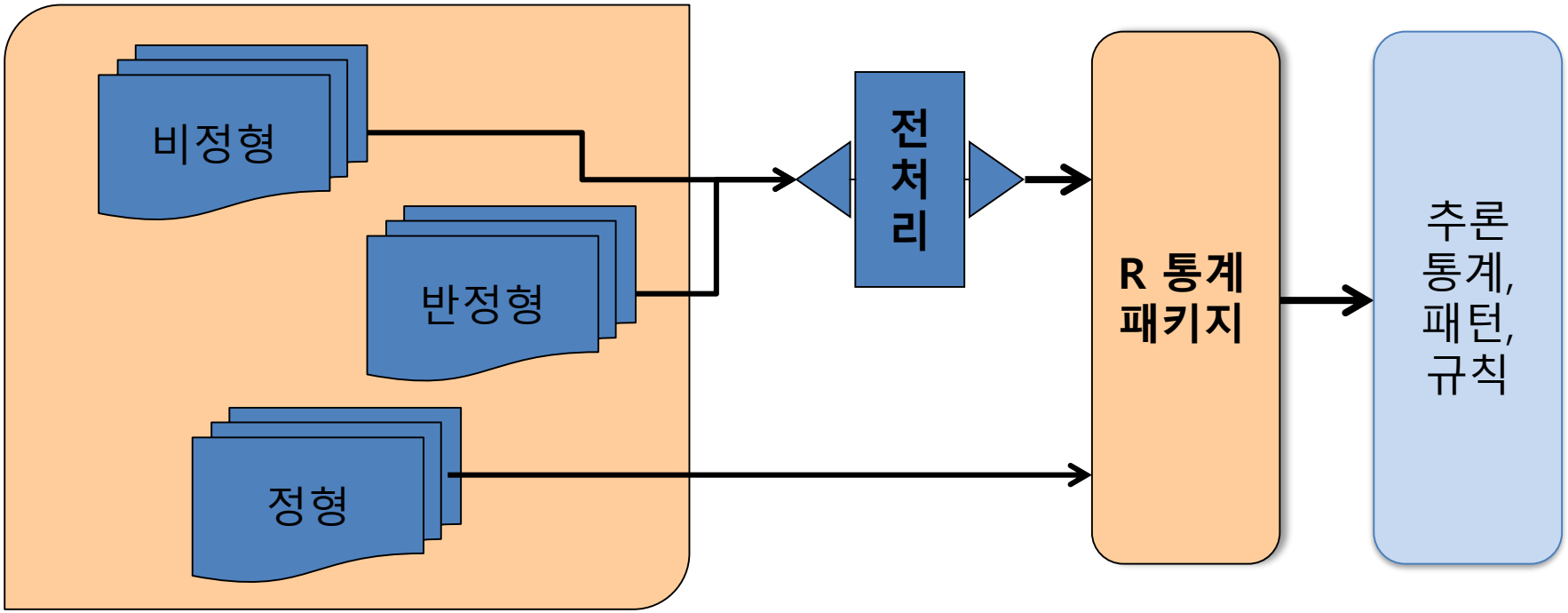
정형/비정형 데이터

목 차

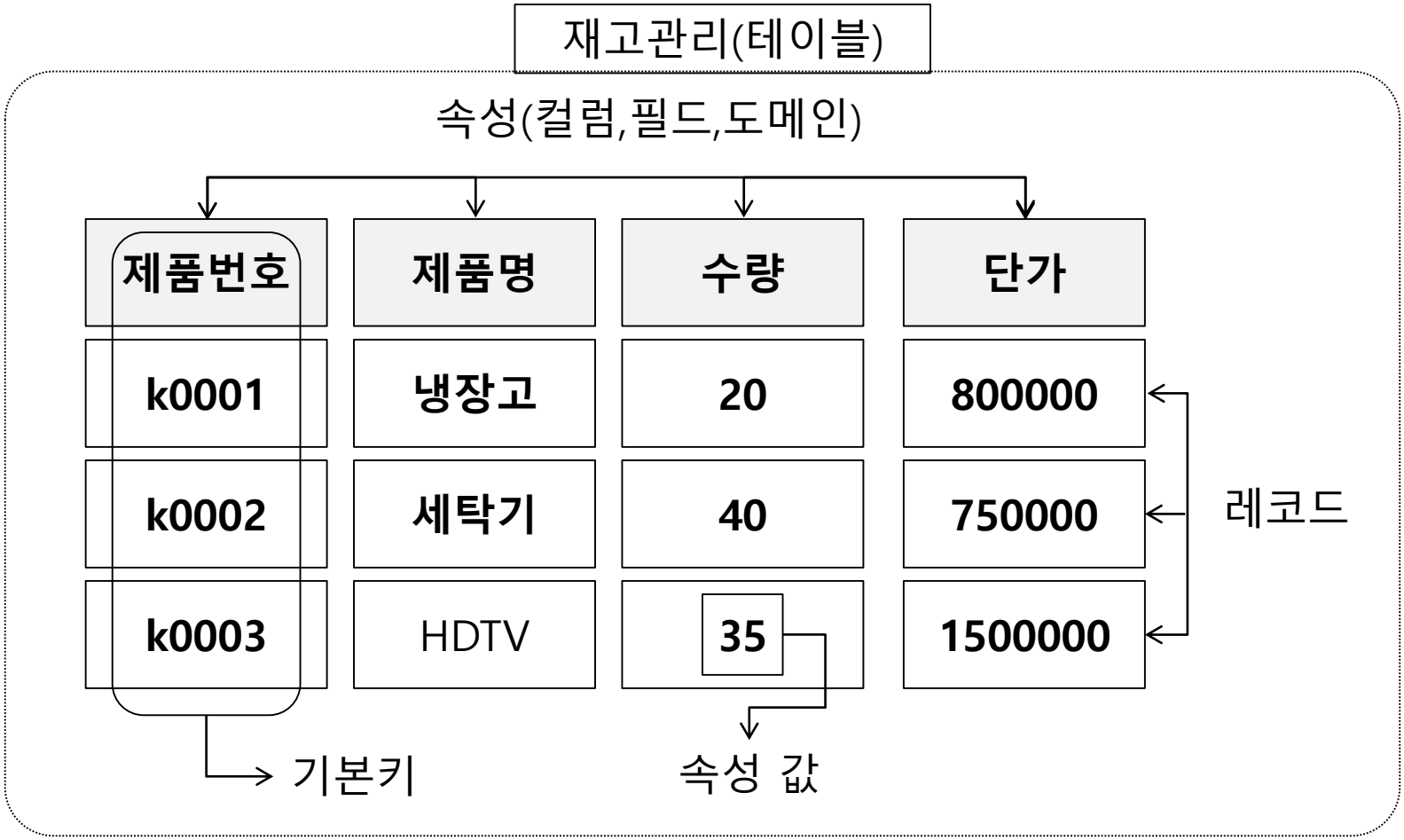
정형/비정형 데이터

- 정형과 비정형 데이터 처리 과정



정형 데이터 처리

- 관계형 데이터베이스의 테이블 구조



정형 데이터 처리

1) 정형 데이터(RDB-Oracle)

① 패키지 설치

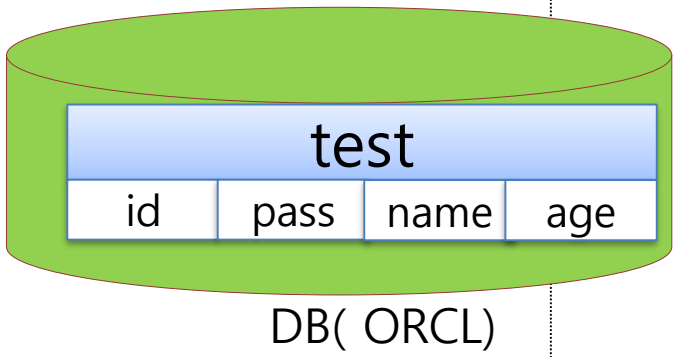
```
# RJDBC 패키지를 사용하기 위해서는 우선 java를 설치해야 한다.
install.packages("rJava")
#install.packages("DBI")
install.packages("RJDBC")

# 패키지 로딩
library(DBI)
Sys.setenv(JAVA_HOME='C:₩₩Program Files₩₩Java₩₩jre1.8.0_xxx')
library(rJava)
library(RJDBC) # rJava에 의존적이다.(rJava 먼저 로딩)
```

정형 데이터 처리

② Oracle 설치(DB:orcl, id: scott, password: tiger)

③ table 생성/레코드 추가 정형 데이터(RDB-Oracle)

```
create table test(
    id varchar(20) primary key,
    pass varchar(20) not null,
    name varchar(20) not null,
    age number(2)
    );

    insert into test values('hong','1234','홍길동',35);
    insert into test values('lee','1234','이순신',45);
```



정형 데이터 처리

④ Oracle 연동

```
# driver
drv<-JDBC("oracle.jdbc.driver.OracleDriver",
  "C:/oraclexe/app/oracle/product/11.2.0/server/jdbc/lib/ojdbc6.jar")
# db연동(driver, url,uid,upwd)
conn<-dbConnect(drv,
        jdbc:oracle:thin:@//127.0.0.1:1521/xe","scott","tiger")
query = "SELECT * FROM test"
dbGetQuery(conn, query)
#     ID   PASS NAME  AGE
#1 hong 1234  홍길동  35
#2  lee   1234  이순신  45
```

