Data Mining Practice5

데이터 다운 및 데이터프레임화

```
library(dplyr)
library(tidyr)
library(stringr)
library(tidytext)
library(ggplot2)
install.packages("tidytext")
# URL에서 텍스트 파일을 읽기


#정규식을 사용하는 것 까지는 맞는데, 이렇게 한장한장을 읽고 바꾸는 것은 너무 비효율적
# 폴더 경로 지정
folder_path <- "C:/Users/silkj/Desktop/한동대학교/5학기/데이터 마이닝 실습/Data-Mining-Practicum/myR/NIV_English_Bible/"

# 폴더 내의 모든 텍스트 파일을 리스트로 가져오기
file_list <- list.files(folder_path, pattern = "*.txt", full.names = TRUE)

# 빈 리스트 생성
bible_texts <- list()

# 파일 리스트를 반복하면서 파일 읽기
for (file in file_list) {
  # 파일 내용 읽기
  bible_content <- readLines(file, encoding = "UTF-8")

  # 파일 이름(책 이름) 추출
  book_name <- tools::file_path_sans_ext(basename(file))  # 확장자 제거

  # 리스트에 저장
  bible_texts[[book_name]] <- bible_content
```

```
}

# 결과 확인 (책 이름과 첫 몇 줄 출력)
for (book in names(bible_texts)) {
  cat("Book:", book, "\n")
  cat("Content:", head(bible_texts[[book]]), "\n\n")
}

# 최종 데이터를 저장할 데이터프레임
bible_df <- tibble(Book = character(), Chapter = numeric(),
Verse = numeric(), Script = character())

# 각 파일을 처리하여 데이터프레임에 추가
for (book_name in names(bible_texts)) {
  script_lines <- bible_texts[[book_name]][-1]  # 첫 줄(책 이
름)을 제외한 나머지

  # 각 줄을 분석해서 "Chapter", "Verse", "Script"로 분리
  chapter <- c()
  verse <- c()
  script <- c()

  for (line in script_lines) {
    # 정규식으로 Chapter:Verse와 Script 분리
    match <- regexpr("^(\\d+):(\\d+)\\s(.+)$", line, perl=T
RUE)

    if (match[1] != -1) {
      # Chapter와 Verse 추출
      chapter_verse <- regmatches(line, regexpr("^(\\d+):
(\\d+)", line))
      chapter_verse_split <- strsplit(chapter_verse, ":")
[[1]]
      chapter <- c(chapter, as.numeric(chapter_verse_split
[1]))
      verse <- c(verse, as.numeric(chapter_verse_split[2]))

      # Script 부분 추출
```

```
      script <- c(script, regmatches(line, regexpr("\\s(.+)
$", line)))
    }
  }

  # 각 파일의 결과를 데이터프레임으로 추가
  temp_df <- tibble(
    Book = rep(book_name, length(chapter)),
    Chapter = chapter,
    Verse = verse,
    Script = script
  )

  # 최종 데이터프레임에 추가
  bible_df <- bind_rows(bible_df, temp_df)
}

# 데이터프레임 확인
print(bible_df)

# Book 변수에서 숫자와 하이픈 제거
bible_df$Book <- gsub("^\\d+-", "", bible_df$Book)

# 결과 확인
print(bible_df)



# 구약성서
ot_books <- c("Genesis", "Exodus", "Leviticus", "Numbers",
"Deuteronomy", "Joshua", "Judges",
              "Ruth", "Samuel-1", "Samuel-2", "Kings-1", "K
ings-2", "Chronicles-1", "Chronicles-2",
              "Ezra", "Nehemiah", "Esther", "Job", "Psalm
s", "Proverbs", "Ecclesiastes",
              "Song of Solomon", "Isaiah", "Jeremiah", "Lam
entations", "Ezekiel", "Daniel",
              "Hosea", "Joel", "Amos", "Obadiah", "Jonah",
```

```
"Micah", "Nahum", "Habakkuk",
              "Zephaniah", "Haggai", "Zechariah", "Malach
i")

#신약성서
nt_books <- c("Matthew", "Mark", "Luke", "John", "Acts", "R
omans", "Corinthians-1", "Corinthians-2",
              "Galatians", "Ephesians", "Philippians", "Col
ossians", "Thessalonians-1",
              "Thessalonians-2", "Timothy-1", "Timothy-2",
"Titus", "Philemon", "Hebrews",
              "James", "Peter-1", "Peter-2", "John-1", "Joh
n-2", "John-3", "Jude", "Revelation")
```

Task 1-1 Perform tokenization on the Bible text dataset and provide descriptive statistics.

토큰화 이후 전체 단어의 수를 세기

```
bible_df%>%
  unnest_tokens(word, Script)%>%
  group_by(Book)%>%
  nrow()
[1] 724429
```

토큰화 이후 중복을 제거한 고유 단어 수 세기

```
bible_df %>%
  unnest_tokens(word, Script)%>%
  distinct(word) %>%
  nrow()

[1] 14302
```

토큰화 이후 1번만 등장한 단어 수 세기

```
bible_df %>%
  unnest_tokens(word, Script) %>%
  count(word) %>%
  filter(n == 1) %>%
  nrow()

[1] 4335
```

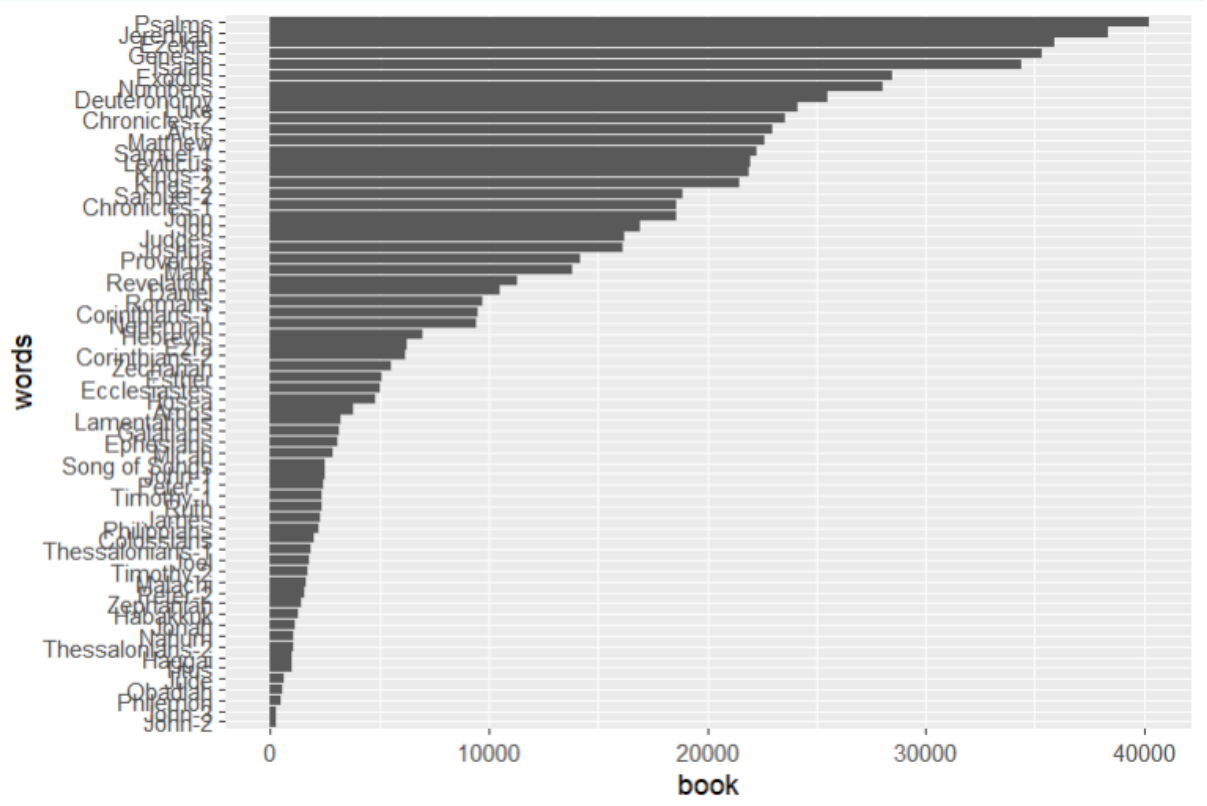
상위 10개 단어의 빈도 수 세기

```
bible_df %>%
  unnest_tokens(word, Script) %>%
  count(word, sort = TRUE) %>%
  head(10)

# A tibble: 10 × 2
   word      n
   <chr> <int>
 1 the   55481
 2 and   29541
 3 of    24958
 4 to    20867
 5 you   13700
 6 in    11333
 7 will  10156
 8 he     9630
 9 a      9015
10 i      8719
```

book별 전체 단어 수 세기

```
bible_df %>%
  unnest_tokens(word, Script) %>%
  group_by(Book) %>%
  summarise(total_words = n())%>%
  ggplot(aes(x = reorder(Book,total_words), y = total_word
s))+
  geom_col()+
```

```
  coord_flip()+
  labs(x = "words", y="book")
```



Task 1-2 For improved Result of text mining, what can be applied during the tokenization and preprocessing phase Explain further process you might attempt

NER 전처리

```
install.packages("udpipe")
library(udpipe)
```

r프로그래밍에서 udpipe 패키지 다운 후 로드

```
ud_model <- udpipe_download_model(language = "english")
#udpipe 모델 로드
ud_model <- udpipe_load_model(file = ud_model$file_model)
```

```
# 성경 텍스트를 대상으로 NER 수행
bidle_token<-bible_df%>%
  unnest_tokens(word, Script)
annotations <- udpipe_annotate(ud_model, x = bidle_token$wo
rd)
# 데이터 프레임으로 변환
annotations_df <- as.data.frame(annotations)
```

udpipe 의 영어 모델 다운, 이후 ud_model에 대해 다운로드한 모델 파일을 지정하여 로드

이후 성경 텍스트를 토큰화하여 script에 대하여 각각의 word 단위로 자름

annotations안에 토큰화한 word에 대하여 품사 태깅 및 NER분석을 시작함 결과는 주석이 달린 텍스트로 변환되기 때문에, as.data.frame을 통해 데이터프레임으로 변경함

```
person_names <- annotations_df %>%
  filter(upos == "PROPN") %>%  # 고유명사(PROPN) 필터링
  select(token) %>%
  distinct()%>%
  filter(token == "paul")

print(person_names)
```

