KoreanAir Comment Analysis(KoNLP) : Description

Name: JeongSeon Park

setting

```
library(tidyverse)    # 라이브러리 불러오기
library(tidytext)
library(tidyselect)
library(tidygraph)
library(textclean)
library(Sejong)
library(showtext)
library(ggwordcloud)
library(rmarkdown)
library(reprex)
library(readxl)
library(NIADic)
library(magrittr)
library(KoNLP)
library(knitr)
library(ggrepel)
library(ggraph)
library(rJava)
library(tm)
library(tokenizers)
library(utf8)
library(rlang)
library(glue)
library(NLP)
library(widyr)
library(remotes)
library(ldatuning)
library(topicmodels)
library(scales)
```

```
setwd('C:/Users/jspar/OneDrive/Documents/학교/전공/텍마')      # 작업공간 설정
raw_comment <- read_xlsx("대한항공201601-201910.xlsx")       # 자료 불러오기
```

1) 전처리 및 토큰화

```
#1. 첨부된 대한항공 댓글을 읽어와 명사/형용사/동사를 토큰으로 선택하고 명사/형용사/
동사의 기능을 하지 못하는 단어들을 불용어들을 stopword로 정의하여 제외, 유사어 처리
등의 향후 텍스트 분석을 하기 위한 전처리 과정
# 댓글 전처리
krair_comment <- raw_comment %>%
              rename('reply' = '댓글(작성내용)') %>%
              select(reply) %>%    # 변수 이름 수정
              mutate(reply_raw = str_squish(replace_html(reply)),  # 댓글 원본 유지
                     reply = str_replace_all(reply, "[^가-힣]", " "),   # 한글 제외 공백처리
                     reply = str_squish(reply),
                     id = row_number())   # 고유 번호 생성


# 품사 분리하여 행 구성
comment_pos <- krair_comment %>%
            unnest_tokens(input = reply, output = word,
            token = SimplePos22, drop = F) %>%   # simplepos22로 품사 토큰화
            separate_rows(word, sep = "[+]") %>%   # + 기준 행 분리
            filter(str_length(word) >= 2) %>% print()  # 2글자 이상



# 명사 추출
noun <- comment_pos %>%
       filter(str_detect(word, "/n")) %>%
       mutate(word = str_remove(word, "/.*$"))
noun <- noun %>% filter(str_length(word) >= 2)
```

```
# 동사 추출
pv <- comment_pos %>%
filter(str_detect(word, "/pv")) %>% # "/pv" 추출
mutate(word = str_replace(word, "/.*$", "다"))      # 어미에 '다' 붙이기


# 형용사 추출
pa <- comment_pos %>%
filter(str_detect(word, "/pa")) %>% # "/pa" 추출
mutate(word = str_replace(word, "/.*$", "다"))      # 어미에 '다' 붙이기


# 명사/동사/형용사 통합
all_comment <- 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_length(word) >= 2) %>%
            arrange(id) %>% print()
```

1) 중복단어, 유사어, 불용어 처리

```
# 중복단어 및 불용어 처리를 위한 빈도 수 파악
# 명사 빈도 생성
count_noun <- noun %>% count(word, sort = T)      # 빈도 많은 순서대로
# 동사 빈도 생성
count_pv <- pv %>% count(word, sort = T)
# 형용사 빈도 생성
count_pa <- pa %>% count(word, sort = T)
# 명사/동사/형용사 통합 빈도 생성
count_all <- all_comment %>% count(word, sort = T)


# 중복단어 처리
noun <- noun %>% add_count(word, sort=TRUE) %>%
                filter(!word %in% c("대한항공", "비행기", "비행", "항공사",
                                    "항공", "공항", "항공권", "항공편", "항공기"))


# 유사어 통일
pv <- pv %>% mutate(word = ifelse(str_detect(word, "트다"), "타다", word),
                    word = ifelse(str_detect(word, "^들어와"), "들어오다", word),
                    word = ifelse(str_detect(word, "^들으니"), "듣다", word),)
pa <- pa %>% mutate(word = ifelse(str_detect(word, "맛나다"), "맛있다", word),
                    word = ifelse(str_detect(word, "맛있다다"), "맛있다", word), # 토큰화 오류 수정
                    word = ifelse(str_detect(word, "예쁘다"), "아름답다", word),
                    word = ifelse(str_detect(word, "훌륭하다다"), "훌륭하다", word),  # 이하 유사 방
식
                    word = ifelse(str_detect(word, "최송하다다"), "최송하다", word),
                    word = ifelse(str_detect(word, "필요하다다"), "필요하다", word),
                    word = ifelse(str_detect(word, "가능하다다"), "가능하다", word),
                    word = ifelse(str_detect(word, "당연하다다"), "당연하다", word))
```

```
# 유사어 통일
noun <- noun %>% mutate(word = ifelse(str_detect(word, "^승무원"), "승무원", word),
                        word = ifelse(str_detect(word, "^직원"), "직원", word),
                        word = ifelse(str_detect(word, "좋았습니"), "좋음", word),
                        word = ifelse(str_detect(word, "좋았"), "좋음", word),
                        word = ifelse(str_detect(word, "좋습니"), "좋음", word),
                        word = ifelse(str_detect(word, "좋은거"), "좋음", word),
                        word = ifelse(str_detect(word, "좋았던거"), "좋음", word),
                        word = ifelse(str_detect(word, "좋으네"), "좋음", word),
                        word = ifelse(str_detect(word, "좋았슴"), "좋음", word),
                        word = ifelse(str_detect(word, "좋았답니"), "좋음", word),
                        word = ifelse(str_detect(word, "짱좋음"), "좋음", word),
                        word = ifelse(str_detect(word, "좋았"), "좋음", word),
                        word = ifelse(str_detect(word, "좋은서비스로"), "좋은서비스", word),
                        word = ifelse(str_detect(word, "^편안"), "편안", word),  #편안으로 시작하는 단어
                        word = ifelse(str_detect(word, "친절"), "친절", word),   #-> 편안
                        word = ifelse(str_detect(word, "^사용"), "사용", word),   # 이하 유사 방식으로 처리
                        word = ifelse(str_detect(word, "^제공"), "제공", word),
                        word = ifelse(str_detect(word, "^훌륭"), "훌륭", word),
                        word = ifelse(str_detect(word, "^사람"), "사람", word),
                        word = ifelse(str_detect(word, "^선택"), "선택", word),
                        word = ifelse(str_detect(word, "전체적"), "전반적", word))
```

