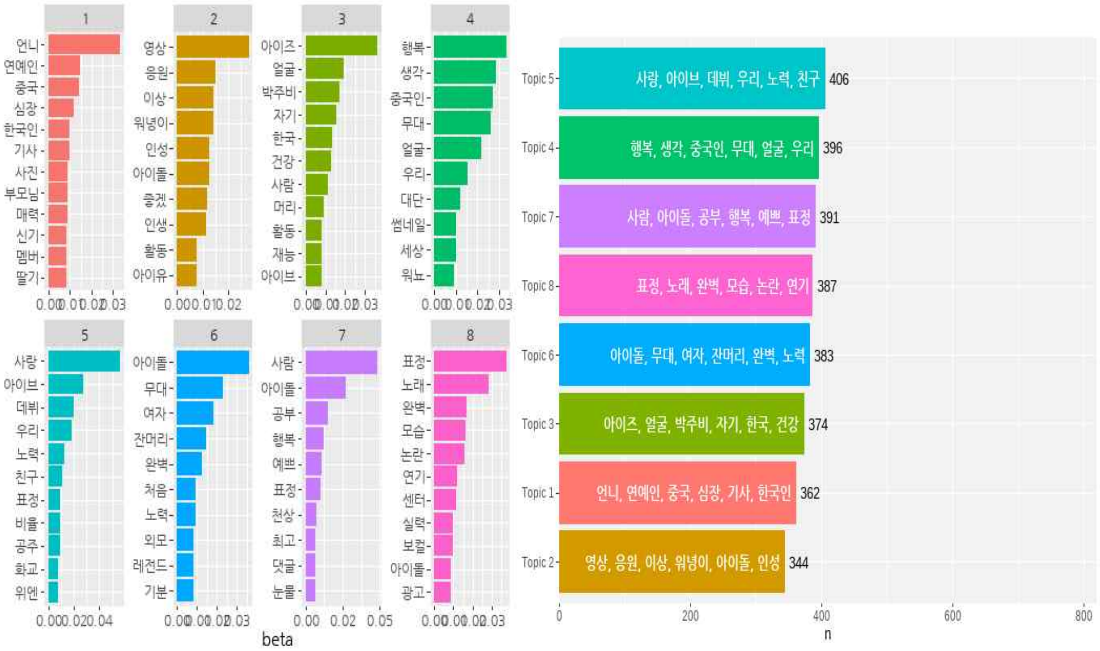
특정 단어와 상대적으로 자주 함께 사용된 단어를 알아보기 위해 댓글의 파이계수를 구했다. 공부, 아이브, 사랑, 중국인, 언니를 관심 단어 목록으로 지정하였고 막대그래프를 만들었다. 장원영이 공부를 잘하고, 사랑받고 있으며, 어린 팬이 많음을 알 수 있다. 또한, 아이브와 관련하여 박주비 논란에 대한 악성댓글이, 개인에 대해서는 중국인 논란에 대한 악성댓글이 많음을 알 수 있다.

오른쪽 그래프는 파이계수를 이용해 만든 네트워크 그래프이며 연결중심성과 커뮤니티를 추가하였다. 자주 함께 사용된 단어쌍을 보니 '노력이 보인다', '인성이 싫다'/'쉴드 패다', '사실 아니다'와 같은 팬과 안티팬의 대립이 눈에 띄었다.

08) 토픽 모델링



LDA 모델을 활용하여 토픽 모델링을 하기 위해 문서별 단어 빈도를 구한 후, 문서 단어 행렬(DTM)을 만들었다. DTM은 8,285 문서 x 6,347,330 단어로 구성되었다. LDA 모델을 만들기 전 최적의 토픽 수를 결정하기 위해 하이퍼 파라미터 튜닝을 하였다. 위 그래프를 보면 토픽 수가 8개가 될 때까지는 성능 지표가 큰 폭으로 증가하다가 그 이후로 약간씩 증가 또는 등락을 반복한다. 따라서 토픽 수를 8개로 결정하였다.