이후 upos에 대한 필터링을 통해 필요한 정보를 추가로 찾을 수 있음

현재 PROPN으로 고유명사에 대한 필터링이 제대로 작동하지 않는 이유로는, unnest_tokens가 모든 단어를 소문자로 변경시키기 때문에 발생하는것으로 유추해볼 수 있다. 때문에 모든 단어를 그대로 유지한 채 동일한 작업을 진행하기로 했다.

```
bidle_token<-bible_df%>%
  unnest_tokens(word, Script, to_lower = FALSE)
annotations <- udpipe_annotate(ud_model, x = bidle_token$wo
rd)
# 데이터 프레임으로 변환
annotations_df <- as.data.frame(annotations)
# 사람 이름 추출 (명사구 중에서 고유명사만 추출)
person_names <- annotations_df %>%
  filter(upos == "PROPN") %>%
```

```
  count(token, sort = TRUE)%>%
  head(10)
> print(person_names)
        token    n
1         God 4072
2      Israel 1878
3       Jesus 1259
4       David  997
5       Judah  836
6        Lord  804
7   Jerusalem  802
8       Egypt  613
9  Israelites  611
10     Christ  528
11       Saul  411
12      Jacob  383
13      Aaron  352
14   Almighty  341
15       King  323
16        Son  297
17    Solomon  294
18  Sovereign  294
19    Levites  282
20    Babylon  281
```

고유명사로 가장 많이 사용된 God, Israel뿐 아니라 Judah, David등도 확인된다.

Task 1-3 Using the prep rocessed the dataset Perform EDA process to know more insight about Bible. For example, which book is the longest and the shortest? Or How many words are in Bible? How many names are in the Bible? What are the mo st fre quent verbs used in the Bible?

가장 많이 사용된 동사 찾기

```
verb_usage <- annotations_df %>%
  filter(upos == "VERB") %>%
```

```
  count(token, sort = TRUE) %>%
  head(10)
   token    n
1   have 4309
2   said 3178
3     do 2710
4    has 2400
5   come 1461
6     go 1402
7   went 1228
8   came 1191
9   made 1115
10   let 1083
```

가장 많이 등장한 고유명사 찾기

```
person_names <- annotations_df %>%
  filter(upos == "PROPN") %>%
  count(token, sort = TRUE)%>%
  head(20)

        token    n
1         God 4072
2      Israel 1878
3       Jesus 1259
4       David  997
5       Judah  836
6        Lord  804
7   Jerusalem  802
8       Egypt  613
9  Israelites  611
10     Christ  528
11       Saul  411
12      Jacob  383
13      Aaron  352
14   Almighty  341
15       King  323
```

```
16        Son  297
17     Solomon  294
18  Sovereign  294
19     Levites  282
20     Babylon  281
```

가장 많이 등장한 고유명사에 대해서, god와 이스라엘, 지져스가 가장 많이 등장하였고, 그 외에 사람에 대해서는 David, 유다가 가장 많이 등장하였다.

자주 등장하는 명사 찾기

```
noun_usage <- annotations_df %>%
  filter(upos == "NOUN") %>%
  count(token, sort = TRUE)%>%
  head(10)
print(noun_usage)
```

```
    token    n
1    lord 7434
2    king 2513
3     son 2355
4     man 2224
5  people 2219
6  israel 1834
7     men 1823
8    land 1456
9     day 1426
10 father 1253
```

Task 2-1 Calculate TF/IDF for each group of Bible. Show top 20 words of high TF/IDF for each group of Bible.

```
ot_books <- c("Genesis", "Exodus", "Leviticus", "Numbers",
"Deuteronomy", "Joshua", "Judges",
              "Ruth", "Samuel-1", "Samuel-2", "Kings-1", "K
```

```
ings-2", "Chronicles-1", "Chronicles-2",
              "Ezra", "Nehemiah", "Esther", "Job", "Psalm
s", "Proverbs", "Ecclesiastes",
              "Song of Solomon", "Isaiah", "Jeremiah", "Lam
entations", "Ezekiel", "Daniel",
              "Hosea", "Joel", "Amos", "Obadiah", "Jonah",
"Micah", "Nahum", "Habakkuk",
              "Zephaniah", "Haggai", "Zechariah", "Malach
i")
ot_Law <- c("Genesis", "Exodus", "Leviticus", "Numbers", "D
euteronomy")
ot_History <- c("Joshua", "Judges",
                "Ruth", "Samuel-1", "Samuel-2", "Kings-1",
"Kings-2", "Chronicles-1", "Chronicles-2",
                "Ezra", "Nehemiah", "Esther")
ot_Poetry <- c("Job", "Psalms", "Proverbs", "Ecclesiastes",
               "Song of Solomon")
ot_Prophecy <- c( "Isaiah", "Jeremiah", "Lamentations", "Ez
ekiel", "Daniel",
                  "Hosea", "Joel", "Amos", "Obadiah", "Jona
h", "Micah", "Nahum", "Habakkuk",
                  "Zephaniah", "Haggai", "Zechariah", "Mala
chi")
nt_books <- c("Matthew", "Mark", "Luke", "John", "Acts", "R
omans", "Corinthians-1", "Corinthians-2",
              "Galatians", "Ephesians", "Philippians", "Col
ossians", "Thessalonians-1",
              "Thessalonians-2", "Timothy-1", "Timothy-2",
"Titus", "Philemon", "Hebrews",
              "James", "Peter-1", "Peter-2", "John-1", "Joh
n-2", "John-3", "Jude", "Revelation")

nt_Gospels <- c("Matthew", "Mark", "Luke", "John")
nt_History<- c("Acts")
nt_Letters <- c("Romans", "Corinthians-1", "Corinthians-2",
                "Galatians", "Ephesians", "Philippians",
"Colossians", "Thessalonians-1",
                "Thessalonians-2", "Timothy-1", "Timothy-
```

```
2", "Titus", "Philemon", "Hebrews",
                "James", "Peter-1", "Peter-2", "John-1",
"John-2", "John-3", "Jude")
nt_Prophecy <- c("Revelation")
```

구약과 신약에 대해 각각을 그룹으로 묶었다. 이제 이 내용들에 대해 Tf-idf를 구해보자

구약-법률

```
bible_token%>% #토큰화된 성경 데이터를 사용
  filter(Book %in% ot_Law)%>% #구약_법률에 대한 데이터를 필터링함
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하
고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계
산
  group_by(Book)%>% #책별로 그룹화를 진행
  slice_max(tf_idf, n = 2) %>% #그룹별(책별) 두 개의 값만 가지고
옴
  arrange(desc(tf_idf)) # 이후 정렬
```

	Book	word	n	tf	idf	tf_idf
	<chr>	<chr>	<int>	<dbl>	<dbl>	<dbl>
1	Genesis	abram	58	0.00164	1.61	0.00264
2	Exodus	cubits	42	0.00147	1.61	0.00237
3	Exodus	moses	295	0.0103	0.223	0.00231
4	Leviticus	value	29	0.00132	1.61	0.00212
5	Numbers	balak	32	0.00114	1.61	0.00184
6	Numbers	moses	228	0.00814	0.223	0.00182
7	Genesis	rachel	39	0.00110	1.61	0.00178
8	Leviticus	unclean	124	0.00564	0.223	0.00126
9	Deuteronomy	purge	11	0.000431	1.61	0.000694
10	Deuteronomy	jordan	30	0.00118	0.511	0.000601

창세기에서는 abram의 tf-idf가 가장 높게 나왔다.

구약-역사 그룹

```
bible_token%>% #토큰화된 성경 데이터를 사용
  filter(Book %in% ot_Law)%>% #구약_역사에 대한 데이터를 필터링함
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계산
  group_by(Book)%>% #책별로 그룹화를 진행
  slice_max(tf_idf, n = 2) %>% #그룹별(책별) 두 개의 값만 가지고옴
  arrange(desc(tf_idf)) # 이후 정렬
```

	Book	word	n	tf	idf	tf_idf
	<chr>	<chr>	<int>	<dbl>	<dbl>	<dbl>
1	Ruth	naomi	26	0.0111	2.48	0.0276
2	Esther	haman	44	0.00862	2.48	0.0214
3	Ruth	ruth	20	0.00856	2.48	0.0213
4	Esther	esther	42	0.00823	2.48	0.0205
5	Samuel-1	saul	256	0.0115	1.39	0.0159
6	Samuel-1	samuel	126	0.00566	1.39	0.00785
7	Judges	gideon	46	0.00284	2.48	0.00706
8	Kings-2	elisha	84	0.00391	1.79	0.00701
9	Judges	samson	36	0.00222	2.48	0.00552
10	Joshua	joshua	163	0.0101	0.539	0.00543

구약-시

```
bible_token%>% #토큰화된 성경 데이터를 사용
  filter(Book %in% ot_Poetry)%>% #구약_시에 대한 데이터를 필터링함
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계산
  group_by(Book)%>% #책별로 그룹화를 진행
  slice_max(tf_idf, n = 2) %>% #그룹별(책별) 두 개의 값만 가지고옴
```

```
  arrange(desc(tf_idf)) # 이후 정렬
```

```
  Book         word            n       tf   idf  tf_idf
  <chr>        <chr>       <int>    <dbl> <dbl>   <dbl>
1 Psalms       lord          784 0.0195   0.288 0.00561
2 Ecclesiastes meaningless    35 0.00697  0.693 0.00483
3 Job          job            52 0.00308  1.39  0.00427
4 Ecclesiastes chasing         9 0.00179  1.39  0.00249
5 Psalms       selah          71 0.00177  1.39  0.00245
6 Proverbs     lord           87 0.00613  0.288 0.00176
7 Job          replied        20 0.00118  1.39  0.00164
8 Proverbs     sluggard       13 0.000916 1.39  0.00127
```