1) 중복단어, 유사어, 불용어 처리

```
# 명사/동사/형용사 통합 처리
all_comment <- all_comment %>%
        add_count(word, sort=TRUE) %>%
        filter(!word %in% c("대한항공", "비행기", "비행", "항공사",
                            "항공", "공항", "항공권", "항공편", "항공기")) %>%
        mutate(word = ifelse(str_detect(word, "^승무원"), "승무원", word),
               word = ifelse(str_detect(word, "^직원"), "직원", word),
               word = ifelse(str_detect(word, "좋았습니"), "좋음", word),
               word = ifelse(str_detect(word, "좋았"), "좋음", word),
               word = ifelse(str_detect(word, "좋습니"), "좋음", word),
               word = ifelse(str_detect(word, "좋은거"), "좋음", word),
               word = ifelse(str_detect(word, "좋았던거"), "좋음", word),
               word = ifelse(str_detect(word, "좋으네"), "좋음", word),
               word = ifelse(str_detect(word, "좋았습"), "좋음", word),
               word = ifelse(str_detect(word, "좋았답니"), "좋음", word),
               word = ifelse(str_detect(word, "짱좋음"), "좋음", word),
               word = ifelse(str_detect(word, "좋았"), "좋음", word),
               word = ifelse(str_detect(word, "좋은서비스로"), "좋은서비스", word),
               word = ifelse(str_detect(word, "^편안"), "편안", word),  #"편안"으로 시작하는 단어
               word = ifelse(str_detect(word, "^친절"), "친절", word),   #-> 편안
               word = ifelse(str_detect(word, "^사용"), "사용", word),   # 이하 유사 방식으로 처리
               word = ifelse(str_detect(word, "^제공"), "제공", word),
               word = ifelse(str_detect(word, "^훌륭"), "훌륭", word),
               word = ifelse(str_detect(word, "^사람"), "사람", word),
               word = ifelse(str_detect(word, "^선택"), "선택", word),
               word = ifelse(str_detect(word, "전체적"), "전반적", word),
               word = ifelse(str_detect(word, "트다"), "타다", word),          #-> 내용 분석 결과
               word = ifelse(str_detect(word, "^들어와"), "들어오다", word),  # '타다'가 잘못
               word = ifelse(str_detect(word, "^들오니"), "들다", word),       토큰화 됨
               word = ifelse(str_detect(word, "맛나다"), "맛있다", word),
               word = ifelse(str_detect(word, "맛있다다"), "맛있다", word), # 토큰화 오류 수정
               word = ifelse(str_detect(word, "예쁘다"), "아름답다", word),
               word = ifelse(str_detect(word, "훌륭하다다"), "훌륭하다", word),  # 이하 유사 방식
               word = ifelse(str_detect(word, "최송하다다"), "최송하다", word),
               word = ifelse(str_detect(word, "필요하다다"), "필요하다", word),
               word = ifelse(str_detect(word, "가능하다다"), "가능하다", word),
               word = ifelse(str_detect(word, "당연하다다"), "당연하다", word))
```

```
# 빈도 갱신
# 명사 빈도 생성
count_noun <- noun %>% count(word, sort = T)
# 동사 빈도 생성
count_pv <- pv %>% count(word, sort = T)
# 형용사 빈도 생성
count_pa <- pa %>% count(word, sort = T)
# 명사/동사/형용사 통합 빈도 생성
count_all <- all_comment %>% count(word, sort = T)

# 불용어 처리
stopword <- c( "들이", "하다", "하게", "하면", "해서", "이번", "하네",
               "해요", "이것", "니들", "하기", "하지", "한거", "해주",
               "그것", "어다", "여기", "까지", "이거", "하신", "만큼",
               "하려", "해라", "하나", "나들", "에서", "그렇다", "어떻다", "들다",
               "일다", "그러다", "우리", "있습니", "정도", "경우", "되었습니",
               "가지", "되다", "번째", "동안", "어떠하다", "이러하다", "그러하다",
               "서울", "인천", "한국") # 단어 빈도 수 파악해 가만서 의미없는 단어 처리

# 불용어 제거(단어 수 카운팅 한 변수에)
count_noun <- count_noun %>%
              filter(!word %in% stopword)
count_pv <- count_pv %>%
            filter(!word %in% stopword)
count_pa <- count_pa %>%
            filter(!word %in% stopword)
count_all <- count_all %>%
             filter(!word %in% stopword)

# 불용어 제거(단어 자체 변수에)
noun <- noun %>% filter(!word %in% stopword)
pv <- pv %>% filter(!word %in% stopword)
pa <- pa %>% filter(!word %in% stopword)

all_comment <- all_comment %>% filter(!word %in% stopword)
```

2) 명사/동사/형용사 워드클라우드 - 코드

```
#2. 전체 댓글의 명사/형용사/동사들에 대한 워드클라우드 & 설명
# 명사 워드클라우드
top100noun <- count_noun %>% head(100) # 가시성을 위해 top100 추출


font_add_google(name = "Nanum Gothic", family = "nanumgothic")
showtext_auto()
top100noun %>% ggplot(aes(label = word, size = n,
                          color = factor(sample.int(n=10,
                                                    size=nrow(top100noun),
                                                    replace = TRUE)))) +
  geom_text_wordcloud(seed = 1234) +
  scale_radius(limits = c(3, NA),
               range = c(3, 15)) +
  theme_minimal()


# 동사 워드클라우드
top100pv <- count_pv %>% head(100)


font_add_google(name = "Nanum Gothic", family = "nanumgothic")
showtext_auto()
top100pv %>% ggplot(aes(label = word, size = n,
                        color = factor(sample.int(n=10,
                                                  size=nrow(top100pv),
                                                  replace = TRUE)))) +
  geom_text_wordcloud(seed = 1234) +
  scale_radius(limits = c(3, NA),
               range = c(3, 15)) +
  theme_minimal()
```

```
# 형용사 워드클라우드
top100pa <- count_pa %>% head(100) # 가시성을 위해 top100 추출

font_add_google(name = "Nanum Gothic", family = "nanumgothic")
showtext_auto()
top100pa %>% ggplot(aes(label = word, size = n,
                        color = factor(sample.int(n=10,
                                                  size=nrow(top100pa),
                                                  replace = TRUE)))) +
  geom_text_wordcloud(seed = 1234) +
  scale_radius(limits = c(3, NA),
               range = c(3, 15)) +
  theme_minimal()

# 명사/동사/형용사 통합 워드클라우드
top100all <- count_all %>% head(100)

font_add_google(name = "Nanum Gothic", family = "nanumgothic")
showtext_auto()
top100all %>% ggplot(aes(label = word, size = n,
                         color = factor(sample.int(n=10,
                                                   size=nrow(top100all),
                                                   replace = TRUE)))) +
  geom_text_wordcloud(seed = 1234) +
  scale_radius(limits = c(3, NA),
               range = c(3, 15)) +
  theme_minimal()
```

2) 명사/동사/형용사 워드클라우드 - 결과

명사 워드클라우드: 최다빈출100개 명사를 빈도순으로 워드클라우딩 한 결과 서비스, 친절, 승무원,

형용사 워드클라우드: 최다빈출100개 형용사를 빈도순으로 워드클라우딩 한 결과 좋다, 없다, 많다, 같다, 편하다, 맛있다, 아니다 등의 단어가 많이 등장함을 알 수 있다.

동사 워드클라우드: 최다빈출100개 동사를 빈도순으로 워드클라우딩 한 결과 타다, 가다, 느끼다,

명사/동사/형용사 통합 워드클라우드: 최다빈출100개 명/동/형을 빈도순으로 워드클라우딩 한 결과 좋다, 없다, 등의 단어가 많이 등장함을 알 수 있다.

3) 명사에 대한 동시출현 네트워크 - 코드

```
#3. 전체 댓글의 명사들에 대한 동시출현 한 단어들의 빈도수를 가지고 연결중심성을 크기로 표시하고 커뮤니티를 나타내는 단어 네트워크 그림


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


# 네트워크 그래프 데이터에 연결 중심성, 커뮤니티 변수 추가
set.seed(1234)
graph_comment <- pair %>% filter(n >= 30) %>%
                 as_tbl_graph(directed = F) %>%
                 mutate(centrality = centrality_degree(),
                 group = as.factor(group_infomap()))


# 네트워크 그래프에 연결 중심성과 커뮤니티 표현
set.seed(1234)
ggraph(graph_comment, layout = "fr") +
geom_edge_link(color = "gray50",
               alpha = 0.5) +
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 = 4,
                family = "nanumgothic") +
theme_graph()
```

3) 명사에 대한 동시출현 네트워크 - 결과

#3. 전체 댓글의 명사들에 대한 동시출현 한 단어들의 빈도수를 가지고 연결중심성을 크기로 표시하고 커뮤니티를 나타내는 단어 네트워크 그림

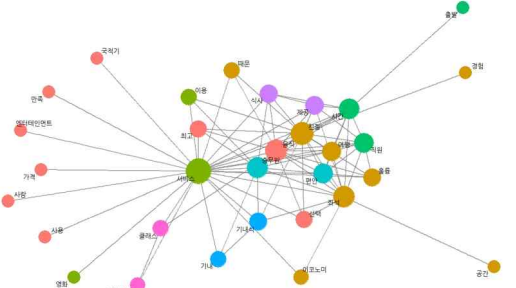
* 서비스, 승무원, 음식, 친절, 좌석, 편안 등이 중심성이 높게 나타남.
* 승무원-친절 / 친절-서비스 / 음식-서비스 / 승무원-서비스/ 직원-친절 등이 동시출현 빈도가 높음

§10.Rmd × 20170279_기말고사.Rmd ×

Filter

	item1	item2	n
1	승무원	친절	133
2	친절	승무원	133
3	친절	서비스	131
4	서비스	친절	131
5	음식	서비스	122
6	서비스	음식	122
7	승무원	서비스	121
8	서비스	승무원	121
9	직원	친절	103
10	친절	직원	103
11	편안	서비스	96
12	서비스	편안	96
13	좌석	서비스	95
14	서비스	좌석	95
15	시간	서비스	93
16	서비스	시간	93
17	여행	서비스	91