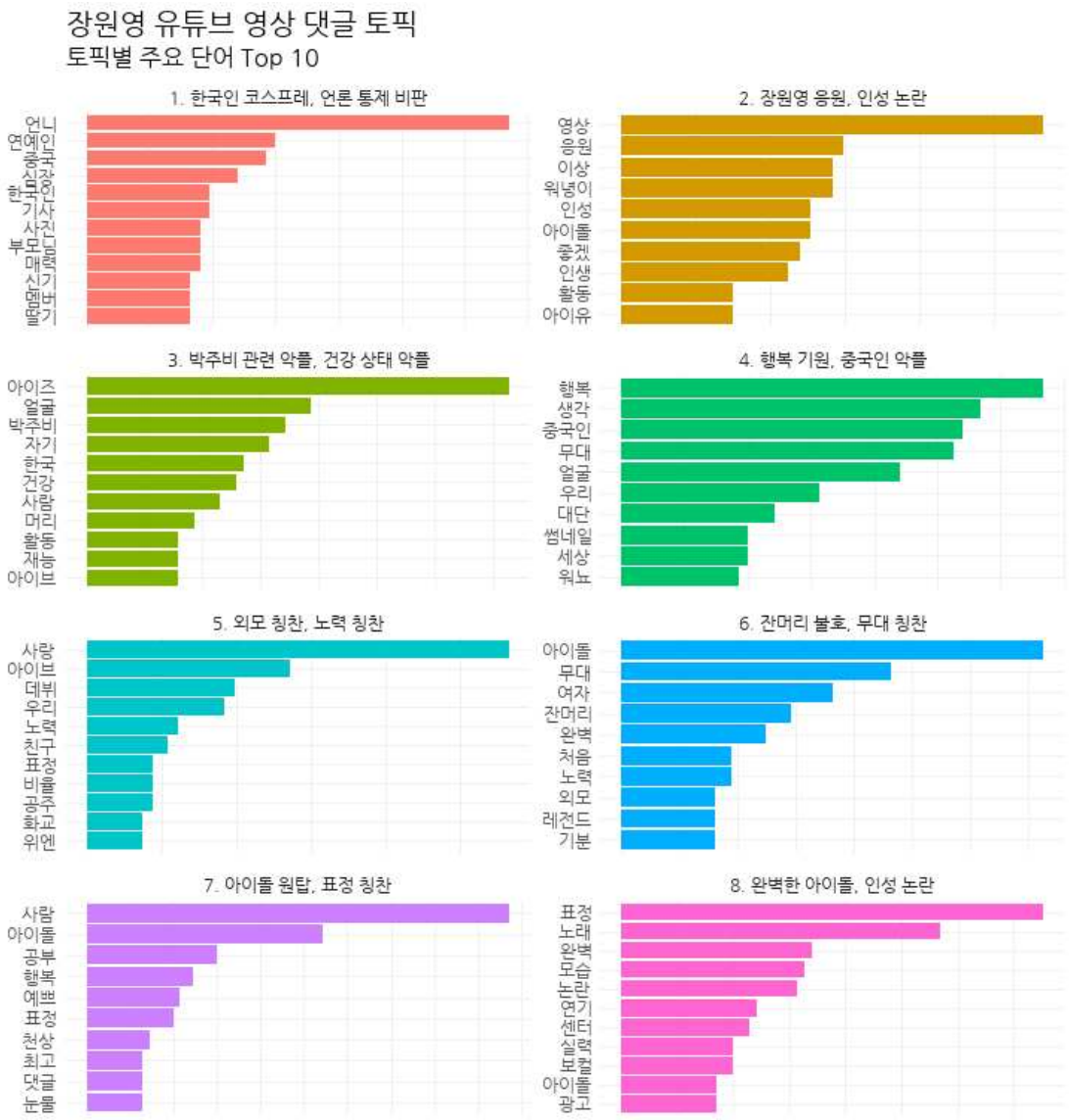


LDA 모델을 생성 후, 단어가 토픽에 등장할 확률(beta)을 기준으로 왼쪽의 막대그래프를 만들었다. 각 토픽에 등장할 확률이 가장 높은 단어는 모두 긍정적인 단어였지만, 토픽1에는 '중국', 토픽2에는 '아이유', 토픽3에는 '박주비', 토픽4에는 '중국인', 토픽5에는 '위엔', 토픽8에는 '논란'과 같은 부정적인 단어가 섞여 있다.

단어	댓글 원문	beta
아이유	**"아니 아이유를 대체 왜 따라하는거야 아이유는 인성 실력 등 아무나 따라 갈 수 없는 어나더레벨인데"**	0.00752
위엔	**"박주비가 중국 대표 미녀 배우 장쯔이랑 닮아서 장위엔링이 질투한 듯"**	0.00756

댓글 원문을 보아 '아이유'는 장원영이 아이유를 따라 한다는 논란에 대한 댓글에서 비롯된 단어이고, '위엔'은 국적 논란과 관련한 조롱 댓글에서 비롯된 단어이다.

오른쪽 그림은 문서가 토픽에 등장할 확률(gamma)을 기준으로 만든 막대그래프이다. 406개로 가장 많은 댓글이 분류된 토픽5는 빈도가 높은 단어가 모두 긍정적인 것으로 보아 선의적인 댓글로 구성되어있을 것으로 추측된다. 또한 8개의 토픽 중 부정적인 단어로만 구성된 토픽이 없는 것으로 보아 악성댓글만 분류된 토픽은 없었다.



문서가 토픽에 등장할 확률(gamma)을 기준으로 토픽 이름을 붙이고 주요 단어를 막대그래프로 만들어 토픽의 특징을 살펴보았다. 이를 통해 장원영은 외모, 노력, 표정 등으로 인기를 얻어 완벽한 아이돌이라는 평가를 받고 있음과 동시에 국적, 인성, 스타일링, 박주비와 관련하여 부정적인 평가를 받고 있음을 알 수 있다. 또한 이러한 부정적인 평가는 기사화되지 않아 대부분의 대중들이 모르고 있음을 토픽1의 주요 단어인 '기사'의 댓글 원문이 비판하고 있었다.

결론

1. 요약

걸그룹 아이브 멤버 장원영은 '초통령', 'MZ세대 아이콘'이라는 별명을 가질 정도로 큰 인기를 끌고 있다. 그러나 인기만큼 최근에는 악성댓글과 각종 논란에 시달리고 있다.