구약-예언서

```
bible_token%>% #토큰화된 성경 데이터를 사용
  filter(Book %in% ot_Prophecy)%>% #구약_시에 대한 데이터를 필터링함
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계산
  group_by(Book)%>% #책별로 그룹화를 진행
  slice_max(tf_idf, n = 1) %>% #그룹별(책별) 한 개의 값만 가지고옴
  arrange(desc(tf_idf)) # 이후 정렬
```

```
   Book         word            n       tf   idf  tf_idf
   <chr>        <chr>       <int>    <dbl> <dbl>   <dbl>
 1 Jonah        jonah          17 0.0149   2.83  0.0421
 2 Haggai       haggai          9 0.00908  2.83  0.0257
 3 Obadiah      esau            5 0.00871  1.73  0.0151
 4 Daniel       daniel         69 0.00654  2.14  0.0140
 5 Nahum        nineveh         7 0.00647  1.45  0.00936
 6 Hosea        ephraim        36 0.00749  1.04  0.00780
 7 Ezekiel      cubits        123 0.00343  2.14  0.00733
```

```
 8 Jeremiah     jeremiah        127 0.00331  2.14  0.00708
 9 Malachi      says             26 0.0158   0.435 0.00688
10 Habakkuk     selah             3 0.00232  2.83  0.00658
11 Habakkuk     tolerate          3 0.00232  2.83  0.00658
12 Joel         locust            4 0.00220  2.83  0.00624
13 Zechariah    angel            21 0.00379  1.45  0.00549
14 Amos         amos              7 0.00185  2.83  0.00525
15 Lamentations affliction        6 0.00187  2.14  0.00401
16 Micah        transgression     5 0.00177  1.73  0.00306
17 Zephaniah    correction        2 0.00137  2.14  0.00294
18 Isaiah       isaiah           16 0.000465 2.83  0.00132
```

신약-복음서

```
bible_token%>% #토큰화된 성경 데이터를 사용
  filter(Book %in% nt_Gospels)%>% #신약_계율에 대한 데이터를 필터링함
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계산
  group_by(Book)%>% #책별로 그룹화를 진행
  arrange(desc(tf_idf))%>% # 이후 정렬
  head(20)
```

	Book	word	n	tf	idf	tf_idf
	<chr>	<chr>	<int>	<dbl>	<dbl>	<dbl>
1	John	remain	16	0.000862	1.39	0.00119
2	John	glorify	9	0.000485	1.39	0.000672
3	John	jewish	9	0.000485	1.39	0.000672
4	Matthew	swears	10	0.000442	1.39	0.000613
5	Matthew	talents	10	0.000442	1.39	0.000613
6	John	glorified	8	0.000431	1.39	0.000597
7	John	realize	8	0.000431	1.39	0.000597
8	Luke	elizabeth	10	0.000414	1.39	0.000573
9	John	hates	7	0.000377	1.39	0.000523

```
10 John    testifies     7 0.000377 1.39  0.000523
11 Matthew weeds         8 0.000354 1.39  0.000491
12 John    believes     13 0.000700 0.693 0.000485
13 John    lazarus      13 0.000700 0.693 0.000485
14 John    true         12 0.000646 0.693 0.000448
15 John    nathanael     6 0.000323 1.39  0.000448
16 John    nicodemus     6 0.000323 1.39  0.000448
17 Mark    whenever      4 0.000290 1.39  0.000402
18 Luke    manager       7 0.000290 1.39  0.000401
19 John    accepts       5 0.000269 1.39  0.000373
20 John    aramaic       5 0.000269 1.39  0.000373
```

신약-사도행전

```
bible_token%>% #토큰화된 성경 데이터를 사용
  filter(Book %in% nt_History)%>% #신약_사도행전에 대한 데이터를 필터링함
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계산
  arrange(desc(tf_idf)) # 이후 정렬
   Book  word      n     tf   idf tf_idf
   <chr> <chr> <int>  <dbl> <dbl>  <dbl>
 1 Acts  the    1546 0.0673     0      0
 2 Acts  and    1039 0.0453     0      0
 3 Acts  to      875 0.0381     0      0
 4 Acts  of      597 0.0260     0      0
 5 Acts  he      401 0.0175     0      0
 6 Acts  they    362 0.0158     0      0
 7 Acts  in      357 0.0155     0      0
 8 Acts  you     317 0.0138     0      0
 9 Acts  a       285 0.0124     0      0
10 Acts  him     271 0.0118     0      0
```

신약 사도행전에 대해서는 tf와 idf를 판별하기가 어려운 듯 보인다.

신약-편지

```
bible_token%>% #토큰화된 성경 데이터를 사용
  filter(Book %in% nt_Letters)%>% #신약_계율에 대한 데이터를 필터링함
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계산
  group_by(Book)%>% #책별로 그룹화를 진행
  slice_max(tf_idf, n = 1) %>% #그룹별(책별) 두 개의 값만 가지고옴
  arrange(desc(tf_idf))%>% # 이후 정렬
  head(20)
```

```
# Groups:   Book [19]
   Book            word           n      tf   idf  tf_idf
   <chr>           <chr>      <int>   <dbl> <dbl>   <dbl>
 1 John-2          lady           2 0.00709 3.04  0.0216
 2 John-3          friend         4 0.0133  1.44  0.0191
 3 Hebrews         priest        28 0.00403 3.04  0.0123
 4 Jude            ungodly        5 0.00840 1.10  0.00923
 5 Timothy-2       lovers         5 0.00292 3.04  0.00889
 6 Philemon        prisoner       3 0.00656 1.25  0.00822
 7 Galatians       law           34 0.0107  0.742 0.00797
 8 Titus           controlled     5 0.00507 1.44  0.00728
 9 Thessalonians-1 asleep         5 0.00267 2.35  0.00627
10 John-1          hates          5 0.00200 3.04  0.00608
11 Colossians      laodicea       4 0.00198 3.04  0.00602
12 Romans          law           73 0.00751 0.742 0.00557
13 Philippians     rejoice        8 0.00358 1.44  0.00513
14 Timothy-1       manage         4 0.00168 3.04  0.00512
15 Timothy-1       widow          4 0.00168 3.04  0.00512
16 Ephesians       realms         5 0.00163 3.04  0.00496
17 Thessalonians-2 idle           3 0.00283 1.66  0.00469
18 Peter-2         heavens        4 0.00260 1.66  0.00431
```

```
19 James           clothes        5 0.00220 1.95  0.00428
20 Peter-1         precious       4 0.00162 2.35  0.00381
```

신약-예언서

```
bible_token%>% #토큰화된 성경 데이터를 사용
  filter(Book %in% nt_Prophecy)%>% #신약_예언서서에 대한 데이터를 필터링함
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계산
  arrange(desc(tf_idf))%>% # 이후 정렬
  head(20)
   Book       word      n      tf   idf tf_idf
   <chr>      <chr> <int>   <dbl> <dbl>  <dbl>
 1 Revelation the    1023 0.0903      0      0
 2 Revelation and     643 0.0567      0      0
 3 Revelation of      460 0.0406      0      0
 4 Revelation to      255 0.0225      0      0
 5 Revelation a       198 0.0175      0      0
 6 Revelation i       178 0.0157      0      0
 7 Revelation who     169 0.0149      0      0
 8 Revelation in      158 0.0139      0      0
 9 Revelation will    148 0.0131      0      0
10 Revelation was     131 0.0116      0      0
11 Revelation his     124 0.0109      0      0
12 Revelation on      122 0.0108      0      0
13 Revelation he      119 0.0105      0      0
14 Revelation you     110 0.00971     0      0
15 Revelation they    105 0.00926     0      0
16 Revelation from     97 0.00856     0      0
17 Revelation for      90 0.00794     0      0
18 Revelation god      87 0.00768     0      0
```

```
19 Revelation is       87 0.00768     0      0
20 Revelation with     85 0.00750     0      0
```

Task 2-2 What can be inferred from the result of 2-1? Which groups are similar to each other ? Howdifferent are those groups of books? Explain the insight you obtain ed from the result?

우선, 단 하나의 그룹으로 묶인 신약 사도행전이나, 신약 예언서는 tf-idf를 구할수 없다. 왜냐하면 전체 그룹 안에서 book당 중요 단어를 찾아야 하는데, 사용할 수 있는 book이 하나뿐이니 제대로 된 text를 찾기 애매하다.