정형 데이터 처리

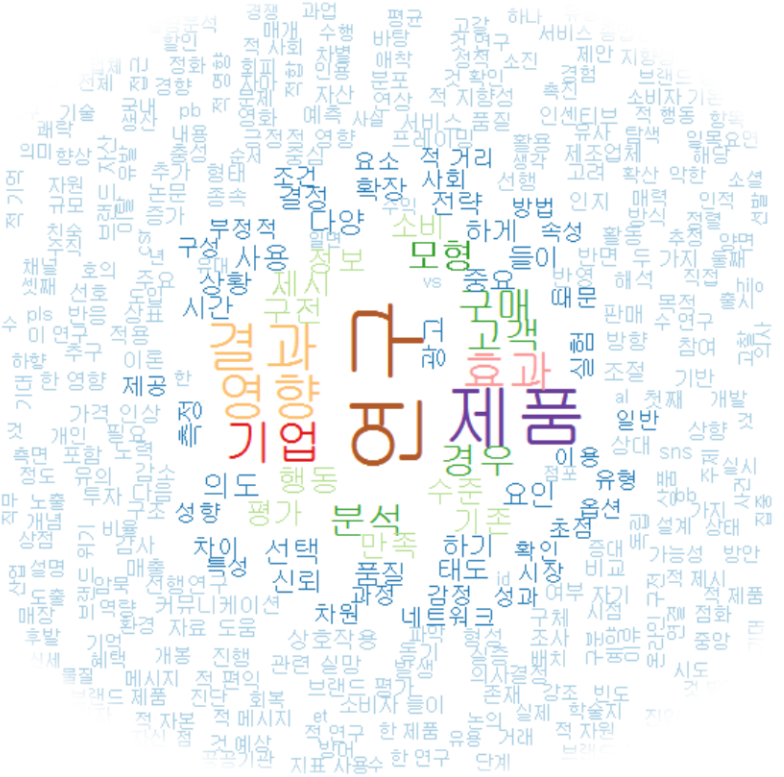
```
# id 내림차순 정렬
query = "SELECT * FROM test order by id desc"
dbGetQuery(conn, query)

# ID   PWD    NAME
#1   yoogs 3333 유관순
#2    test 1111    test
#3   leess 2222 이순신
#4   kimys 4444 김유신
#5 honggd 1111 홍길동
```

정형 데이터 처리

```
######################## MySql ####################
library(DBI)
library(rJava)
library(RJDBC)
drv <- JDBC("com.mysql.jdbc.Driver", "/usr/share/java/mysql-connector-java.jar", identifier.quote="`")
conn <- dbConnect(drv, "jdbc:mysql://<db_ip>:<db_port>/<dbname>", "<id>", "<passwd>")
df.table <- dbGetQuery(conn, "select * from DBTABLE")
df.table
##########################################################
```

비정형 데이터 처리

2) 비정형 데이터 – 텍스트 마이닝 분석



비정형 데이터 처리

- SNS 데이터 분석(텍스트 마이닝) 특징
  - ✓ Social 데이터, 디지털데이터를 대상으로 미리 만들어 놓은 사전을 비교하여 단어의 빈도를 분석한다.
  - ✓ 한계점 : 사전 작성이 어려움
  - ✓ KoNLP : 한글 자연어 처리 사전, 세종사전(카이스트 개발) 적용
    - 상용프로그램 사용 권장
  - ✓ tm : 영문 텍스트 마이닝 패키지
  - ✓ 데이터 Crawling 시스템 or 전문 사이트 의뢰 -> 데이터 수집

비정형 데이터 처리

- SNS / 문헌 데이터 분석 절차

  단계1 : 토픽분석(단어의 빈도수)

  - 형태소 분석으로 사전에 단어 추가
  - 사전과 텍스트 데이터 비교 → 단어 빈도 분석
  - 시각화 : Wordcloud

  단계2 : 연관어 분석(관련 단어 분석)

  - 특정 단어의 연관단어 빈도 분석
  - 시각화 : 단어를 기준으로 망 형태로 시각화

  단계3 : 감성 분석(단어의 긍정/부정 분석)