Showing 1 to 17 of 198,454 entries, 3 total colu



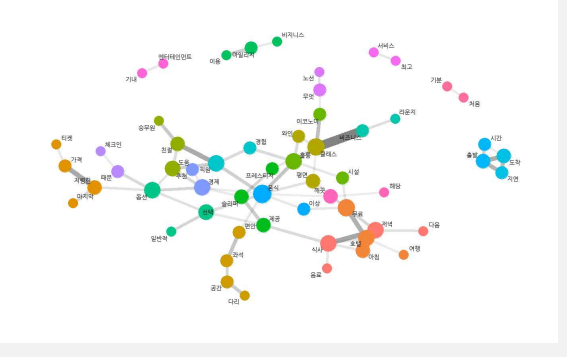
4) 파이계수 네트워크 그림 - 코드

```
#4. 전체 댓글의 명사들의 상관성을 나타내는 파이계수들을 가지고 연결중심성을 크기로 표시하고 커뮤니티를 나타내는 단어 네트워크 그림을 그리기


# 두 명사 쌍 상관계수 구하기
word_cors <- noun %>%
            add_count(word) %>%
            filter(n >= 20) %>%
            pairwise_cor(item = word, feature = id, sort = T)


# 관심 단어 목록 생성
target <- c("서비스", "가격", "좌석", "음식", "승무원", "직원")    # 중심성이 큰 단어로 추출
top_cors <- word_cors %>%
            filter(item1 %in% target) %>%
            group_by(item1) %>%
            slice_max(correlation, n = 10)


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


# 파이계수 네트워크 그래프 그리기
font_add_google(name = "Nanum Gothic", family = "nanumgothic")
showtext_auto()
```

```
set.seed(1234)
graph_cors %>%
ggraph(layout = "fr") +
geom_edge_link(color = "gray50", aes(edge_alpha = correlation,
                                     edge_width = correlation),
               show.legend = F) +
scale_edge_width(range = c(1, 4)) +
geom_node_point(aes(size = centrality, color = group),
                show.legend = F) +
scale_size(range = c(4, 8)) +
geom_node_text(aes(label = name), repel = T, size = 4, family = "nanumgothic") +
theme_graph()
```

4) 명사에 대한 동시출현 네트워크 - 결과

#4. 전체 댓글의 명사들의 상관성을 나타내는 파이계수들을 가지고 연결중심성을 크기로 표시하고 커뮤니티를 나타내는 단어 네트워크 그림 그리기

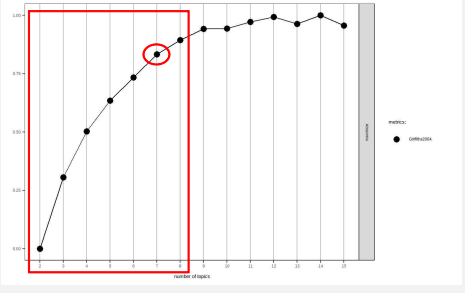


* 중심성과 네트워크성이 큰 단어 쌍을 위주로 분석한 결과, 직원이 친절함 / 비즈니스 클래스 이용 / 출발 및 도착시간 지연 등 고객의 평가와 관련된 내용을 유추해볼 수 있다.

5) 적절한 토픽 수 찾기

#5번~12번: 명사/형용사/동사를 이용한 토픽모델 활용.
#5. 토픽을 찾기 위한 적합한 토픽의 수를 찾고 설명하기. 단 최적의 토픽 수는 절대 8개를 넘지 말아야 함.

```
all_comment_id <- all_comment %>%
                  add_count(word) %>%
                  select(id, word, n)

# 문서별 단어 빈도
count_word <- all_comment_id %>% count(id, word, sort = T)

# DTM 생성
dtm_krair <- count_word %>%
             cast_dtm(document = id, term = word, value = n)

# DTM 내용 확인하기
as.matrix(dtm_krair[1:15, 1:15])

# 토픽 수 비교하여 성능 비교
models_krair <- FindTopicsNumber(dtm = dtm_krair,
                  topics = 2:15,
                  return_models = T,
                  control = list(seed = 1234))
models_krair %>% select(topics, Griffiths2004)

# 성능 지표 그래프
FindTopicsNumber_plot(models_krair) # 최적 토픽 수: 7개

# 토픽 별 주요 단어 확인
aaa <- terms(lda_model, 20) %>% data.frame()
```



* 7번째부터 완만해지므로 [2,8]의 범위에서 가장 적절한 토픽의 수는 7개

6) 토픽 별 워드클라우드(토픽 별 최다빈도단어) - 코드

```
#6. 각 토픽 내에서의 어떠한 단어들이 중심인지를 알아보기 위해 토픽 별 워드클라우드를
    한 페이지에 그리고 설명하기


# 토픽 수 7개인 lda 모델 생성
lda_model <- LDA(dtm_krair, k = 7, method = "Gibbs", control = list(seed = 1234))


# gamma 추출
doc_topic <- tidy(lda_model, matrix = "gamma") %>%
                mutate(topic_name = paste( "Topic", topic))


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


# 데이터셋을 결합하기 위해 기준 변수 타입을 integer로 변환
doc_class$document <- as.integer(doc_class$document)

# 단어 별 토픽 이름 부여
all_comment_topic <- all_comment %>%
                        left_join(doc_class, by = c("id" = "document")) %>%
# 전처리 작업을 거치지 않은 raw data에 결합했으므로 topic에 NA 존재
                        na.omit()

# 토픽 별 단어 빈도
count_alltp <- all_comment_topic %>%
                  group_by(topic_name) %>%
                 count(word, sort = T)
```

```
# 토픽 별 워드클라우드 (토픽별 최다 빈도)
count_alltp %>% group_by(topic_name) %>%
                    ggplot(aes(label = word, size = n,
                    color = factor(sample.int(n=10,
                                               size=nrow(count_alltp),
                                               replace = TRUE)))) +
facet_wrap(~ topic_name, scales = "free", ncol = 3) +
geom_text_wordcloud(seed = 1234) +
scale_radius(limits = c(10, NA), range = c(3, 15)) +
theme_minimal()
```

6) 토픽 별 워드클라우드(토픽 별 최다빈도단어) - 결과

<토픽 별 최다 빈출 단어>

Topic1: 시간, 좋다, 여행, 서비스, …
-> 시간적인 부분에서 (긍정적) 평가

Topic2: 친절, 서비스, 승무원, 좋다, …
-> 승무원 서비스 부분에서 (긍정적) 평가

Topic3: 좋다, 서비스, 음식, 제공, …
-> 음식 서비스 부분에서 (긍정적) 평가

Topic4: 서비스, 좋다, 타다, 기내식, …
-> 기내식 부분에서 (긍정적) 평가

Topic5: 좌석, 좋다, 없다, 서비스, …
-> 좌석 부분에서 (긍정적) 평가

Topic6: 서비스, 좋다, 승무원, 친절, …
-> 전반적인 서비스 긍정적 평가

Topic7: 좋다, 직원, 직원, 친절, 클래스, …
-> ㅇㅇ 클래스 서비스 (긍정적) 평가

7) 토픽 별 tf-idf 중요단어 & 막대그래프 - 코드

```
#7. 각 토픽 내에서 어떠한 단어가 중요한지를 알기 위해 토픽 별 TF-IDF를 구하고 각 토픽
별 TF-IDF에 의한 중요단어 10개에 대한 막대그래프를 한 페이지에 그리고 설명하기


# 토픽 별 단어 빈도수를 기반으로 tf-idf 구하기
tfidf_topic <- count_alltp %>%
            bind_tf_idf(term = word,
                        document = topic_name,
                        n = n) %>%
          arrange(-tf_idf)


# 주요 단어 10개 추출
top10 <- tfidf_topic %>%
        group_by(topic_name) %>%
        slice_max(tf_idf, n = 10, with_ties = F)



# 막대그래프 생성
# 한글폰트
font_add_google(name = "Nanum Gothic", family = "nanumgothic")
showtext_auto()
```

```
top10$topic_name %>% ggplot(aes(x = reorder_within(word, tf_idf, topic_name),
                                y = tf_idf, fill = topic_name)) +
geom_col(show.legend = F) +
coord_flip() +
facet_wrap(~ topic_name, scales = "free", ncol = 7) +
scale_x_reordered() +
scale_y_continuous(n.breaks = 5,
labels = number_format(accuracy = .001)) +
labs(title = "대한항공 댓글 주요 단어",
     subtitle = "토픽 별 TF-IDF Top 10",
     x = NULL) +
theme_minimal() +
theme(text = element_text(family = "nanumgothic"),
      plot.title = element_text(size = 14, face = "bold"),
      plot.subtitle = element_text(size = 12),
      strip.text = element_text(size = 11)) # 카테고리명 폰트
group_by(topic_name) %>%
            ggplot(aes(label = word, size = n,
                       color = factor(sample.int(n=10,
                                                 size=nrow(count_alltp),
                                                 replace = TRUE)))) +
facet_wrap(~ topic_name, scales = "free", ncol = 3) +
geom_text_wordcloud(seed = 1234) +
scale_radius(limits = c(10, NA), range = c(3, 15)) +
theme_minimal()
```