이를 해결하기 위한 방법으로, 사도행전과, 요한계시록은 성경의 전체 내용 혹은 신약 전체 내용에 대해 tf-idf를 수행한 후, 사도행전과 요한계시록만 필터링하는 방식을 사용할 수 있다.

```
bible_token%>% #토큰화된 성경 데이터를 사용
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계산
  filter(Book %in% nt_History)%>% #신약_사도행전에 대한 데이터를 필터링함
  arrange(desc(tf_idf)) # 이후 정렬
   Book  word          n       tf   idf  tf_idf
   <chr> <chr>     <int>    <dbl> <dbl>   <dbl>
 1 Acts  paul        172 0.00749  1.48  0.0111
 2 Acts  peter        71 0.00309  1.99  0.00616
 3 Acts  jews         68 0.00296  1.36  0.00402
 4 Acts  saul         32 0.00139  2.58  0.00360
 5 Acts  barnabas     29 0.00126  2.80  0.00354
 6 Acts  jesus        75 0.00327  0.932 0.00304
 7 Acts  disciples    26 0.00113  2.40  0.00272
 8 Acts  john         27 0.00118  2.24  0.00264
 9 Acts  antioch      19 0.000828 3.09  0.00256
10 Acts  festus       14 0.000610 4.19  0.00255
```

사도행전에 대해서 전체 성경에 대한 tf-idf를 수행해보니 paul, peter등의 단어가 가장 중요한 단어라고 나온다,

```
bible_token%>% #토큰화된 성경 데이터를 사용
  filter(Book %in% nt_books)%>%
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계산
  filter(Book %in% nt_History)%>% #신약_사도행전에 대한 데이터를 필터링함
  arrange(desc(tf_idf)) # 이후 정렬

# A tibble: 2,499 × 6
   Book  word          n       tf   idf  tf_idf
   <chr> <chr>     <int>    <dbl> <dbl>   <dbl>
 1 Acts  saul         32 0.00139  3.30  0.00459
 2 Acts  paul        172 0.00749  0.588 0.00440
 3 Acts  said        135 0.00588  0.588 0.00346
 4 Acts  peter        71 0.00309  1.10  0.00340
 5 Acts  jerusalem    59 0.00257  0.993 0.00255
 6 Acts  had         163 0.00710  0.351 0.00249
 7 Acts  went         78 0.00340  0.731 0.00248
 8 Acts  barnabas     29 0.00126  1.91  0.00241
 9 Acts  jews         68 0.00296  0.811 0.00240
10 Acts  ship         21 0.000915 2.60  0.00238
```

신약만 가지고 수행하면 saul, paul이 자주 등장하는데, said, had, went등의 명사가 아닌 동사들도 자주 등장하며 tf-idf가 상당히 높게 나온 특이한 점을 확인할 수 있었다.

```
bible_token%>% #토큰화된 성경 데이터를 사용

  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계
```

```
산
  filter(Book %in% nt_Prophecy)%>% #신약_예언서서에 대한 데이터를
필터링함
  arrange(desc(tf_idf))%>% # 이후 정렬
  head(20)
# A tibble: 20 × 6
   Book       word          n       tf   idf  tf_idf
   <chr>      <chr>     <int>    <dbl> <dbl>   <dbl>
 1 Revelation beast        39 0.00344  1.89  0.00649
 2 Revelation angel        56 0.00494  1.05  0.00521
 3 Revelation dragon       14 0.00124  4.19  0.00518
 4 Revelation 12,000       10 0.000882 4.19  0.00370
 5 Revelation throne       41 0.00362  1.01  0.00366
 6 Revelation lamb         31 0.00274  1.30  0.00355
 7 Revelation seven        55 0.00485  0.634 0.00308
 8 Revelation angels       23 0.00203  1.25  0.00253
 9 Revelation abyss         8 0.000706 3.50  0.00247
10 Revelation overcomes     8 0.000706 3.50  0.00247
11 Revelation churches     13 0.00115  2.11  0.00242
12 Revelation voice        33 0.00291  0.788 0.00230
13 Revelation plagues      10 0.000882 2.40  0.00212
14 Revelation golden       11 0.000971 2.11  0.00205
15 Revelation sounded      11 0.000971 2.11  0.00205
16 Revelation loud         21 0.00185  1.10  0.00204
17 Revelation white        17 0.00150  1.30  0.00195
18 Revelation peals         5 0.000441 4.19  0.00185
19 Revelation creatures    14 0.00124  1.48  0.00183
20 Revelation earth        69 0.00609  0.298 0.00181
```

요한계시록을 전체 성경에 대해 tf-idf를 진행했을 때,

```
bible_token%>% #토큰화된 성경 데이터를 사용
  filter(Book %in% nt_books)%>%
  count(Book,word, sort = TRUE)%>% # Book별로 단어수를 카운트하
고 정렬
  ungroup()%>% #count이후 book별 그룹을 해제
  bind_tf_idf(word,Book,n)%>% #Book의 word에 대한 tf-idf를 계
산
```

```
  filter(Book %in% nt_Prophecy)%>% #신약_예언서서에 대한 데이터를
필터링함
  arrange(desc(tf_idf))%>% # 이후 정렬
  head(20)
```

Book	word	n	tf	idf	tf_idf
<chr>	<chr>	<int>	<dbl>	<dbl>	<dbl>
1 Revelation	beast	39	0.00344	2.60	0.00896
2 Revelation	throne	41	0.00362	1.69	0.00610
3 Revelation	angel	56	0.00494	1.22	0.00601
4 Revelation	seven	55	0.00485	1.22	0.00590
5 Revelation	dragon	14	0.00124	3.30	0.00407
6 Revelation	earth	69	0.00609	0.657	0.00400
7 Revelation	saw	43	0.00379	0.993	0.00377
8 Revelation	lamb	31	0.00274	1.35	0.00369
9 Revelation	sounded	11	0.000971	3.30	0.00320
10 Revelation	four	24	0.00212	1.50	0.00319
11 Revelation	her	63	0.00556	0.523	0.00291
12 Revelation	12,000	10	0.000882	3.30	0.00291
13 Revelation	horns	10	0.000882	3.30	0.00291
14 Revelation	plagues	10	0.000882	3.30	0.00291
15 Revelation	voice	33	0.00291	0.993	0.00289
16 Revelation	smoke	12	0.00106	2.60	0.00276
17 Revelation	city	31	0.00274	0.993	0.00272
18 Revelation	creatures	14	0.00124	2.20	0.00271
19 Revelation	third	24	0.00212	1.22	0.00258
20 Revelation	white	17	0.00150	1.69	0.00253

신약에 대해서만 수행했을 때,

각 성경 그룹에 대해서는

구약의 법률 부분에서는 인물로 아브라함, 모세 등의 인물의 tf-idf가 높게 나와 중요한 인물임을 알 수 있었고, History부분에서는 룻의 어머니인 나오미나, 하만, 룻, 에스더 등등의 인물이 중요한 키워드로 뽑혔다. 이렇게 각각 키워드로서 중요한 인물이 자주 등장했다.

Task 3-1 Perform topic mode ling for both the New and Old Testaments separately. How the topic s are different compared to when you perform top ic modeling to entire Bible?

전체 성경에 대한 LDA,

```
install.packages("topicmodels")
install.packages("reshape2")
library(reshape2)
library(tidytext)
library(topicmodels)
LDA수행을 위한 라이브러리 설치
```

```
data("stop_words")
custom_stopwords <- tibble(word = c("the", "and", "of", "t
o", "you", "in","will","he","a","i","is","his",
                                    "for","they","your","wh
o","my","with","from","him","that","it"))
all_stopwords <- bind_rows(stop_words, custom_stopwords)
```

불용어를 처리하기 위함

```
bible_token_clean <- bible_token %>%
  anti_join(stop_words, by = c("word" = "word"))  %>%
  anti_join(all_stopwords, by = "word")
```

토큰화된 bible 데이터에 불용어들도 제거함

```
dtm <- bible_token_clean %>%
  count(Book, word) %>%  # 각 책(Book)에서 단어별 빈도 계산
  cast_dtm(Book, word, n)
```

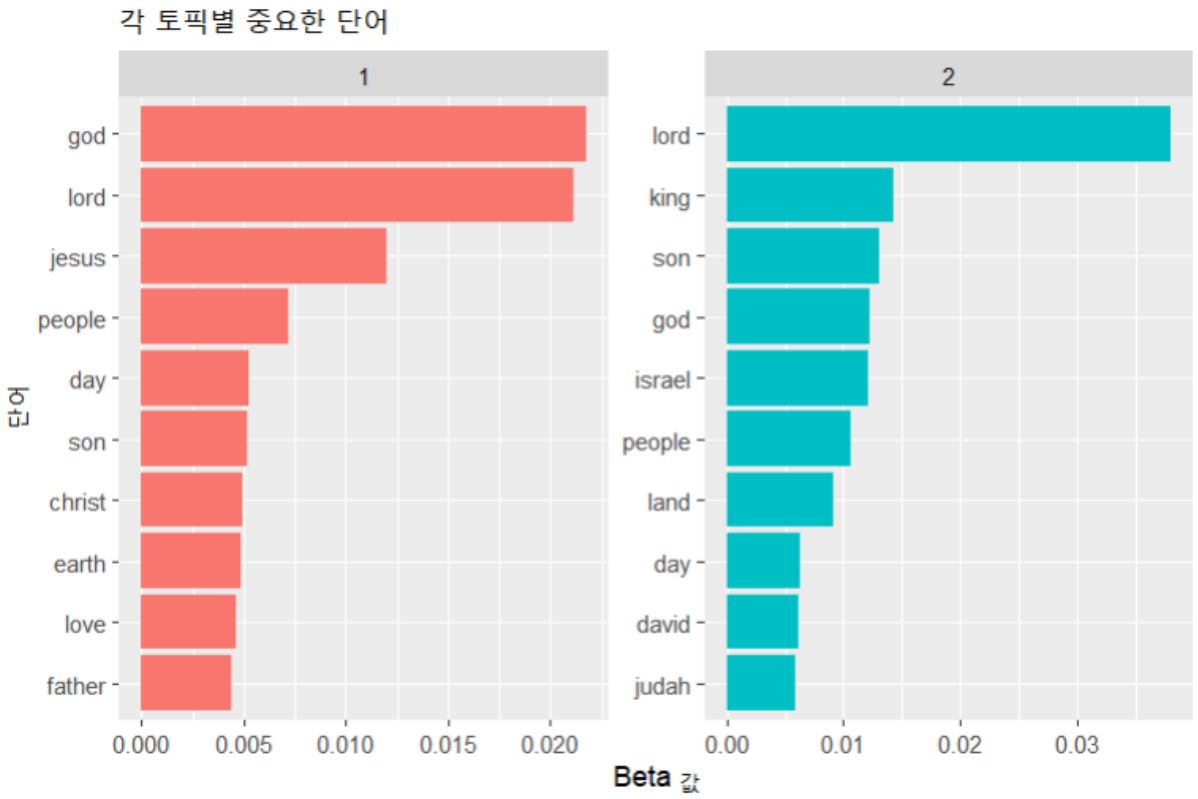
각 책에서 단어별 빈도를 계산한 후, book과 word간의 행렬을 생성하여 저장함

```
# LDA 모델 생성: k는 찾고자 하는 토픽의 수
lda_model <- LDA(dtm, k = 2, control = list(seed = 1234))
```

```
# 토픽별로 중요한 단어들 추출 (beta 값이 높은 단어들)
topics <- tidy(lda_model, matrix = "beta")
```