장원영과 관련된 유튜브 영상 댓글을 분석하여 보니 장원영의 인기 요인은 얼굴이 예쁘고, 무대에서 표정을 잘 쓰고, 타고난 아이돌처럼 보이는 끼를 가지고 있으며, 공부를 잘하고, 항상 노력하는 것에 있었다.

반면 부정적인 평판의 원인은 중국인이라는 국적 논란과 연습생 박주비를 괴롭혀 아이브에서 강제 탈퇴시켰다는 인성 논란, 아이유를 따라한다는 스타일링(잔머리) 논란에 있었다. 또한 너무 말라서 건강에 이상이 있어 보인다는 걱정을 빙자한 악성댓글도 있었다.

그러나 악성댓글은 전체 댓글의 12.1%밖에 되지 않았고, 아이브가 최근에 받은 상이나 광고 촬영, 화보 촬영 등 잠도 자지 못할만큼 개인 스케줄이 많다는 점을 고려할 때, 장원영이 대세 아이돌이며 장원영을 좋아하는 사람이 더 많다는 점을 인정할 수밖에 없어 보인다.

2. 제언

댓글을 분석하여 보니 장원영의 악성댓글에서 가장 많이 언급되는 국적 논란, 박주비 논란에 대해서 소속사가 언론을 통제하고 있다는 비판이 있었다. 소속사가 무작정 해당 논란에 대한 기사화를 막을 것이 아니라 정확한 해명을 한다면 악성댓글의 비율이 크게 줄어들 것으로 보인다.

붙임

1. 출처

● 데이터 출처
장원영, 당신이 몰랐던 15가지 사실 https://www.youtube.com/watch?v=Jod8j5PHBw8
소고기 10인분 먹는 장원영의 119가지 https://www.youtube.com/watch?v=i6r8c45cfQU
장원영, 12세~19세까지 성장 과정 https://www.youtube.com/watch?v=h08k0PqjV3w
10분동안 원영이가 심장 두드리는 영상 https://www.youtube.com/watch?v=alGJ-iCQsX8
[입덕직캠] 아이브 장원영 직캠 https://www.youtube.com/watch?v=433HGrqXk_k
장원영 저격한 박주비.. https://www.youtube.com/watch?v=L9Noe94JS3I
장원영 아이유 따라하다가 심각해진 탈모 https://www.youtube.com/watch?v=ZBaZJJIMxIY

● 사진 출처
https://www.topstarnews.net/news/articleView.html?idxno=14691601
https://www.mhns.co.kr/news/articleView.html?idxno=531975