8) 토픽 별 beta값 큰 순서대로 워드클라우드 - 코드

```
#8. 각각의 토픽들에 대해서 어떠한 단어가 토픽의 내용을 구분한지 알기 위해 각 토픽 별
beta의 값의 큰 순서대로 단어를 나타내는 워드클라우드를 한 페이지에 그리고 설명하기


# 토픽 수 7개인 모델 생성
lda_model <- LDA(dtm_krair, k = 7, method = "Gibbs", control = list(seed = 1234))


# 주요 단어 확인
# 단어들이 토픽 별로 들어갈 확률 beta 추출
term_topic <- tidy(lda_model, matrix = "beta") %>%
        mutate(topic_name = paste("Topic", topic))


# 토픽별 beta 값 큰 순서대로 단어 추출
term_topic %>% group_by(topic) %>% slice_max(beta)



# 토픽 별 워드클라우드
term_topic %>% group_by(topic_name) %>%
        ggplot(aes(label = term, size = beta*10000,
                color = factor(sample.int(n=10,
                                size=nrow(term_topic),
                                replace = TRUE)))) +
facet_wrap(~ topic_name, scales = "free", ncol = 3) +
geom_text_wordcloud(seed = 1234) +
scale_radius(limits = c(10, NA), range = c(3, 15)) +
theme_minimal()
```

8) 토픽 별 beta값 큰 순서대로 워드클라우드 - 결과

#8. 각각의 토픽들에 대해서 어떠한 단어가 토픽의 내용을 구분한지 알기 위해 각 토픽 별 beta의 값의 큰 순서대로 단어를 나타내는 워드클라우드를 한 페이지에 그리고 설명하기

<각 토픽에 등장할 수 있는 주요 단어>

Topic1: 여행, 시간, 가다, 출발, 편안 …
-> 시간적인 부분에서 평가

Topic2: 친절, 많다, 승무원, 맛있다, …
-> 승무원 서비스 부분에서 평가

Topic3: 음식, 제공, 영화, 선택, …
-> 음식 및 영화 선택 폭 평가

Topic4: 타다, 이용, 기내식, 같다, 비빔밥, ...
-> 기내식 부분에서 평가

Topic5: 좌석, 없다, 크다, 이코노미 …
-> 좌석 부분에서 평가

Topic6: 서비스, 좋다, 일등석, 프레스티지, …
-> 상위 클래스 좌석 평가

Topic7: 좋다, 직원, 훌륭, 비즈니스, 클래스, ...
-> 특정 클래스 서비스 평가

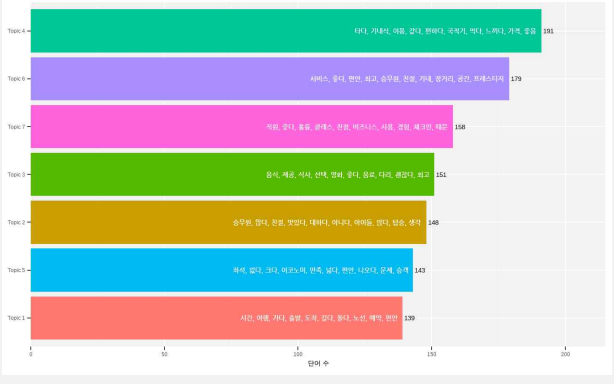
9) 토픽 별 beta값이 큰 중요단어 10개 막대그래프 - 코드

```
#9. 각각의 토픽들에 대해서 어떠한 단어가 토픽의 내용을 구분한지 알기 위해 각 토픽
    별 beta 값이 큰 중요단어 10개에 대한 막대그래프를 한 페이지에 그리고 설명하기

# 단어가 들어 있는 term_topic을 이용하여 토픽 별 beta 값이 높은 단어 10개를 추려냄
top_terms <- term_topic %>%
                group_by(topic_name) %>%
                slice_max(order_by=beta, n = 10, with_ties = F) %>%
                summarise(term = paste(term, collapse = ", "))

# 원문과 토픽 번호가 들어 있는 krair_topic을 이용하여 토픽 별 문서를 구함
count_topic <- krair_topic %>%
                  count(topic_name) %>% na.omit()

# count_topic에 top_terms를 결합한 다음 막대 그래프의 x축에 Topic 1의 형태로
  토픽 번호를 표시하기 위해 topic_name으로 결합
count_topic_word <- count_topic %>%
                        left_join(top_terms, by = "topic_name")
```

```
# 토픽 별 문서 수와 주요 단어로 막대그래프
# geom_text()를 이용해 막대 끝에 문서 빈도를 표시하고, 막대 안에 토픽의 주요 단어를 표
시함
font_add_google(name = "Nanum Gothic", family = "nanumgothic")
showtext_auto()

count_topic_word %>% ggplot(aes(x = reorder(x=topic_name, X=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", # 색
          fontface = "bold", # 글씨 두께
          family = "nanumgothic") + # 폰트
          scale_y_continuous(expand = c(0, 0), # y축-막대 간격 줄이기
          limits = c(0, 215)) + # y축 범위
labs(x = NULL) +
ylab( "단어 수")
```

9) 토픽 별 beta값이 큰 중요단어 10개 막대그래프 - 결과

#9. 각각의 토픽들에 대해서 어떠한 단어가 토픽의 내용을 구분한지 알기 위해 각 토픽 별 beta 값이 큰 중요단어 10개에 대한 막대그래프를 한 페이지에 그리고 설명하기



1. 여행을 위해 대한항공을 예약한 고객의 여정(노선, 긴 시간)
2. 친절하게 아이들을 대하는 승무원
3. 음식 및 영화 서비스 측면에서 평가
4. 편안함, 기내식, 가격 측면에서의 평가
5. 이코노미 석을 이용한 고객의 좌석 공간 만족
6. 장거리 운항 속 프레스티지 석의 기내 공간
7. 퍼스트&비즈니스 석을 이용한 고객의 체크인 등 서비스 이용

10) 감마값 큰 문장5개 추출

```
#10. 각각의 토픽 별 감마값이 큰 문장들을 5개씩만 뽑아 실제 문장과 감마값만을 출력하고
주로 어떠한 내용인지를 설명하기

# 토픽별 주요 문서 추출
reply_topic <- krair_topic %>%
        group_by(topic) %>%
        slice_max(gamma, n = 100)


# 토픽 1 내용 살펴보기
reply_topic %>% filter(topic == 1) %>% pull(gamma, reply_raw) %>% head(5)
# 토픽 2 내용 살펴보기
reply_topic %>% filter(topic == 2) %>% pull(reply_raw, gamma) %>% head(5)
# 토픽 3 내용 살펴보기
reply_topic %>% filter(topic == 3) %>% pull(reply_raw, gamma) %>% head(5)
# 토픽 4 내용 살펴보기
reply_topic %>% filter(topic == 4) %>% pull(reply_raw, gamma) %>% head(5)
# 토픽 5 내용 살펴보기
reply_topic %>% filter(topic == 5) %>% pull(reply_raw, gamma) %>% head(5)
# 토픽 6 내용 살펴보기
reply_topic %>% filter(topic == 6) %>% pull(reply_raw, gamma) %>% head(5)
# 토픽 7 내용 살펴보기
reply_topic %>% filter(topic == 7) %>% pull(reply_raw, gamma) %>% head(5)
```

10) 감마값 큰 문장5개 추출 - 결과

* 결과

<토픽1>
인천->콜롬보(경유)->말레 행 항공편을 탑승했습니다. 인천 출발시 매번 지연 출발(최대 1시간 가량)하기 때문에 출발 시간을 잘 확인하는 것이 좋습니다. 항상 정시보다 늦게 출발할거라면 아예 스케줄 자체를 그렇게 공지하는게 낫지 않을까 하네요.. 돌아올 때 역시 말레에서 1시간 가량 지연되어 출발하더군요 0.3351648