LDA모델을 생성함, 만들어둔 행렬에서 토픽 2개를 정해 LDA모델 생성 이후 토픽별로 가장 중요한 단어 추출 beta가 중요도에 따른 내용인듯

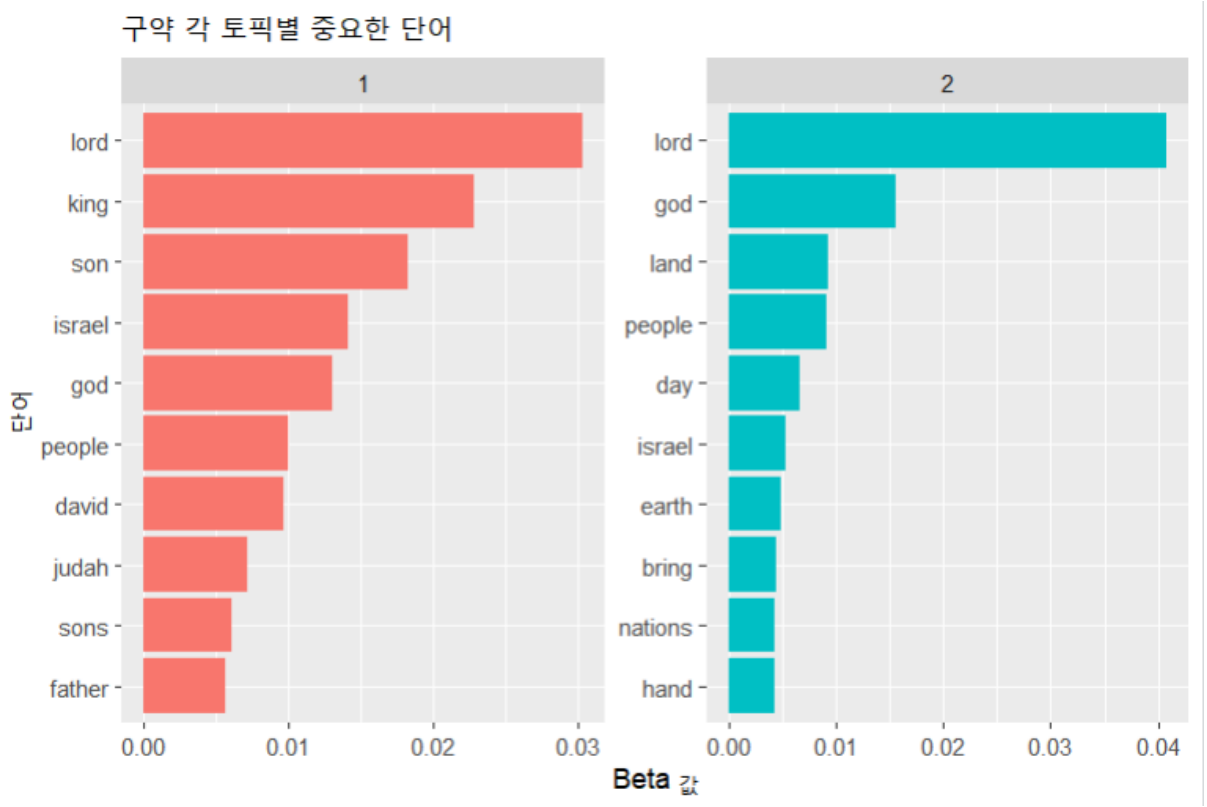
```
top_terms <- topics%>%
  group_by(topic)%>%
  top_n(10,beta)%>%
  ungroup()%>%
  arrange(topic,-beta)

top_terms %>%
  mutate(term = reorder_within(term,beta,topic))%>%
  ggplot(aes(x = reorder_within(term, beta,topic), y = bet
a, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip() +
  scale_x_reordered()+
  labs(x = "단어", y = "Beta 값", title = "각 토픽별 중요한 단
어")
```



성경 전체에 대해 2개의 토픽으로 나누어 가장 많이 사용된 단어들을 확인해 보았다. 단, 이것만으로는 뚜렷한 토픽별 그룹화의 기준을 잘 모르겠다.

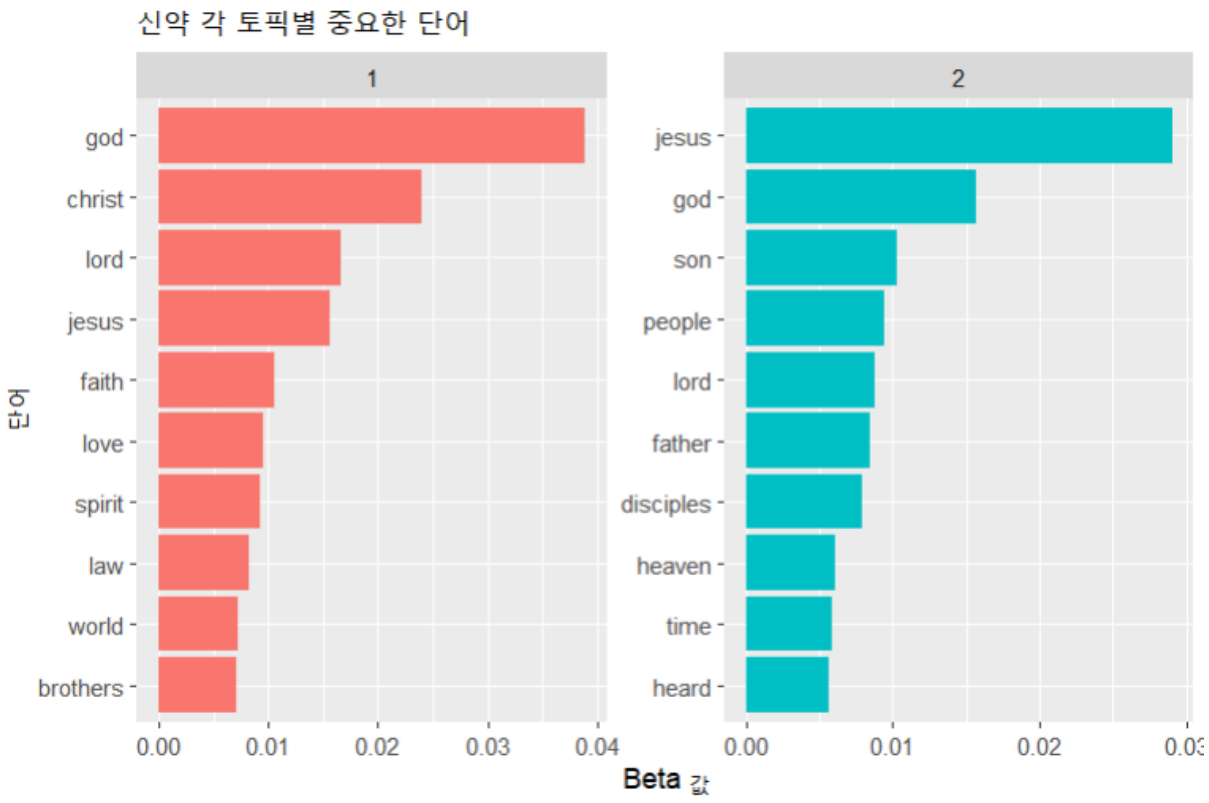
```
dtmot<-bible_token_clean%>%
  filter(Book %in% ot_books)%>%
  count(Book, word) %>%  # 각 책(Book)에서 단어별 빈도 계산
  cast_dtm(Book, word, n)
# LDA 모델 생성: k는 찾고자 하는 토픽의 수
lda_model <- LDA(dtmot, k = 2, control = list(seed = 1234))

# 토픽별로 중요한 단어들 추출 (beta 값이 높은 단어들)
topics <- tidy(lda_model, matrix = "beta")

head(topics)

topics%>%
  group_by(topic)%>%
  top_n(10,beta)%>%
  ungroup()%>%
  arrange(topic,-beta)
```

```
top_terms <- topics%>%
  group_by(topic)%>%
  top_n(10,beta)%>%
  ungroup()%>%
  arrange(topic,-beta)
top_terms %>%
  mutate(term = reorder_within(term,beta,topic))%>%
  ggplot(aes(x = reorder_within(term, beta,topic), y = bet
a, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip() +
  scale_x_reordered()+
  labs(x = "단어", y = "Beta 값", title = "구약 각 토픽별 중요한
단어")
```