● 인물 소개 참고
https://namu.wiki/w/%EC%9E%A5%EC%9B%90%EC%98%81

2. R code

```
# TM final 201822037 통계학과 한민주 R code
# 셀레니움 환경조성(명령창)
# cd C:\selenium
# java -Dwebdriver.gecko.driver="geckodriver.exe" -jar selenium-server-standalone-4.0.0-alpha-1.jar -port 4445

# 패키지
library(httr)
library(rvest)
library(tidytext)
library(RSelenium)
library(dplyr)
library(stringr)
library(textclean)
library(KoNLP)
library(scales)
library(ggplot2)
library(readr)
library(tidyr)
library(widyr)
library(tidygraph)
library(ggraph)
```

```
library(showtext)
library(topicmodels)
library(ldatuning)
library(ggwordcloud)

# 디렉토리 지정(생략)

# 웹 크롤링
driver <- rsDriver(browser=c("chrome"), chromever="108.0.5359.71")
remote_driver <- driver[["client"]]
remote_driver$open()
remote_driver$navigate("https://www.youtube.com/watch?v=Jod8j5PHBw8")

# 재생
btn = remote_driver$findElement(using = "css selector" ,
                                value = '.html5-main-video')
btn$clickElement()

# 스크롤

scroll = function(start, end){
  remote_driver$executeScript(paste0("window.scrollTo(",start,",",end,")"))
}

scroll(2500,10000) ; scroll(10000,20000)
scroll(20000,25000) ; scroll(25000,50000)
scroll(50000,100000) ; scroll(100000,5000000)
scroll(500000,1000000) ; scroll(1000000,1500000)
scroll(1500000,2000000) ; scroll(2000000,2500000)
scroll(2500000,3000000) ; scroll(3000000,3500000)

# 댓글 가져오기
res <- remote_driver$getPageSource() %>% `[[`(1)
html = res %>% read_html()
comment1 = html %>% html_nodes('#content-text') %>% html_text()

## (크롤링-> 댓글) 함수 만들기
crawling = function(url){
  # url로 이동
  remote_driver$navigate(url)
  # 스크롤
  scroll(2500,10000) ; scroll(15000,20000)
  scroll(20000,25000) ; scroll(25000,30000)
  scroll(30000,35000) ; scroll(30000,40000)
  scroll(40000,45000) ; scroll(45000,50000)
  scroll(50000,55000) ; scroll(55000,60000)
  scroll(60000,65000) ; scroll(65000,70000)
  # 댓글 가져오기
  res <- remote_driver$getPageSource() %>% `[[`(1)
  html = res %>% read_html()
  comment = html %>% html_nodes('#content-text') %>% html_text()
  return(comment)
}

# comment2
comment2 = crawling("https://www.youtube.com/watch?v=i6r8c45cfQU" , "comment2")
# comment3
comment3 = crawling("https://www.youtube.com/watch?v=L9Noe94JS3I" , "comment3")
# comment4
```

```
comment4 = crawling("https://www.youtube.com/watch?v=alGJ-iCQsX8" , "comment4")
# comment5
comment5= crawling("https://www.youtube.com/watch?v=h08k0PqjV3w" , "comment5")
# comment6
comment6 = crawling("https://www.youtube.com/watch?v=433HGrqXk_k" , "comment6")
# comment7
comment7 = crawling("https://www.youtube.com/watch?v=ZBaZJJIMxIY" , "comment7")

comment_ = c(comment1,comment2,comment3,comment4,comment5,comment6,comment7)

# csv 파일로 내보내기
write.csv(comment_ ,
          file = "C:/Users/user/Downloads/comment.csv")

#########################################################################

# data 불러오기
raw_comment = read.csv("C:/Users/user/Downloads/comment.csv")

names(raw_comment) = c("id" , "reply")
glimpse(raw_comment)

# 전처리
comment = raw_comment %>%
  filter(str_count(reply, " ") >= 1) %>%                    # 띄어쓰기 1개 이상 추출
  mutate(reply_raw = str_squish(replace_html(reply)),       # 원문 보유
         reply = str_replace_all(reply, "[^가-힣]", " "),   # 한글만 남기기
         reply = str_squish(reply))                         # 중복 공백 제거


# 명사 기준 토큰화
word_noun <- comment %>%
  unnest_tokens(input = reply,
                output = word,
                token = extractNoun,
                drop = F)

# 단어 빈도 구하기
frequency <- word_noun %>%
  count(word, sort = T) %>%     # 단어 빈도 구해 내림차순 정렬
  filter(str_count(word) > 1)  # 두 글자 이상만 남기기

# 상위 단어 추출
frequency %>%  head(40)

# 불용어 목록 생성
stopword_noun <- c("진짜", "해서", "이거", "하나", "해","하게")

# 유의어 처리
frequency  = frequency %>% mutate(word = ifelse(str_detect(word , "박주") , "박주비" ,word))

# 폰트
font_add_google(name = "Black Han Sans", family = "blackhansans")
showtext_auto()

# 1) 워드 클라우드
ggplot(frequency,
       aes(label = word,
           size = n,
```

```
            col = n)) +
  geom_text_wordcloud(seed = 1234,
                      family = "nanumgothic") +  # 폰트 적용
  scale_radius(limits = c(3, NA),
               range = c(3, 30)) +
  scale_color_gradient(low = "#66aaf2",
                       high = "#004EA1") +
  theme_minimal()

# 2) 주요 단어 목록 만들기
top20_noun <- frequency %>%
  filter(!word %in% stopword_noun) %>%
  head(20)

top20_noun

# 글씨체
font_add_google(name = "Nanum Gothic", family = "nanumgothic")
showtext_auto()

ggplot(top20_noun, aes(x = reorder(word, n), y = n)) +
  geom_col(fill = "orange") +
  coord_flip() +