대한항공은 러시아에 4개(모스크바, 상페테부르크, 이루쿠츠크, 블라디보스토크) 노선이 있으며 상페테부르크와 이루쿠츠크 노선은 날씨/탑승률 등의 관계로 하계 노선만 유지하며 동계에도 가는 노선은 모스크바와 블라디보스토크 2개 노선만 있습니다. 블라디보스톡은 기본적으로 소형기인 737-800/900 등을 활용하며 하계 기간에는 A330-200/300 등 중형기를 활용하고 있습니다. 한국에서 블라디보스톡까지는 2시간 30분 정도 소요되며 중국을 거쳐서 가므로 출발이 지연되는 경우도 가끔 있습니다. 0.3310924

김해공항 전산이 멈춰 수속이 늦어져서 비행기도 자연스럽게 연착이 되었음 이후 출발을 하기 위해 움직이던 도중 어떠한 문제로 인해 계류장으로 돌아옴 하지만 제대로 된 안내도 하지 않다가 결국 항의를 받고나서야 대응을 시작함 시간만 계속 낭비하다가 정비 못하고 다른 대체비행기 편으로 출국함. 너무 늦어져 탑승객 식사라도 할 수 있게 해줘야 하지 않냐고 했으나 대응하지 않다가 결국 항의가 커지자 해줌 0.2822300

작년 이맘 때... 태국으로 여행을 갔었습니다. 그 전까지는 태국국적 항공(타이항공) 이나 케세이를 주로 이용하였는데... 모처럼 동행하는 동생의 말-국적기를 이용하자... 대한항공... 태국의 친구들에게 도착시간을 미리 알려 놓고 마중 나와주기를 청했는데.... 9시간 늦게 도착을 했네요. 태국으로 가야할 비행기가... 중국에서 아직 안 돌아 와서... 출발이 지연된다는 이야기는 출발이 세번이나 지연된 후에 알았습니다. 그리고 보상으로 저녁식사 티켓 한장 받았는데.... 그 때 이후로 0.2807309

밀라노에서 로마 로 가는 항공 후 시드니, 시드니. 비행 시간 이 작업의 경우, 한국에 있는 공항 호텔 입니다 (매우 좋은 호텔!) 야간. 우리는 이 수에 인첩 합니다. 우리는 이른 아침 시드니에, 늦은 오후에 서울에 도착 (시간이 한 시간 차이, 수 갔다가 한 호텔에 머물러 본. 특정한 뷔페를 아침과 저녁 식사도 하였고. 유럽 여행은 좋은 휴식을 하는 훌륭한 자로 약 2시까지 로마에 도착한 다음 날 남겨두지 않았다. 우리는 밀라노에서 돌아오는 길에 들려서 두 시간이면 서울 이라고 확인하지 0.2762951

-> 노선 수, 지연 및 연착 대응에 아쉬움을 호소하는 대한항공 이용 고객

10) 감마값 큰 문장5개 추출 - 결과

* 결과

<토픽2>
0.33868378812199
"지인분들이 대한항공도 많이 괜찮아졌다고 해서 한번 다시 타 보았으나 아쉽게도 어떤 부분이 많이 좋은지 잘 모르는 비행이였습니다. 그냥 평타 치는 항공사가 아닌가 쉽습니다. 본인은 항상 싱가포르항공 1순위로 고려합니다. 싱가포르항공과 비교 해봤을때 엔터테인먼트 시스템도 많이 질이 떨어지고 전반적으로 봤을 때 싱가포르항공에 비하면 많이 좀 부족 하지 않나 쉽습니다. 비행시 옆에 계신 승객분이 Sky Priority 가지고 계신 분이 앉아 계셨는데 그 분 덕시 싱가포르항공이 더 좋은거 같다고"

0.29047619047619
"2 4 2 배열의 비행기였는데(기종 A330) 4배열에 아기가 이륙 전부터 착륙 할때까지 울어 굉장히 힘든 여행이 되었어요ㅠㅠ승무원님을 불러 다른 자리로 옮겨도 될까요 하고 물어봤더니 된다고는 하시고 권유해 주셨지만 옮기려는 자리가 좀은 관계로 저는 그냥 비상구에 앉아 왔죠. 그대로 비상구에 앉겠다고 하니 승무원님께서 이어플러그를 준비해주셨어요! 세심한 배려라고 느꼈어요 ㅎㅎ 이륙부터 착륙까지 틈틈히 아기가 자지러지게 울때 와서 아기를 달래셨습니다. 빠른 대처 감동이네요~인천-상하이 ,"

0.277504105090312
"인천에서 LA이까지 가는 항공편이었습니다. 일단 10시간이 넘는 비행시간이기 때문에 자리가 제일 중요했는데, 체크인할 당시에 티켓팅을 해주시는 분이 중간에 끼인 자리랑 옮겨주시려고 다른 곳 자리까지 직접 연락하며 알아봐 주셨습니다. 비록 옮기지만 상당히 신경을 많이 써주시는 모습에서 다른 외국 항공사에서 느끼지 못했던 친절함과 배려를 느낄 수 있었습니다. 대한항공이 인천 제2 터미널로 옮겼기 때문에 공항도 좀더 여유 있고 쾌적하게 이용할 수 있는 부분도 좋습니다. 기내에서는 역시 국적기에 맞게"

0.261595547309833
"직원들 친절하고 정시 출발하고 좋습니다 하지만 타이트한 승무원들의 복장 똑같은 피부톤, 성형수술한 승무원들을 보면 인조인간을 보는 느낌이다. 대한항공 오너들의 갑질로 이미지가 안좋은 항공사이지만 직원들도 그리 서비스 마인드가 좋은 항공사는 아닌 것 같다 오래된 항공기도 많은 것 같다"

0.258241758241758
"중남미여행을 위해 24시간이 넘는 비행시간을 이코노미클래스로 여행하기는 너무 힘들거 같아 조금은 호사스런 여행을 함 돈은 많이 지불했지만 훌륭한 서비스와 편안한 좌석/승무원들의 친절함/맛있는 기내식/선택의 여지가 많은 엔터테인먼트 등 지출이 아깝지 않았음"

-> 친절함과 배려를 느낄 수 있는 서비스 경험 / 상위 클래스 좌석에 대한 긍정적 평가 / 외국 항공사와의 비교(대한항공 부정적 평가)

10) 감마값 큰 문장5개 추출 - 결과

* 결과

<토픽3>
0.351118760757315
"제가 아는 많은 사람들이 음식을 할 수도 있지만 일반적으로 항공사 여기기 꽤 괜찮은 시설 이 작업이 제한됩니다. 하지만, 이 비행기가 약간 이상한 음식. 우리는 8 시: 아침 식사 시간에 제공되는 25도 저녁 식사를 할 수 없이 정상적인 아침 식사. 등의 와인! 끝으로 비행기 저녁 식사 시간에, 다른 식사는 제공 되지만 와인이 있었습니다. 이 두 번째 식사 해 있지 않은 모든 선택을 제공하지 않는 나. 물은 모든 샌드위치 또는 간식을 대신 수 있는. 아니, 아무것도 있었습니다. 결국 난 빵과"

0.334776334776335
"모든 면에서 훌륭한 항공사 이 기 때문이다. 자리에서 보딩 시작, 찾을 수 있는 베개, 담요, 물 한 병. 그 후 그들은 손으로 써 앉아 헤드폰 및 하우스 슬리퍼가 마련되어 있습니다. 어떤 가방을 함께 제공하는 작은 집 화장실 슬리퍼나 칫솔 치약 튜브 케이스를 신선하게 하다. 제공된 현재의 많은 영화, 음악, 게임. 음식은 또 다른 멋진 경험! 마지막으로 이 항공사 제공을 포함하여 금속 기구. 한끼 식사 가 제공됩니다. 일반적인 음료 및 와인 제공된다. 퍼스트 클래스 좌석 만 제공되는 와인은"