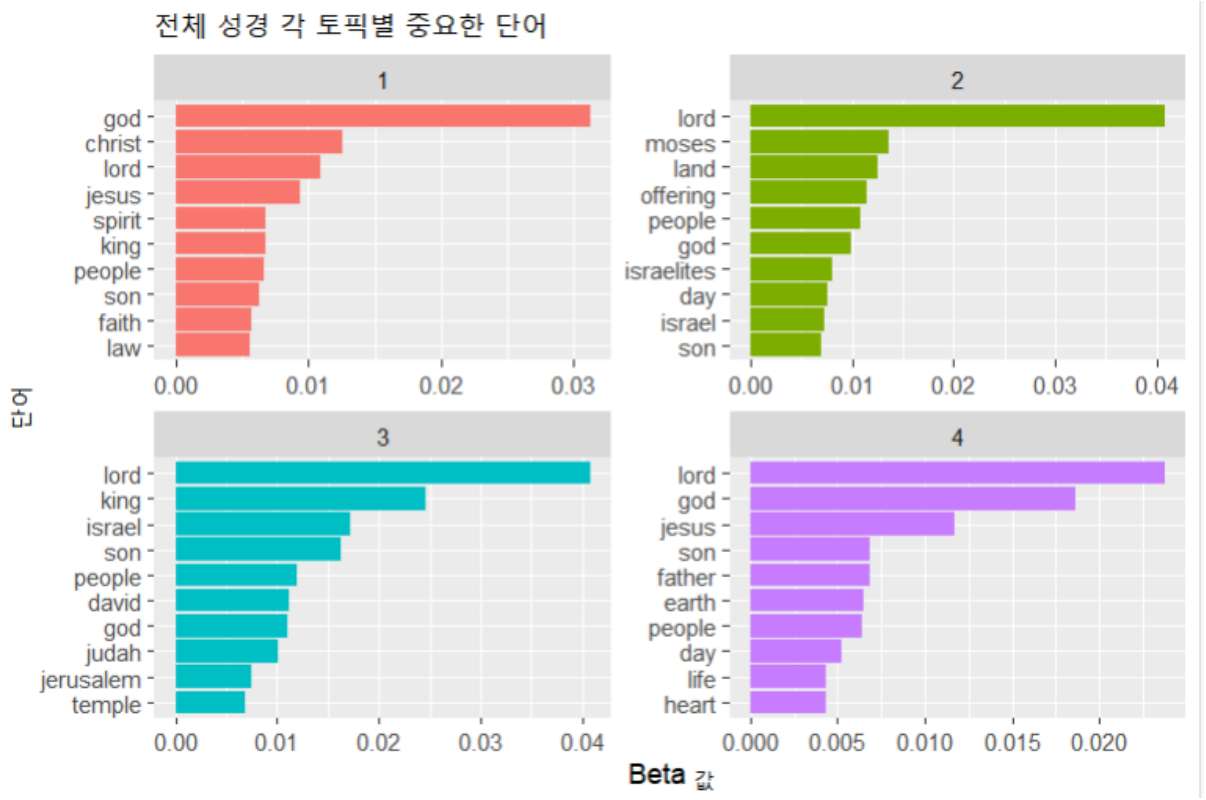
동일한 과정을 구약만 필터링한 후 시각화 하였다.

```r
dtmnt <-bible_token_clean%>%
  filter(Book %in% nt_books)%>%
  count(Book, word) %>%  # 각 책(Book)에서 단어별 빈도 계산
  cast_dtm(Book, word, n)

# LDA 모델 생성: k는 찾고자 하는 토픽의 수
lda_model <- LDA(dtmnt, k = 2, control = list(seed = 1234))

# 토픽별로 중요한 단어들 추출 (beta 값이 높은 단어들)
topics <- tidy(lda_model, matrix = "beta")

head(topics)

topics%>%
  group_by(topic)%>%
  top_n(10,beta)%>%
  ungroup()%>%
  arrange(topic,-beta)

top_terms <- topics%>%
  group_by(topic)%>%
  top_n(10,beta)%>%
  ungroup()%>%
  arrange(topic,-beta)

top_terms %>%
  mutate(term = reorder_within(term,beta,topic))%>%
  ggplot(aes(x = reorder_within(term, beta,topic), y = bet
a, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip() +
  scale_x_reordered()+
  labs(x = "단어", y = "Beta 값", title = "신약 각 토픽별 중요한
단어")
```

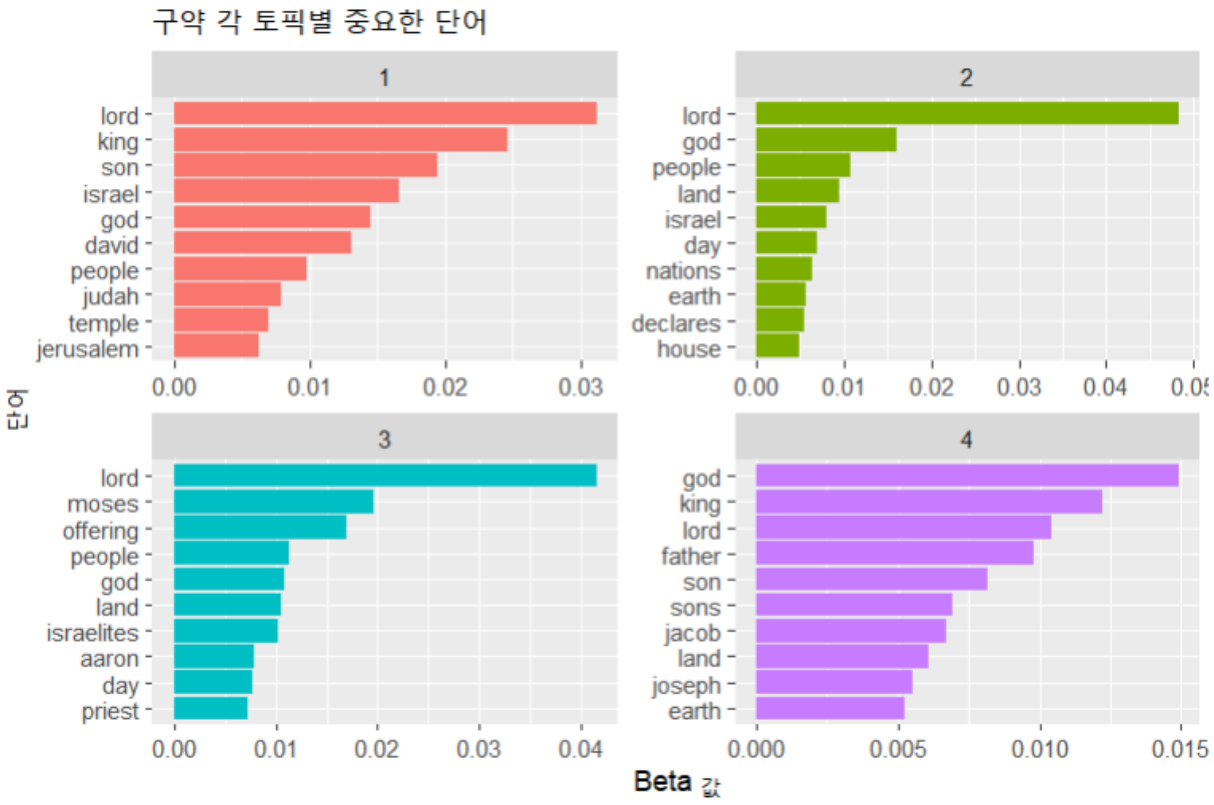


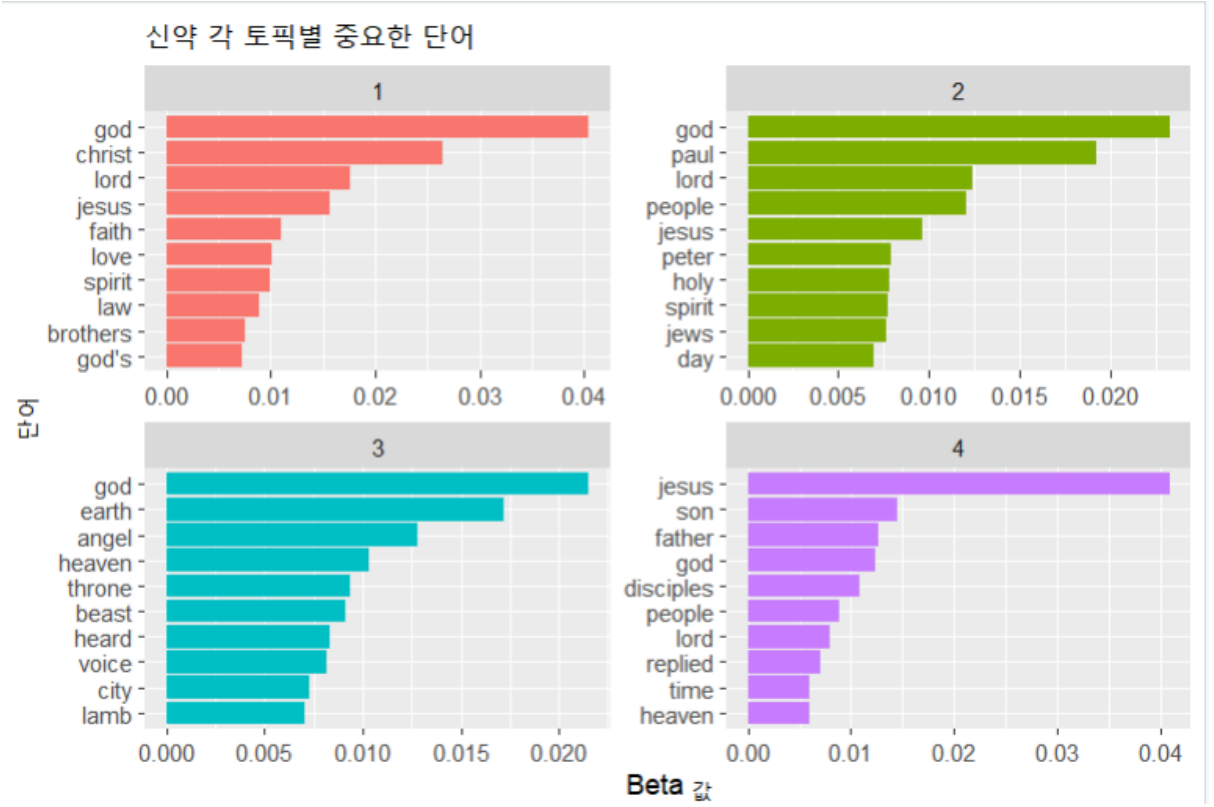
3-2 Is your result of topic modeling easily interpretable? If not, how can y ou adjust your work to get more interpreta ble result? Try and learn fro m errors.