  geom_text(aes(label = comma(n, accuracy = 1)), hjust = -0.3) +
  scale_y_continuous(limits = c(0, 470)) +

  labs(title = "장원영 영상 댓글 주요 단어",
       subtitle = "언급 빈도 Top 20",
       x = NULL) +

  theme_minimal() +
  theme(text = element_text(family = "nanumgothic", size = 12),
        plot.title = element_text(size = 14, face = "bold"),        # 제목 폰트
        plot.subtitle = element_text(size = 13))                    # 부제목 폰트


# 3) 감정분석
dic <- read_csv("C:/Users/user/Downloads/knu_sentiment_lexicon.csv") # 감정사전

glimpse(comment)

# 감정 점수 부여
word_comment <- word_noun %>%
  mutate(word = ifelse(str_detect(word , "박주") , "박주비" ,word)) %>%
  left_join(dic, by = "word") %>%
  mutate(polarity = ifelse(is.na(polarity), 0, polarity))

word_comment %>%
  select(word, polarity)

# 감정 분류
word_comment <- word_comment %>%
  mutate(sentiment = ifelse(polarity ==  2, "pos",
                            ifelse(polarity == -2, "neg", "neu")))

word_comment %>% count(sentiment)

# 긍정,부정 단어 top10
top10_sentiment <- word_comment %>%
```

```
  filter(sentiment != "neu") %>%
  filter(!word %in% stopword_noun) %>%
  filter(str_count(word) > 1) %>%
  count(sentiment, word) %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10)

top10_sentiment

ggplot(top10_sentiment, aes(x = reorder(word, n),
                            y = n,
                            fill = sentiment)) +
  geom_col() +
  coord_flip() +
  geom_text(aes(label = n), hjust = -0.3) +
  facet_wrap(~ sentiment, scales = "free") +
  scale_y_continuous(expand = expansion(mult = c(0.05, 0.15))) +
  labs(x = NULL) +
  theme(text = element_text(family = "nanumgothic"))

# 중립 단어 중 긍정/부정 단어 추가
top_neu <- word_comment %>%
  filter(sentiment == "neu") %>%
  count(sentiment, word) %>%
  filter(str_count(word) > 1) %>%
  slice_max(n, n = 40)

top_neu # 중립 단어 확인

word_comment %>%
  filter(str_detect(reply, "표정")) %>%
  select(reply)

word_comment %>%
  filter(str_detect(reply, "무대")) %>%
  select(reply)

word_comment %>%
  filter(str_detect(reply, "언니")) %>%
  select(reply)

word_comment %>%
  filter(str_detect(reply, "중국인")) %>%
  select(reply)

word_comment %>%
  filter(str_detect(reply, "쉴드")) %>%
  select(reply)

word_comment %>%
  filter(str_detect(reply, "논란")) %>%
  select(reply)

new_dic <- dic %>%
  mutate(polarity = ifelse(word %in% "쉴드", -2, polarity))

new_dic= new_dic %>% add_row(word = c("중국인", "나락", "논란","표정", "무대", "언니", "응원")
                             , polarity = c(-2,-2,-2,2,2,2,2))
```

```
new_dic= new_dic %>% add_row(word = c("노력", "대단","박주비","중국","재능")
                             , polarity = c(2,2,-2,-2,2))

new_dic %>% filter(word %in% c("중국인", "나락", "쉴드", "논란"))
new_dic %>% filter(word %in% c("표정", "무대", "언니", "응원"))
new_dic %>% filter(word %in% c("노력", "대단","박주비","중국","재능"))

# 감정 점수 부여
word_comment <- word_noun %>%
  mutate(word = ifelse(str_detect(word , "박주") , "박주비" ,word)) %>%
  mutate(word = ifelse(str_detect(word , "중국") , "중국인" ,word)) %>%
  left_join(new_dic, by = "word") %>%
  mutate(polarity = ifelse(is.na(polarity), 0, polarity))

# 감정 분류
word_comment <- word_comment %>%
  mutate(sentiment = ifelse(polarity ==  2, "pos",
                            ifelse(polarity == -2, "neg", "neu")))

word_comment %>% count(sentiment)

# 긍정,부정 단어 top10
top10_sentiment <- word_comment %>%
  filter(sentiment != "neu") %>%
  filter(!word %in% stopword_noun) %>%
  filter(str_count(word) > 1) %>%
  count(sentiment, word) %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10)

top10_sentiment

ggplot(top10_sentiment, aes(x = reorder(word, n),
                            y = n,
                            fill = sentiment)) +
  geom_col() +
  coord_flip() +
  geom_text(aes(label = n), hjust = -0.3) +
  facet_wrap(~ sentiment, scales = "free") +
  scale_y_continuous(expand = expansion(mult = c(0.05, 0.15))) +
  labs(x = NULL) +
  theme(text = element_text(family = "nanumgothic"))


# 4) 댓글(문장)별 감정 점수
score_comment <- word_comment %>%
  group_by(id, reply) %>%
  summarise(score = sum(polarity)) %>%
  ungroup()

score_comment %>%
  select(score, reply)

# 긍정 댓글
score_comment %>%
  select(score, reply) %>%
  arrange(-score)

# 부정 댓글
```

```
score_comment %>%
  select(score, reply) %>%
  arrange(score) %>% pull() %>% head()


# 감정 댓글 비율
frequency_score <- score_comment %>%
  count(sentiment) %>%
  mutate(ratio = n/sum(n)*100)

frequency_score

# 막대 그래프 만들기
ggplot(frequency_score, aes(x = sentiment, y = n, fill = sentiment)) +
  geom_col() +
  geom_text(aes(label = n), vjust = -0.3) +