0.303571428571429
"4/5성급 호텔을 경유하는 항공 이 서울에서 무료로 시드니에서 런던으로. 중요한 것은 서둘러 않은 경우 분할 이 티켓은 시드니에서 런던으로 긴 여행에 완벽한 곳. 항공사에서 직접 이동하여 주는 기회에 호텔에서 인천, 이른 오후에 도착, 호텔 바로 앞으로 잘 정리되어 있다 저녁 식사와 아침식사가 제공되는데 시설의 호텔. 여유 있게 환승 공항 다음 점심 시간 이후 런던으로 가는 비행기. 나는 아무런 제트 - 런던에 도착했을 때 의 이점 및 지연 되었다 그리고 충분한 음식과 음료가 호텔에서 잠을 잘"

0.299392097264438
"저는 최근에 auckland 항공에서 돌아옵니다. 저는 아주 만족스럽고 직원들의 서비스의 질을 가는 비행기. 적의 경우 큰 문제가 없었다는 버튼, 누군가가 와서 바로. 제가 여행을 반환. 저는 숙소 에서 무료로 제공하는 5성급 호텔, 그랜드 하얏트 호텔 서울. 매우 쉽게 전송하는 공항에서 반갑게 맞아 주었고 호텔의 매니저. 저녁 식사를 하면서, 침대와 아침식사가 제공되는 뷔페 스타일의 레스토랑에서 음식을 제공하는 다양한 음식을 요리사가 신선하게 준비되어 있어서. 이 항공사는 가격이 매우 경쟁력"

0.289285714285714
"항공사는 많은 다리 방은 매우 편안했고, 다른 평면 두 개 이상 있습니다. 한 것은 잘 하지 않는 한 잔 하는 음료와 술을 반만 제공. 하지만 그들이 식사 하는 2 가지의 쌀 요리 모두 인식도 일반적으로 매운 닭, 소고기, 또는 생선. 하지만 미국에서 아시아로 가격을 적은 가격으로 마셔서 다리 방, 작은 음료, 음식과 이상 한 가치가 있었어요. 또한 좋은 영화 가 함께 새로운 릴리즈 및 오래된 영화."

-> 주로 식사에 대한 평가. 긍정적 / 부정적 평가로 나뉨

10) 감마값 큰 문장5개 추출 - 결과

* 결과

<토픽4>
0.276295133437991
"비지니스 트립에서는 편안한 항공기가 가장 중요한 부분이다. 왜 국적기인지를 말해주는 대한한공의 서비스는 저가 항공 그 어디에서도 따라할 수 없는 차별성이 있다. 비지니스 클래스 고객을 대하는 태도는 역시 최고구나 할 수 있다. 사무장의 별도 인사와 필요한 부분에 대한 응대 , 그리고 신문을 보고 있으면 더 필요한게 있는지 물어봐주는 센스 그 어느것 하나 부족함이 없었다. 9월9일 3시 20분 부산에서 제주로 오는 비행기에 탑승했을때 시작부터 사무장 덕분에 기분이 좋았는데 내릴때까지 기분이"

0.255533199195171
"착석간의 앞뒤 간격이 타 항공사에 비해 비교적 여유롭고 국적기라는 점에서 심리적인 부분 뿐만 아니라 기내식 등 제반 지원이 편안하였습니다. 다만, 어린이들에게 주는 탑승기념품이 오랜 기간 동안 동일하여, 그러한 부분에 있어서는 다소 개선이 필요할 것 같습니다."

0.2543171114459969
"비상구자리에 앉은 할아버지가, 식사도중에 의자를 젖혀서 머리에 맞음. 참다가, 나중에 너무 젖혀버리길래 앞을 봤더니 다리까지 뻗고 있었음. 할아버지에게 조금만 접어달라고 양해를 구했더니, \"너 편하라고 내가 불편해야겠냐\"라는 말을 들음. 이기적인 노인네때문에 기분이 상했고 말로 통하지 않을 것 같아서 승무원에게 이야기했으나, 본인은 말을 따로 할 수 없다고 말함. 아쉬운 대응력으로 기분이 엉망났으나, 알고 보니 승무원들과 친한 할아버지 같아서 더 기분이 좋지 않았음."

0.251984126984127
"8월 가족여행 시 대한항공 비즈니스석을 이용했습니다.. 기내 좌석도 넓고 여유 있으며 완전히 누울수 있는 구조여서 아주 편안히 다녀왔습니다.. 기내식은 괜찮은 편인데, 미국에서 귀국편은 좀 더 다양한 음식이 서비스 되면 좋을것 같네요. 승무원들의 서비스는 무척 좋았습니다."

0.24853228962818
"9월에 벤쿠버 왕복으로 해서 다녀왔는데 기내 엔터테이먼트가 잘 되어 있었고 아시아나에 비해서 기내 어메니티가 좋아서 편했습니다. 치약이나 칫솔도 아시아나는 화장실에 구비해 줬는데 대한은 개인마다 하나씩 다 주어서 승무원에게 부탁하지도 않아서 편하고 좋았습니다."

-> 주로 승무원이 고객을 대하는 태도에 대한 인적서비스 차원에서의 긍정적, 부정적 평가

10) 감마값 큰 문장5개 추출 - 결과

* 결과

<토픽5>
0.287284144427002
"부모님 모시고 가는 여행이라 이런 방법 저런 방법 써서 비즈니스로 모시고 감. 비즈니스다운 탑승 수속이나 좌석에 만족. 식사도 멋지게 나오던 중 스테이크를 미디움으로 주문. 그렇나 그냥 미지근한 정도에 안쪽도 미디움이라고 하기에는 좀 많이 날 것... 부모님이 드시기 좀 어려워 하시길래 재조리를 요청했지만 결국 큰 변화 없이 다시 나옴. 원인은 기내 조리방법의 한계라고 하는 얘기를 들음. 돌아오는 길에는 웰던으로 주문. 으음... 기내식 스테이크는 웰던으로!!!"

0.260032102728732
"제 남편과 저는 다리를 1st 시카고는 한국 볏가을을. 우리는 인천으로 계속 쓰고. 1st 항공 시간을 내라는. 서비스와 미학 이 항공사 서비스에 감동 받았습니다. 여기 원하는 좌석, 긴급 상황에 있는 앉아있기 좋은 곳이나 탈출문... 다리가 걸려 내 마음의 공간 이다. 직원들은 모두 아주 깔끔한 팀 / 보았다, 매우 공손하고 신중한. 나는 목소리를 제공하지 않아 승객 중 거의 들리지 않는다. 그들은 이 일을 사랑합니다! 저희는 에어컨이 있는 작은 문제가 아니지만 신경 쓰는 것은 결국 남편의"

0.248677248677249
"회사에서 권장하는 비행기는 에미레이츠나 두바이-인천 구간 에미레이츠 a380 은 이야기와는 다르게 앞뒤 공간 넓은거 말고는 장점이 별로 없음 이번에 처음으로 대한항공을 이용해봤는데, 해외 구간 탑승 승무원들이 딱 봐도 경력과 노하우가 상당함이 엿보였음 . 대한항공 특유의 라면 서비스와 한국으로 갈때 나오는 비빔밥, 죽 음식들도 대체적으로 만족합니다."

0.24860853432282
"작년에 대한항공으로 뉴욕 갈 때 이코노미석의 좌석 간격이 좁아 고생했는데,이번 오클랜드 왕복 여행 시 좌석 간격이 넓어 아주 좋았습니다. 기내식도 괜찮았고,볼 영화도 많았습니다. 적당하다고 생각되는 비행기 가격이었고,오클랜드로 갈 때는 탑승객들도 얼마 없어 아주 편안했습니다."

0.24860853432282
"짧은 비행구간이라 작은 비행기일 줄 알았는데, 큰 비행기에 새로운 기종이라서 특히 더 커진 모니터에 만족했습니다. 좌석도 편안하고, 무엇보다 대한항공은 아시아나보다 시간대가 좋습니다. 단, 유아동반을 주로 하는데, 타항공사는 제가 먼저 말하기 전에 패스트트랙 안내나 유아동반 우선 탑승 서비스를 말해주는데, 여기는 먼저 말을 안하니 제가 물어보고 그 서비스를 이용했습니다."