구약이든 신약이든 토픽의 수가 2개뿐이기에 분류를 나누는 방법에도 2가지밖에 나타나지 않은 것 같다. 이번에는 분류를 4개로 늘려보자



전체 성경에 대해 토픽을 4개로 늘렸다는 것은, 전체 성경의 텍스트 단어를 4가지로 그룹화를 시킨 것이며, 1번에서 등장하는 christ, lord, jesus나 spirit, king son등으로 보아 창세기부터 시작하는 법률에 대한 내용이 이 토픽에 들어간것이 아닌가 하고, 토픽2에서는 moses, israelites, israel등으로 보아 출애굽기부터 가나안에 이르는 여정에 대한 토픽이 아닌가 싶다. 그러나 전체적으로 lord나 son, god 등의 단어가 많이 사용되다 보니 중요도가 높다고 인식되었고, 또 불용어 처리를 하지 않았을때는 the, a같은 단어들의 중요도가 높다고 나왔기 때문에 정확한 기준을 알기 다





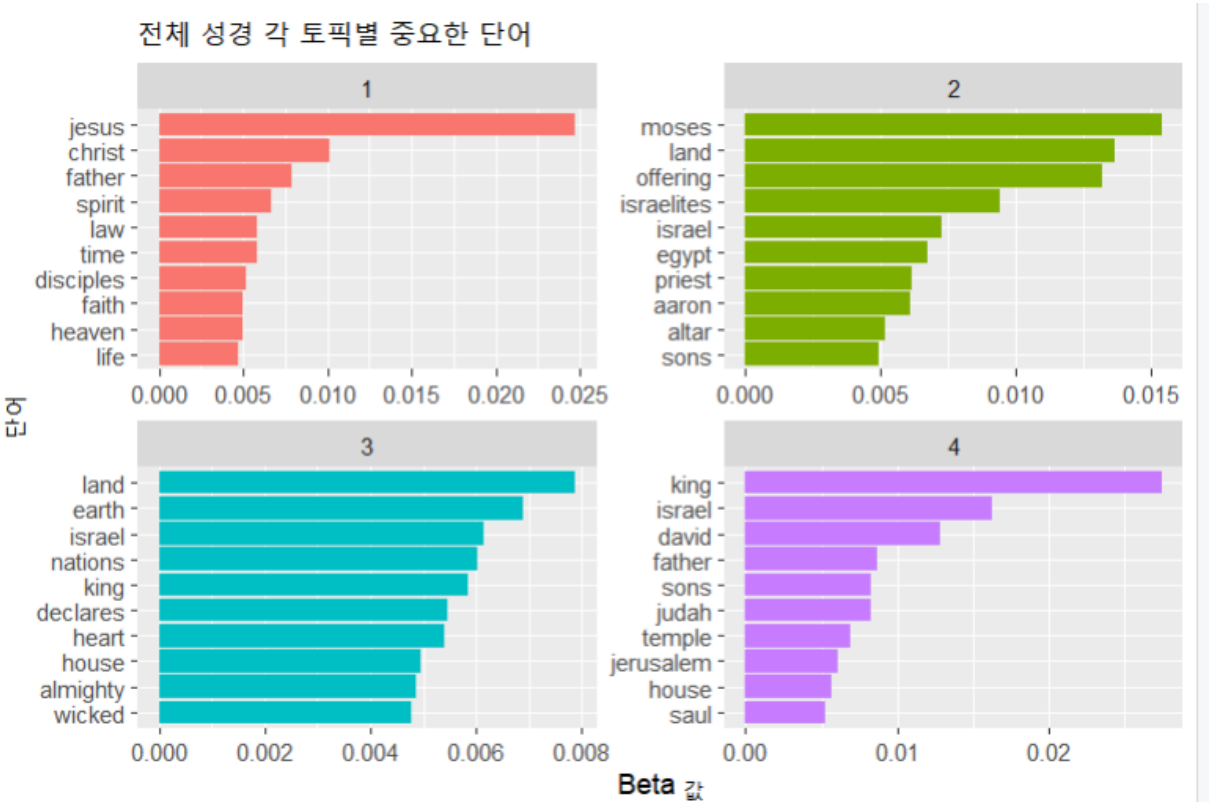
사용된 단어가 지속적으로 사용되는 것도 많은 것 같다. 특히 성경의 특수성때문에 god, jesus, load등의 단어가 많이 등장하는 것 같다.

불용어에 자주 등장하는 단어들을 포함시킨 후 다시 LDA를 수행하면 다른 결과가 나올까?

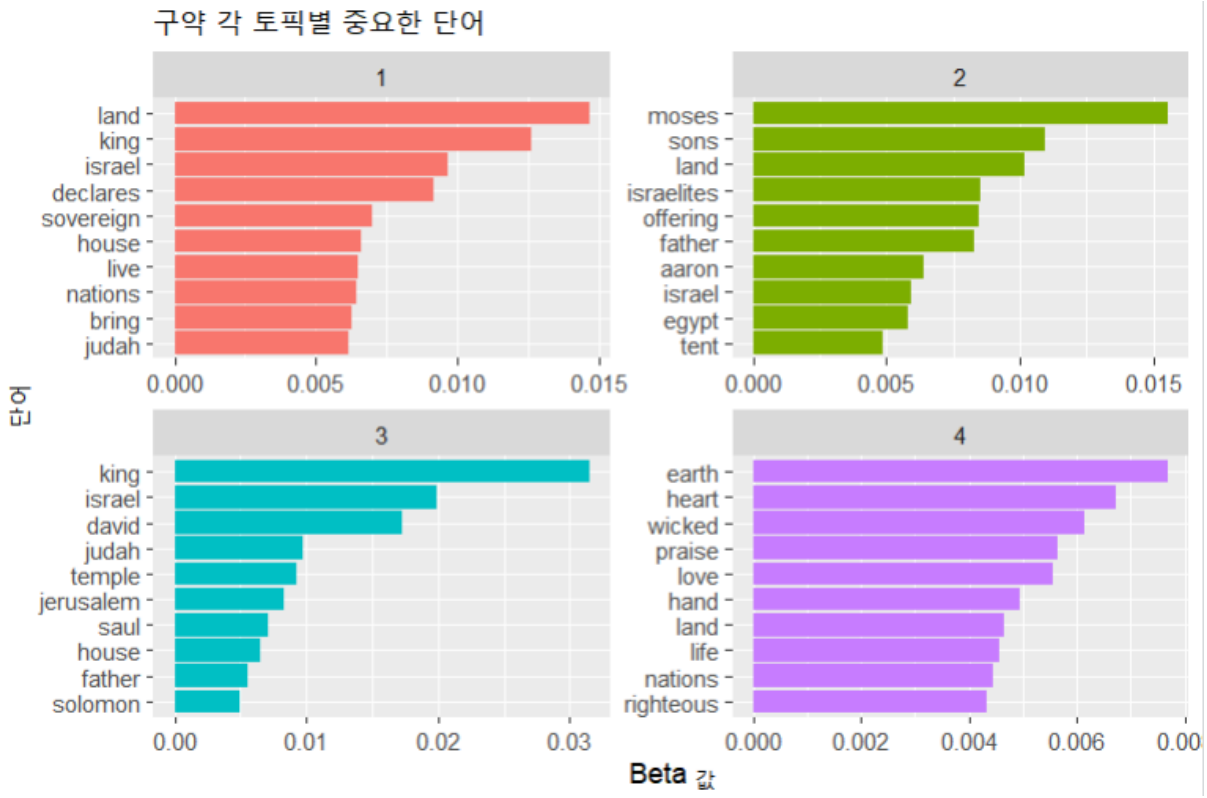
```
custom_stopwords <- tibble(word = c("the", "and", "of", "t
o", "you", "in","will","he","a","i","is","his",
                                   "for","they","your","wh
o","my","with","from","him","that","it","lord",
                                   "god","people","day","s
on"))
all_stopwords <- bind_rows(stop_words, custom_stopwords)


bible_token_clean <- bible_token %>%
  anti_join(stop_words, by = c("word" = "word"))  %>%
  anti_join(all_stopwords, by = "word")
```

```
dtm <- bible_token_clean %>%
  count(Book, word) %>%  # 각 책(Book)에서 단어별 빈도 계산
  cast_dtm(Book, word, n)

# LDA 모델 생성: k는 찾고자 하는 토픽의 수
lda_model <- LDA(dtm, k = 4, control = list(seed = 1234))

# 토픽별로 중요한 단어들 추출 (beta 값이 높은 단어들)
topics <- tidy(lda_model, matrix = "beta")

top_terms <- topics%>%
  group_by(topic)%>%
  top_n(10,beta)%>%
  ungroup()%>%
  arrange(topic,-beta)

top_terms %>%
  mutate(term = reorder_within(term,beta,topic))%>%
  ggplot(aes(x = reorder_within(term, beta,topic), y = bet
a, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip() +
```