  scale_x_discrete(limits = c("pos", "neu", "neg"))


# 누적막대 비율
frequency_score$dummy <- 0
frequency_score

ggplot(frequency_score, aes(x = dummy, y = ratio, fill = sentiment)) +
  geom_col() +
  geom_text(aes(label = paste0(round(ratio, 1), "%")),
            position = position_stack(vjust = 0.5)) +
  theme(axis.title.x = element_blank(),   # x축 이름 삭제
        axis.text.x  = element_blank(),   # x축 값 삭제
        axis.ticks.x = element_blank())   # x축 눈금 삭제


comment <- score_comment %>%
  unnest_tokens(input = reply,              # 단어 기준 토큰화
                output = word,
                token = "words",
                drop = F) %>%
  filter(str_detect(word, "[가-힣]") &   # 한글 추출
           str_count(word) >= 2)            # 두 글자 이상 추출

# 감정 및 단어별 빈도 구하기
frequency_word <- comment %>%
  filter(str_count(word) >= 2) %>%
  mutate(word = ifelse(str_detect(word , "중국") , "중국인" ,word)) %>%
  mutate(word = ifelse(str_detect(word , "논란") , "논란" ,word)) %>%
  mutate(word = ifelse(str_detect(word , "언니") , "언니" ,word)) %>%
  count(sentiment, word, sort = T)

# 긍정 댓글 고빈도 단어
frequency_word %>%
  filter(sentiment == "pos")

# 부정 댓글 고빈도 단어
frequency_word %>%
  filter(sentiment == "neg")

comment_wide <- frequency_word %>%
  filter(sentiment != "neu") %>%  # 중립 제외
  pivot_wider(names_from = sentiment,
```

```
                values_from = n,
                values_fill = list(n = 0))

comment_wide

# 5) 로그 오즈비 구하기
comment_wide <- comment_wide %>%
  mutate(log_odds_ratio = log(((pos + 1) / (sum(pos + 1))) /
                                ((neg + 1) / (sum(neg + 1)))))

comment_wide

top10 <- comment_wide %>%
  group_by(sentiment = ifelse(log_odds_ratio > 0, "pos", "neg")) %>%
  slice_max(abs(log_odds_ratio), n = 10, with_ties = F)

top10

# 로그 오즈비 막대 그래프 만들기
ggplot(top10, aes(x = reorder(word, log_odds_ratio),
                  y = log_odds_ratio,
                  fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(x = NULL) +
  theme(text = element_text(family = "nanumgothic"))


# 댓글 원문
score_comment %>%
  filter(score > 0 & str_detect(reply, "무대")) %>%
  select(reply) %>% pull() %>% head()

score_comment %>%
  filter(score<0 & str_detect(reply, "겁나")) %>%
  select(reply) %>% pull() %>% head()

### 6) 의미망 분석

# 전처리
comment <- raw_comment %>%
  select(reply) %>%
  mutate(reply = str_replace_all(reply, "[^가-힣]", " "),
         reply = str_squish(reply),
         id = row_number())

# 형태소 기준 토큰화
comment_pos <- comment %>%
  unnest_tokens(input = reply,
                output = word,
                token = SimplePos22,
                drop = F)

# 품사별 행 분리
comment_pos <- comment_pos %>%
  separate_rows(word, sep = "[+]")

# 명사 추출
noun <- comment_pos %>%
```

```
  filter(str_detect(word, "/n")) %>%
  mutate(word = str_remove(word, "/.*$"))

# 동사, 형용사 추출
pvpa <- comment_pos %>%
  filter(str_detect(word, "/pv|/pa")) %>%           # "/pv", "/pa" 추출
  mutate(word = str_replace(word, "/.*$", "다"))  # "/"로 시작 문자를 "다"로 바꾸기

# 품사 결합
comment <- bind_rows(noun, pvpa) %>%
  filter(str_count(word) >= 2) %>%
  arrange(id)

# 최종 품사별 토큰화
comment_new <- comment_pos %>%
  separate_rows(word, sep = "[+]") %>%
  filter(str_detect(word, "/n|/pv|/pa")) %>%
  mutate(word = ifelse(str_detect(word, "/pv|/pa"),
                       str_replace(word, "/.*$", "다"),
                       str_remove(word, "/.*$"))) %>%
  filter(str_count(word) >= 2) %>%
  arrange(id)

# 동시 출현 빈도 구하기
pair <- comment %>%
  pairwise_count(item = word,
                 feature = id,
                 sort = T)
pair

pair %>% filter(item1 == "장원영")
pair %>% filter(item1 == "예쁘다")

graph_comment <- pair %>%
  filter(n >= 15) %>%
  as_tbl_graph()

graph_comment # 12개 노드 32개 엣지

# 네트워크 그래프
# 함수화
word_network <- function(x) {
  ggraph(x, layout = "fr") +
    geom_edge_link(color = "gray50",
                   alpha = 0.5) +
    geom_node_point(color = "lightcoral",
                    size = 5) +
    geom_node_text(aes(label = name),
                   repel = T,
                   size = 5,
                   family = "nanumgothic") +
    theme_graph()
}

set.seed(1234)
word_network(graph_comment)

# 유의어 처리하기
comment <- comment %>%
```

```
  mutate(word = ifelse(str_detect(word, "장원영") ,"원영", word))

# 단어 동시 출현 빈도 구하기
pair <- comment %>%
  pairwise_count(item = word,
                 feature = id,
                 sort = T)

# 연결중심성/커뮤니티
set.seed(1234)
graph_comment <- pair %>%
  filter(n >= 18) %>%
  as_tbl_graph(directed = F) %>%          # 방향성 제거
  mutate(centrality = centrality_degree(),          # 연결 중심성
         group = as.factor(group_infomap()))       # 커뮤니티