-> 음식에 대한 아쉬움 및 긍정적 평가, 깔끔함, 좌석 공간 만족, 영화 및 모니터 만족 등등 다양한 서비스를 아우르고 있음. 특정 서비스를 주제삼을 만한 것이 해당 문장에선 안 보임

10) 감마값 큰 문장5개 추출 - 결과

* 결과

<토픽6>
0.29047619047619
"말레 노선은 프레스티지 슬리퍼 2-2-2 배열 A330-200 라서 밤 비행임에도 불구하고 누워서 편하게 갈 수 있었어요 ㅎ 승무원들 팀 마다 분위기가 달라서인지,, 왕편보단 복편에 서비스가 훨씬 좋았던것 같아요 사실 갈때 서비스나 승무원분들 친절도가 별로였어서.. 좀 실망했었는데 돌아올땐 팀 분위기도 좋아보이고 무엇보다 이쁘시기도하고 ㅋㅋ; 서비스도 좋았거든요~ 프레스티지석이라두 가격차이가 많이 심하지 않고 괜찮아서 전 다음에 몰디브 갈 때도 마마 프레스티지를 끊을 것같내요. 클롬보 경유"

0.2682539968253968
"밸리노선은 일등석을 운영하지 않는데 프레스티지 승객중 업그레이드를 시켜줍니다. 서비스는 프레스티지 서비스지만 좌석은 일등석에 앉을수 있는것입니다. 기종은 구기종이라 기존 프레스티지 슬리퍼에 발을 놓는 공간이 있는것이 추가되었다고 보시면 됩니다. 항상 대한항공을 이용하는데 이날따라서 다른날에 비해 서비스가 좋다고는 못느꼈던것 같습니다. 항상 서비스가 좋은편이지만 엘번중 한번정도는 이러네요. 하지만 만족스러운 식사와 깔끔한 항공기내부 상태를 갖추고 있어 항상 대한항공만을 이용합니다."

0.2517857142857143
"6시간 40분 가량 비행하는 내내 불편함은 전혀 없었다. 이코노미석과 크게 다른 점은 좀 더 넓은 좌석과 한 층 업그레이드된 서비스.. 조용하고 편안한 분위기가 너무 좋았고, 가는 내내 서비스를 챙겨줘서 오히려 부담스러울 정도 였다. 하지만, 가끔은 이런 서비스도 즐기면서 가는 것도 좋을 듯하다. 승무원들이 너무 친절.. 외국사람이 나에게 대한항공이 최고라고 한 게 기억이 난다."

0.245421245421245
"최신 영화들이 준비되어 있어서 편안하게 엔터테인먼트를 즐기며 맛있는 식사 (한식과 양식)이 준비되어져 있으며 간식으로 아이스크림이 굉장이 좋은 것 같다. 화장실에는 칫약과 칫솔등이 구비되어져 있으며 항상 청결하게 준비되어져 있어서 기분이 좋다. 승무원들은 굉장히 친절하며 도움을 요청하면 바로바로 답을 주어서 굉장히 편하게 비행을 즐길 수 있었던 것 같다."

0.237560192616372
"김해공항에서 오사카 간사이 공항으로 운항하는 대한항공을 왕복으로 이용했습니다. 모닝캄 회원이 되고나서 가족과 처음으로 같이 나갔는데 김해공항 혼잡도가 심해는데 모닝캄 전용카운터로 기다리지 않고 쉽게 티켓팅했습니다. 김해공항에 있는 조그마한 대한항공 라운지도 이용했구요. 항공기가 다소 작고 노후된 기종이긴 했지만 짧은 거리니 괜찮았습니다. 어린이에겐 탑승시에 작은 선물을 주더군요 매우 감사했습니다. 또 좋았던 것은 간사이 공항에서 입국수속 후 짐을 찾으로 갔더니 대한항공 직원분이 모닝캄 회원이라고"

->프레스티지 석을 이용한 고객들의 만족스러운 서비스 평가

10) 감마값 큰 문장5개 추출 - 결과

* 결과

<토픽7>
0.312698412698413
"항공권 예약 하는 마지막 순간 이지만 일반적으로 에어 뉴질랜드 항공 여행에 있어서 저렴한 일반적인 경우. 여행사, 비행 센터, 정말 좋은 우리의 날짜 만 특별한 것을 발견할 수 있는 에어 뉴질랜드 항공 이코노미 가 조금 이상의 비즈니스 클래스. 안 해 봅니다. 아주 인상적이었습니다. 여행자의 경우 는 일반 비즈니스 클래스 항공사 변경다른 이야기를 하지 않은 좋은, 하지만 무엇을 지불 한 후 그는 이코노미 여행자 이 한 서비스도 좋거나, 았구요. 유일한 단서를 주었고 우리 agent 는 아시아"

0.3100303951367778
"저는 방콕에 뉴욕, 서울 을 통해 항공, 비즈니스 클래스. 이는 내 첫 번째 비즈니스 클래스 여행, 회사가 비교할 수 있습니다. 아, 회사는 테스트 실패함, 전용 서비스, 섬세한 서비스, 시설, 음식, 등이 있습니다. 직원들은 좋았어요, 하지만. 저는 그와 같은 큰 회사 슬프 카 상관 관계를 실시간으로 매핑하여 무비 카탈로그 있을 수 없습니다, 영화 및 추가할 수 없고, 비행기를 타고 같은 일 의 첫 귀국하는 비행기는 4월 5월 25일! 음식은 아닙니다 이벤트, 풍부한 에 없는,"

0.2825040012841092
"저는 최근에 비즈니스 클래스 가 (대한항공 프레스티지 클래스 또는) 와 런던에서, 서울에서 두 평면을 변경. 오클랜드에서 서울 (및 역방향 사용하는 꽤 오래된 낡고 있는 777, 그럼에도 불구하고 가격 지불하긴 했지만 괜찮았습니다. 직원들은 매우 친절했어요. 좌석은 괜찮았어요, 특별한 것은. 서울에서 런던으로 가는 제가 380 해 (및 역방향) 을 사용. 더 편안하게 앉을 수 있는, 조용한 엔진, 전반적으로 새로운 제품 및 후면에 있는 바에 빠지지 었습니다. 뉴질랜드 여행 에서 런던으로"

0.277504105090312
"저는 한 5 항공 에 시작하는 동안 음식도 놀라울 정도로 편안한 비행기 (사실 꽤 좋은 빌어 먹을 수 있 - 특히 였는데 가격도 아주 좋았습니다!), 완벽한 서비스이기 때문에. 모든 직원들은 사랑스럽고 환상적인 직원들이 있는 곳이 인천... 그 다음에 그들을 잃어버린 카메라 관련 stayover 제 부반송 여행 및 저장된... 저는 지금 제가 정말 우수한 고객 서비스 때문에 다시 가고 싶을 것입니다 서양식 (경험) 를 실행할 수 있습니다. 좋은 가격과 훌륭한 고객 서비스 항공사 로 적극"

0.276785714285714
"저는 토론토에서 오 년 동안 비행 했습니다 (인수가 나중에 델타) 를 사용하여 노스웨스트, 에어 캐나다, 캐세이 퍼시픽. 것입니다. 직원들의 멋진 환자들이 항공은 항공사 중 하나. 직원들은 항상 미소와 함께 서비스를 제공하고 투덜거리 지원함은 이온 화 하지 않은 직원들은 다른 항공사 도 표시되지 않습니다. 확실히 음식은 음식, 빵을 살 수 있지만 다른 항공사 마법사와는 달리 2 비교. 발견한 항공 서비스 품질의. 저는 주로 충성 고객 이 항공사."