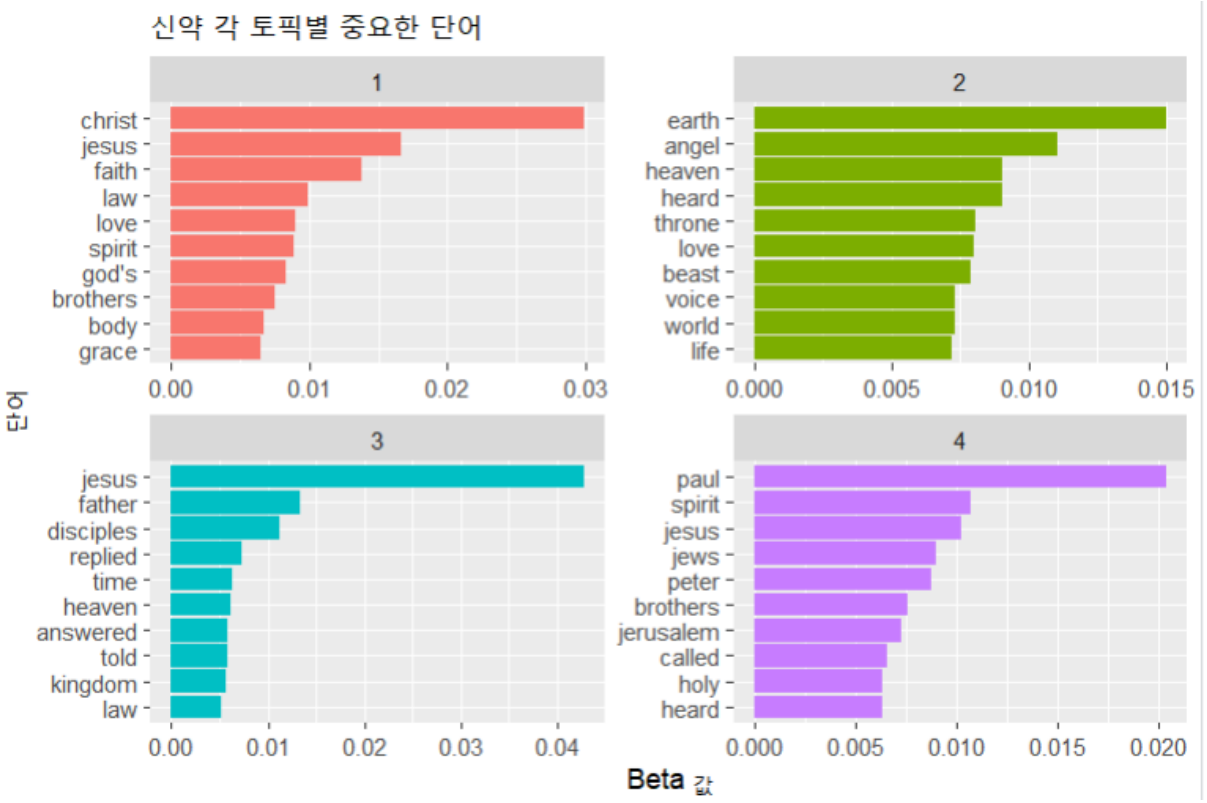
```
  scale_x_reordered()+
  labs(x = "단어", y = "Beta 값", title = "전체 성경 각 토픽별
중요한 단어")
```



확실히 각 토픽별로 중복되는 단어는 많이 줄어들었다,



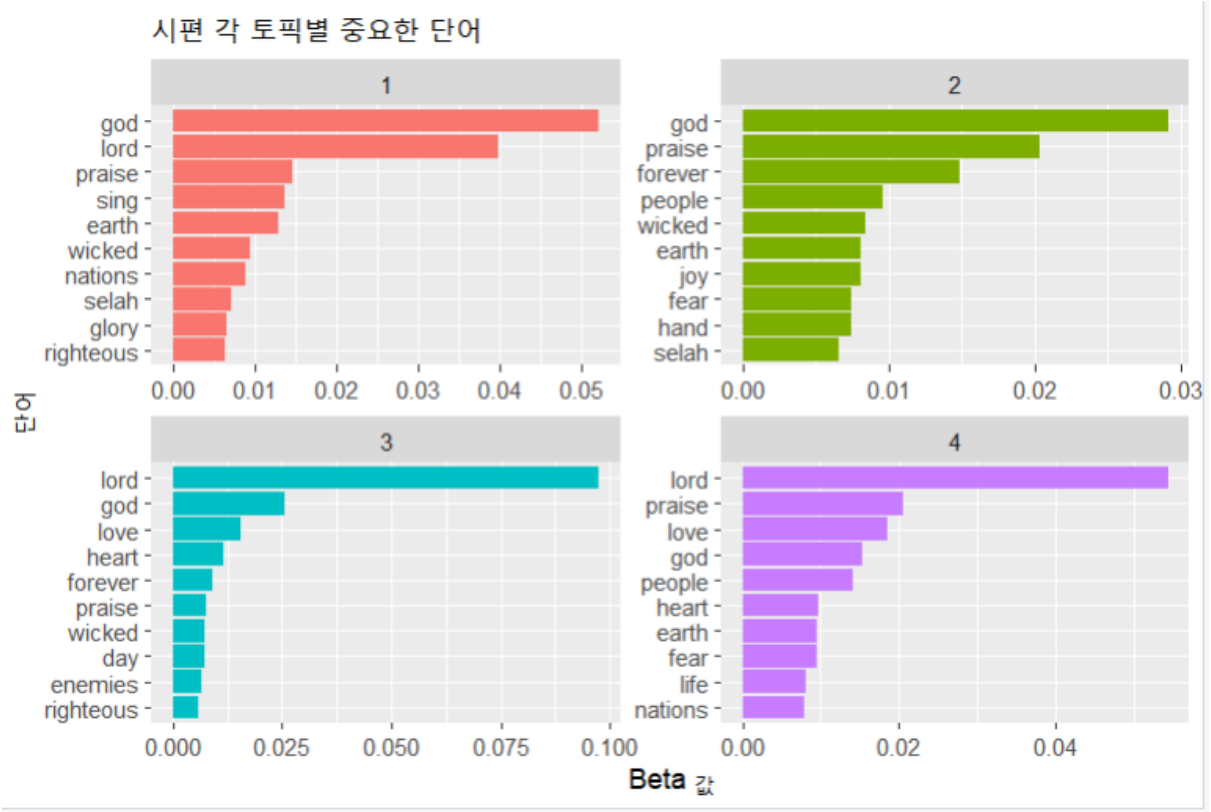
구약의 각 토픽별로 중요한 단어를 확인하면, 이스라엘과 왕, 국가, 유다 등이 하나의 토픽으로 구별되었고, 모세와 아들, 이집트 등이 나온 출애굽기에 대한 내용과 부분이 2번 토픽으로 나온 것 같다. 3번 토픽에서는 유다와 신전, 이스라엘이나 사울 부모 솔로몬 등이 등장하여 시편과 솔로몬의 노래 부분의 토픽이 들어간 것 같고, 4번 토픽에서는 earth, heart, wicked등의 단어가 구별되어 들어간 것 같다.



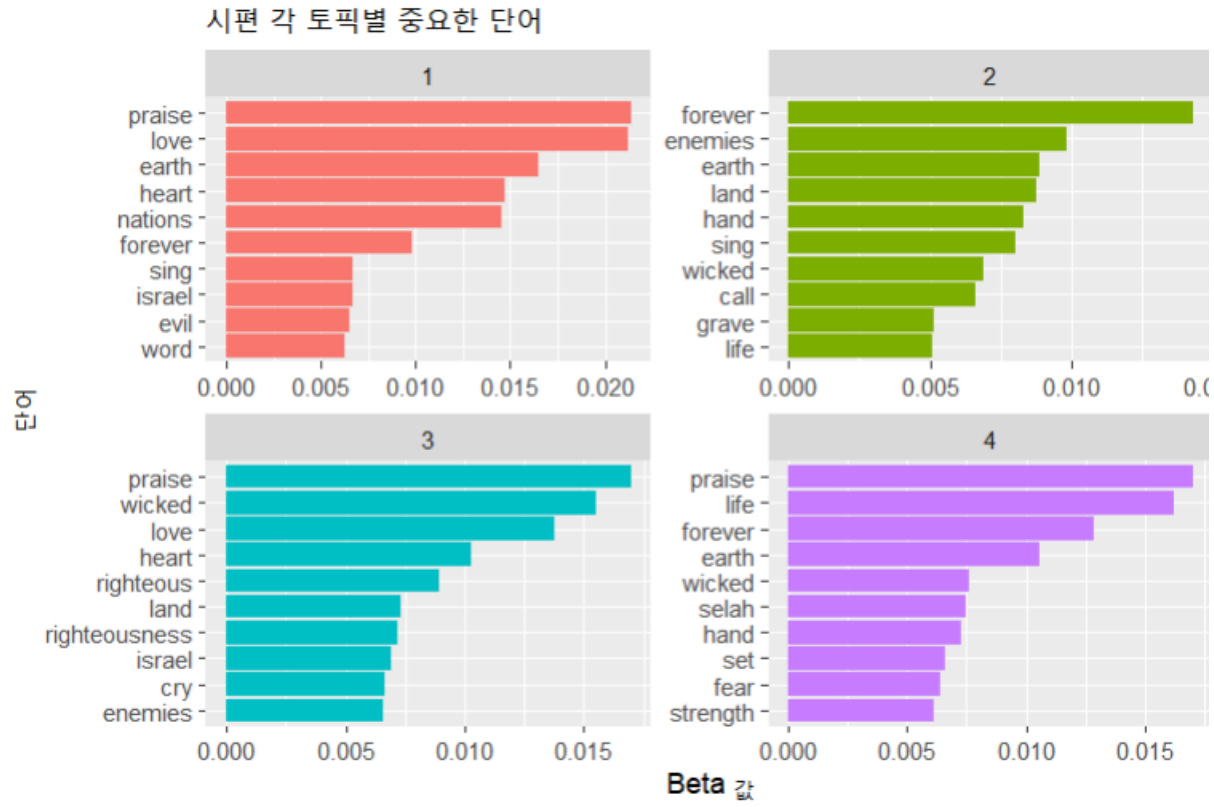
신약에서도 마찬가지로 자주 등장하는 단어들에 대해 불용어 처리를 하였고, 각 토픽별 중요한 단어 순서를 시각화 하였는데, 1번 토픽에서는 크리스트, 지져스 운명, 법 등의 단어가 중요도가 높은 것으로 나왔고, 2번 토픽에서는 천국과 천사들에 대한 내용이 주로 등장하였다, 3번 토픽에서는 예수님을 뜻하는 jesus가 매우 중요도가 높게 나왔고 부모, 제자들 등의 내용이 많이 나왔다. 마지막 4번 토픽에서는 paul과 spirit, jews 등의 내용이 많이 등장하였다.

3-3 Choose books of bible of your interest and perform topic modeling on the specific parts. Share your result and insight you obtained about the Bible.

시편에 대해 한번 LDA를 해 보았다.



다윗이나 사울 왕, 등등에 대한 내용이 나올 줄 알았는데, 그렇지 않은 것 같다. 불용어를 추가로 처리한 후 다시 확인하자



세 토픽에서 자주 중복되는 단어가 praise가 있고 그 외에는 sing,wicked,등등 시편이 다윗의 시인 만큼 찬양과 노래에 대한 단어들이 자주 사용되며 이를 통해 각 토픽별 단어를 확인할 수 있는 것 같다.