graph_comment

set.seed(1234)
ggraph(graph_comment, layout = "fr") +       # 레이아웃

  geom_edge_link(color = "gray50",            # 엣지 색깔
                 alpha = 0.5) +               # 엣지 명암

  geom_node_point(aes(size = centrality,     # 노드 크기
                      color = group),        # 노드 색깔
                  show.legend = F) +         # 범례 삭제
  scale_size(range = c(5, 15)) +             # 노드 크기 범위

  geom_node_text(aes(label = name),           # 텍스트 표시
                 repel = T,                   # 노드밖 표시
                 size = 5,                    # 텍스트 크기
                 family = "nanumgothic") +  # 폰트

  theme_graph()                               # 배경 삭제


graph_comment %>%
  filter(name == "원영") # 1번 커뮤니티로 분류

# 1번 커뮤니티 - 원영이 예쁘다
graph_comment %>%
  filter(group == 1) %>%
  arrange(-centrality) %>%
  data.frame()

graph_comment %>%
  arrange(-centrality)

# 2번 커뮤니티 - 진짜 예쁘다
graph_comment %>%
  filter(group == 2) %>%
  arrange(-centrality) %>%
  data.frame()

# 단어쌍이 사용된 댓글 원문 추출
comment %>%
  filter(str_detect(reply, "원영") & str_detect(reply, "예쁘다")) %>%
  select(reply)
```

```
comment %>%
  filter(str_detect(reply, "표정") & str_detect(reply, "좋다")) %>%
  select(reply)

# 7) 파이계수
word_cors <- comment %>%
  add_count(word) %>%
  filter(n >= 20) %>%
  pairwise_cor(item = word,
               feature = id,
               sort = T)

word_cors

# 상대적으로 자주 함께 사용된 단어
word_cors %>%
  filter(item1 == "아이브")

word_cors %>%
  filter(item1 == "쉘드")

# 관심 단어 목록 생성
target <- c("공부", "아이브", "사랑", "중국인", "언니")

top_cors <- word_cors %>%
  filter(item1 %in% target) %>%
  group_by(item1) %>%
  slice_max(correlation, n = 8)

# 그래프 순서 정하기
top_cors$item1 <- factor(top_cors$item1, levels = target)

# 파이계수로 막대그래프
ggplot(top_cors, aes(x = reorder_within(item2, correlation, item1),
                     y = correlation,
                     fill = item1)) +
  geom_col(show.legend = F) +
  facet_wrap(~ item1, scales = "free") +
  coord_flip() +
  scale_x_reordered() +
  labs(x = NULL) +
  theme(text = element_text(family = "nanumgothic"))

set.seed(1234)
graph_cors <- word_cors %>%
  filter(correlation >= 0.15) %>%
  as_tbl_graph(directed = F) %>%
  mutate(centrality = centrality_degree(), #연결중심성
         group = as.factor(group_infomap())) # 커뮤니티

# 네트워크 그래프
set.seed(1234)
ggraph(graph_cors, layout = "fr") +

  geom_edge_link(color = "gray50",
                 aes(edge_alpha = correlation,    # 엣지 명암
                     edge_width = correlation),   # 엣지 두께
                 show.legend = F) +                # 범례 삭제
```

```
  scale_edge_width(range = c(1, 4)) +              # 엣지 두께 범위

  geom_node_point(aes(size = centrality,
                      color = group),
                  show.legend = F) +
  scale_size(range = c(5, 10)) +

  geom_node_text(aes(label = name),
                 repel = T,
                 size = 5,
                 family = "nanumgothic") +

  theme_graph()

## 8) 토픽 모델링
#  전처리
comment_ <- raw_comment %>%
  mutate(reply = str_replace_all(reply, "[^가-힣]", " "),
         reply = str_squish(reply)) %>%
  # 중복 댓글 제거
  distinct(reply, .keep_all = T) %>%
  # 짧은 문서 제거 - 3 단어 이상 추출
  filter(str_count(reply, boundary("word")) >= 3)

# 명사 추출
comment <- comment_ %>%
  unnest_tokens(input = reply,
                output = word,
                token = extractNoun,
                drop = F) %>%
  filter(str_count(word) > 1) %>%

  # 댓글 내 중복 단어 제거
  group_by(id) %>%
  distinct(word, .keep_all = T) %>%
  ungroup() %>%
  select(id, word)

# 빈도 높은 단어 제거
count_word <- comment %>%
  add_count(word) %>%
  filter(n <= 200) %>%
  select(-n)

# 불용어, 유의어 확인하기
count_word %>%
  count(word, sort = T) %>%
  print(n = 200)

# 불용어 목록 만들기
stopword <- c("들이", "하다", "하게", "하면", "해서", "이번", "하네",
              "해요", "이것", "니들", "하기", "하지", "한거", "해주",
              "그것", "어디", "여기", "까지", "이거", "하신", "만큼",
              "정도", "하나","사실","아이", "누구","때문","뭔가",
              "한거","해보", "가지","하지","한게","언제",
              "한데","해주","돌이","해도","하는거")

# 불용어, 유의어 처리
count_word <- count_word %>%
```

```
  filter(!word %in% stopword) %>%
  mutate(word = recode(word,
                       "박주" = "박주비"))

# LDA 모델 만들기
# 문서별 단어 빈도 구하기
count_word_doc <- count_word %>%
  count(id, word, sort = T)

count_word_doc

# DTM 만들기
dtm_comment <- count_word_doc %>%
  cast_dtm(document = id, term = word, value = n)

dtm_comment  # 8285문서, 6347330단어

# 하이퍼 파라미터 튜닝
models <- FindTopicsNumber(dtm = dtm_comment,
                           topics = 2:20,
                           return_models = T,