-> 다양한 항공서비스에 대한 후기들. 해당 댓글만으로는 특색을 잡기는 어려워보임. / 영어 번역의 한계

11) 토픽 별 감정분석 - 긍정/부정

#11. 각 토픽들의 감정 단어들을 사용한 감정분석을 통해 각 토픽들을 긍정과 부정으로 구분하기

```
# 감정 사전 불러오기
senti_dic <- read_delim("SentiWord_Dict.txt", delim='\t',
                        col_names=c("word","polarity"))


# 감정 점수 부여
all_comment_topic <- all_comment_topic %>%
                     left_join(senti_dic, by = "word") %>% # 감성 사전 결합
                     mutate(polarity = ifelse(is.na(polarity), 0, polarity)) # 감성 사전에 없으면 중립


# 감정이 분명한 단어를 살펴보기 위해 2 이상이면 'positive', -2 이하이면 'negative' 그 외는 'neutural'로 분류
all_comment_topic <- all_comment_topic %>%
                     mutate(sentiment = ifelse(polarity >= 2, "positive",
                                               ifelse(polarity <= -2, "negative", "neutural")))
# 토픽 별 점수 합산
score_comment <- all_comment_topic %>%
                 group_by(topic_name) %>%
                 summarise(score = sum(polarity)) %>% ungroup() %>%
                 mutate(sentiment = ifelse(score > 0, "positive",
                                           ifelse(score < 0, "negative", "neutural")))


# 댓글의 감정 빈도와 비율 생성
frequency_score <- all_comment_topic %>%
                   group_by(topic_name) %>%
                   count(sentiment) %>%
                   mutate(ratio = n/sum(n)*100) %>% print()
```

	topic_name	score	sentiment
1	Topic 1	400	positive
2	Topic 2	576	positive
3	Topic 3	779	positive
4	Topic 4	563	positive
5	Topic 5	517	positive
6	Topic 6	967	positive
7	Topic 7	783	positive

	topic_name	sentiment	n	ratio
1	Topic 1	negative	68	2.139037
2	Topic 1	neutural	2840	89.336269
3	Topic 1	positive	271	8.524693
4	Topic 2	negative	78	2.359347
5	Topic 2	neutural	2855	86.358137
6	Topic 2	positive	373	11.282517
7	Topic 3	negative	52	1.485714
8	Topic 3	neutural	3038	86.800000
9	Topic 3	positive	410	11.714286
10	Topic 4	negative	94	2.439024
11	Topic 4	neutural	3385	87.830825
12	Topic 4	positive	375	9.730150
13	Topic 5	negative	45	1.485149
14	Topic 5	neutural	2679	88.415842
15	Topic 5	positive	306	10.099010
16	Topic 6	negative	54	1.618220
17	Topic 6	neutural	2775	83.158526
18	Topic 6	positive	508	15.223254
19	Topic 7	negative	49	1.340629
20	Topic 7	neutural	3213	87.906977
21	Topic 7	positive	393	10.752394

Showing 1 to 21 of 21 entries, 4 total columns

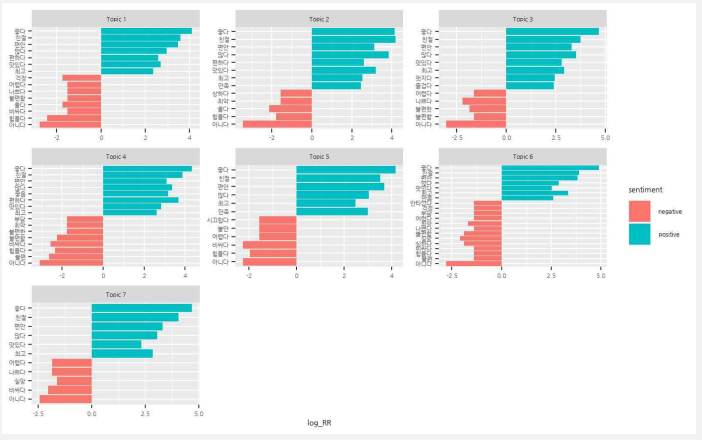
12) 토픽 별 로그RR 그래프

#12. 각 토픽들 내에서 사용된 긍정과 부정 단어들을 가지고 각 토픽 별 단어들에 대한 로그RR을 적당한 페이지를 할당하여 그래프를 그리고 설명하기

```
new_frequency_word <- all_comment_topic %>%
                  group_by(topic_name) %>%
                  count(sentiment, word, sort = T)


new_comment_wide <- new_frequency_word %>% # Wide form으로 변환
                  filter(sentiment != "neu") %>%
                  pivot_wider(names_from = sentiment,
                              values_from = n,
                              values_fill = list(n = 0))


# 로그RR 구하기
new_comment_wide <- new_comment_wide %>%
                  mutate(log_RR = log(((positive + 1) / (sum(positive + 1))) /
                                      ((negative + 1) / (sum(negative + 1)))))


new_top10 <- new_comment_wide %>%
          group_by(sentiment = ifelse(log_RR > 0, "positive", "negative")) %>%
          slice_max(abs(log_RR), n = 50, with_ties = F)
```

```
new_top10 %>% group_by(topic_name) %>%
              ggplot(aes(x = reorder(word, log_RR), y = log_RR,
                         fill = sentiment)) +
geom_col() +
facet_wrap(~ topic_name, scales = "free", ncol = 3) +
scale_radius(limits = c(10, NA), range = c(3, 15)) +
coord_flip() +
labs(x = NULL) +
theme(text = element_text(family = "nanumgothic"))
theme_minimal()
```

12) 토픽 별 로그RR 그래프 - 결과



<댓글에서 긍정, 부정 중
상대적으로 자주 사용된 단어 비교>

- 두드러지는 단어
Topic1: 걱정, 울다
Topic2: 상하다, 최악, 울다
Topic3: 즐겁다, 멋지다
Topic4: 부담, 최악
Topic5: 시끄럽다, 불만, 만족
Topic6: 안타깝다, 걱정, 최악, 실망, 심하다
Topic7: 실망

(긍정 단어는 비교적 유사함)

12) 토픽 별 로그RR 그래프

#12. 위의 모든 것을 고려하여 각 토픽에 대한 이름을 짓고 전체적으로 대한항공 이용 댓글에 대한 텍스트마이닝을 통해 제시하고자 하는 시사점을 설명하기

토픽 별 베타값이 높은 10개 단어가 포함된 top_terms 변수와 베타값을 중심으로 한 워드클라우드(term_topic), 감성분석 결과를 중점으로 함

top_terms

	topic_name	term
1	Topic 1	시간, 여행, 가다, 출발, 도착, 걸다, 돌다, 노선, 예약, 편안
2	Topic 2	승무원, 댈다, 친절, 맛있다, 대하다, 아니다, 아이들, 앉다, 합...
3	Topic 3	음식, 제공, 식사, 선택, 영화, 좋다, 음료, 다리, 편찮다, 최고
4	Topic 4	타다, 기내식, 이용, 같다, 편하다, 국적기, 먹다, 느끼다, 가격, ...
5	Topic 5	좌석, 없다, 크다, 이코노미, 만족, 넓다, 편안, 나오다, 문제, 승...
6	Topic 6	서비스, 좋다, 편안, 최고, 승무원, 친절, 기내, 장거리, 공간, 프...
7	Topic 7	직원, 좋다, 좋똉, 클래스, 친절, 비즈니스, 사용, 경험, 체크인, ...

```
# 토픽 이름 목록 작성
name_topic <- tibble(topic = 1:7,
                     name = c("1.노선 변경 및 딜레이(출발, 도착 시간)에 따른 아쉬움",
                              "2. 승무원 서비스(직원 서비스) 부분에서 평가 - (아이들을)친절히 대함",
                              "3. 음식 및 영화 선택 폭 평가 - 선택 폭이 별로 없다 & 괜찮다",
                              "4.  기내식 부분에서 평가 - 비빔밥 두드러짐",
                              "5. 이코노미 이용자의 좌석 부분에서 평가 - 편하고 넓음",
                              "6. 프리스티지 장거리 이용자의 좌석 평가 - 편하고 공간적 여유",
                              "7. 특정 클래스(퍼스트, 비즈니스) 서비스 평가 - 훌륭"))
```

-> 감성 분석 결과 긍정적인 감성 지수가 높게 나왔으므로 주제를 베타 값이 높은 단어 위주로 긍정적인 방향으로 작성함. Topic1은 자체 감성이 긍정적이지만 내용을 살펴봤을 때 시간적인 부분에 대한 아쉬움을 토로하는 내용이 비교적 존재하고 긍정 점수 또한 가장 낮기 때문에 다소 부정적인 토픽으로 차별화를 둠.

* 시사점: 대한항공 서비스 이용객들의 전반적인 서비스 만족도를 파악할 수 있음. 직원 서비스, 음식 서비스, 엔터테인먼트 서비스, 좌석 서비스 등 다양한 영역에서의 만족도를 알아볼 수 있음. 특히 좌석 등급 별 고객 만족도를 파악 할 수 있어 향후 부족한 측면을 보완하는 데 도움이 될 수 있음. 기내식 부분에서도 자주 언급된 음식명(ex. 비빔밥, 닭고기 등)과 음식에 대한 평가를 나타내는 형용사를 매칭하여, 향후 보완점을 파악하여 더 나은 식사 제공을 할 수 있을 것으로 보임. (대체로 높은 수준의 만족도를 보이지만, 시간 지연, 연착, 노선 부족 및 변경에 대한 대응이 아쉬운 점으로 꼽힘)