                           control = list(seed = 1234))

models %>%
  select(topics, Griffiths2004)

FindTopicsNumber_plot(models)

# 토픽 모델 만들기
lda_model <- LDA(dtm_comment,
                 k = 8,
                 method = "Gibbs",
                 control = list(seed = 1234))
lda_model # 8개 토픽

# 모델 내용 확인(단어가 각 토픽에 등장할 확률/문서가 각 토픽에 등장할 확률)
glimpse(lda_model)

term_topic <- tidy(lda_model, matrix = "beta")
term_topic

# 원탑은 토픽1에 등장할 확률이 0.00528로 가장 높다.
term_topic %>%
  filter(term == "원탑" )

# 모든 토픽의 주요 단어
terms(lda_model, 15) %>%
  data.frame()

# 토픽별 beta 상위 10개 단어 추출
top_term_topic <- term_topic %>%
  group_by(topic) %>%
  slice_max(beta, n = 10)

# 토픽별 beta기준 막대 그래프
ggplot(top_term_topic,
       aes(x = reorder_within(term, beta, topic),
           y = beta,
           fill = factor(topic))) +
```

```
  geom_col(show.legend = F) +
  facet_wrap(~ topic, scales = "free", ncol = 4) +
  coord_flip() +
  scale_x_reordered() +
  scale_y_continuous(n.breaks = 4,
                     labels = number_format(accuracy = .01)) +
  labs(x = NULL) +
  theme(text = element_text(family = "nanumgothic"))

# 문서별 토픽 확률 gamma 추출
doc_topic <- tidy(lda_model, matrix = "gamma")
doc_topic

# 문서별로 확률이 가장 높은 토픽 추출
doc_class <- doc_topic %>%
  group_by(document) %>%
  slice_max(gamma, n = 1)

doc_class

# integer로 변환
doc_class$document <- as.integer(doc_class$document)

# 원문에 토픽 번호 부여
comment_topic <- raw_comment %>%
  left_join(doc_class, by = c("id" = "document"))

comment_topic <- comment_topic %>% na.omit()

#토픽별 분류된 문서 개수
comment_topic %>%count(topic)

# 문서를 한 토픽으로만 분류
doc_topic %>%
  group_by(document) %>%
  slice_max(gamma, n = 1) %>%
  count(document) %>%
  filter(n >= 2)

set.seed(1234)
doc_class_unique <- doc_topic %>%
  group_by(document) %>%
  slice_max(gamma, n = 1) %>%
  slice_sample(n = 1)

doc_class_unique

# 토픽별 주요 단어 목록
top_terms <- term_topic %>%
  group_by(topic) %>%
  slice_max(beta, n = 6, with_ties = F) %>%
  summarise(term = paste(term, collapse = ", "))

top_terms

# 토픽별 문서 빈도
count_topic <- news_comment_topic %>%
  count(topic) %>% na.omit()
```

```
count_topic

# 문서 빈도에 주요 단어 결합
count_topic_word <- count_topic %>%
  left_join(top_terms, by = "topic") %>%
  mutate(topic_name = paste("Topic", topic))

count_topic_word

# 막대그래프
ggplot(count_topic_word,
       aes(x = reorder(topic_name, n),
           y = n,
           fill = topic_name)) +
  geom_col(show.legend = F) +
  coord_flip() +

  geom_text(aes(label = n) ,                  # 문서 빈도 표시
            hjust = -0.2) +                   # 막대 밖에 표시

  geom_text(aes(label = term),                # 주요 단어 표시
            hjust = 1.03,                     # 막대 안에 표시
            col = "white",                    # 색깔
            fontface = "bold",                # 두껍게
            family = "nanumgothic") +         # 폰트

  scale_y_continuous(expand = c(0, 0),        # y축-막대 간격 줄이기
                     limits = c(0, 820)) +  # y축 범위
  labs(x = NULL)

# 토픽 이름 짓기
comment_topic <-comment_topic %>%
  mutate(reply = str_squish(replace_html(reply))) %>%
  arrange(-gamma)

#토픽 이름 짓기

# 토픽 1~8내용 살펴보기
comment_topic %>%
  filter(topic == 1 & str_detect(reply, "행복")) %>%
  head(50) %>%
  pull(reply)

comment_topic %>%
  filter(topic == 1 & str_detect(reply, "완벽")) %>%
  head(50) %>%
  pull(reply)

comment_topic %>%
  filter(topic == 1 & str_detect(reply, "워녕이")) %>%
  head(50) %>%
  pull(reply)

# 토픽 이름 목록 만들기
name_topic <- tibble(topic = 1:8,
                     name = c("1. 한국인 코스프레, 언론 통제 비판",
                              "2. 장원영 응원, 인성 논란",
                              "3. 박주비 관련 악플, 건강 상태 악플",
                              "4. 행복 기원, 중국인 악플",
```

```
                                "5. 외모 칭찬, 노력 칭찬",
                                "6. 잔머리 불호, 무대 칭찬",
                                "7. 아이돌 원탑, 표정 칭찬",
                                "8. 완벽한 아이돌, 인성 논란"))

# 토픽 이름 결합하기
top_term_topic_name <- top_term_topic %>%
  left_join(name_topic, name_topic, by = "topic")

top_term_topic_name

# 막대 그래프 만들기
ggplot(top_term_topic_name,
       aes(x = reorder_within(term, beta, name),
           y = beta,
           fill = factor(topic))) +
  geom_col(show.legend = F) +
  facet_wrap(~ name, scales = "free", ncol = 2) +
  coord_flip() +
  scale_x_reordered() +

  labs(title = "장원영 유튜브 영상 댓글 토픽",
       subtitle = "토픽별 주요 단어 Top 10",
       x = NULL, y = NULL) +

  theme_minimal() +
  theme(text = element_text(family = "nanumgothic"),
        title = element_text(size = 12),
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank())
```