  - 시각화 : 긍정(파랑), 부정(빨강) -> 불만고객 시각화

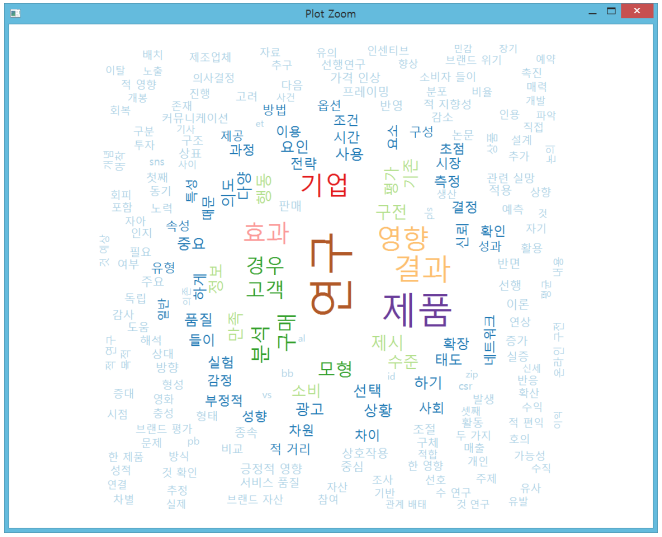
❖ 형태소 분석 : 문장을 분해 가능한 최소한의 단위로 분리하는 작업

비정형 데이터 처리

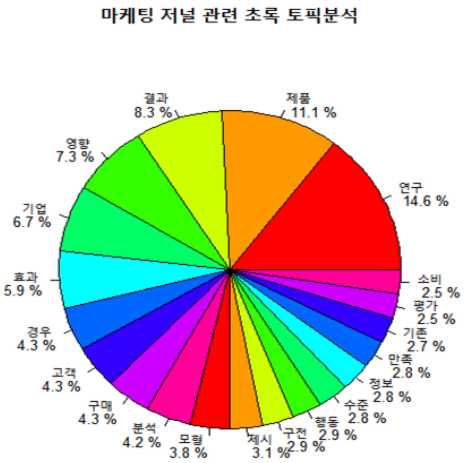
```
##############################################
#     단계1 - 토픽분석(텍스트 마이닝)                    #
#          √ 시각화 : 단어 빈도수에 따른 WordCloud      #
##############################################
```

비정형 데이터 처리

단계1 : 토픽분석(결과물)



단어구름(Wordcloud)



Pie 차트 시각화

비정형 데이터 처리

- 토픽분석을 위한 패키지 설치

1. java install : http://www.oracle.com/index.html(Oracle 사이트)
   -> java 프로그램 설치(64비트 환경 - R(64bit) - java(64bit))

2. rJava 설치 : R에서 java 사용을 위한 패키지

```
install.packages("rJava")
Sys.setenv(JAVA_HOME='C:₩₩Program Files₩₩Java₩₩jre1.8.0_xxx')
library(rJava) # 로딩
```

3. install.packages

```
install.packages(c("KoNLP", "tm", "wordcloud"))
# KoNLP - 한글처리 패키지, (자바기반-> rJava 패키지 설치되어야 함)
# tm - 텍스트 마이닝 패키지
# wordcloud - 단어구름 패키지(결과 출력)
```

비정형 데이터 처리

4. 패키지 설치 확인

```
library(KoNLP)
library(tm)
library(wordcloud)

# KoNLP에서 제공하는 명사 추출 함수
extractNoun("안녕하세요. 홍길동 입니다.") # 명사만 추출하는 함수
# [1] "안녕"    "홍길동"
```

비정형 데이터 처리

1. 데이터셋(abstract.txt) 가져오기

```
data = read.csv("C:/workspaces/Rwork/data/abstract.txt",
                header=TRUE, stringsAsFactors=FALSE)
# stringsAsFactors=FALSE : string을 범주로 사용하지 않음
data
str(data)
# data.frame : 300 obs. of  4 variables: - 행/열 데이터 (관찰치 300개, 변수 4개)
```

데이터 셋(abstract.txt) 설명
- 경영학관련 저널에서 초록만 300개 추출-저널,년도,초록
- 트위터보다 검증된 텍스트 내용

비정형 데이터 처리

2. 데이터 셋 대상 자료집(documents)생성

```
# Corpus( ) : 벡터 대상 자료집(documents)생성 함수, tm 패키지 제공
result.text <- Corpus(VectorSource(data[1:100,4]))
# 4번째 컬럼(abstract)만 100개 추출하여 corpus(자료집) 생성
result.text
# <<VCorpus (documents: 100, metadata (corpus/indexed): 0/0)>>
```

3. 분석 대상 자료집을 대상으로 NA를 공백으로 처리

```
result.text[is.na(result.text)]    <- ""
result.text # documents: 100
```

비정형 데이터 처리

4. 세종 사전에 단어 추가

```
# 세종 사전 불러오기
useSejongDic() # 87007 word 제공
#Backup was just finished!
#87007 words were added to dic_user.txt

# # 세종 사전에 없는 단어 추가
mergeUserDic(data.frame(c("비정규직","빅데이터", "한미fta"), c("ncn")))
# ncn –명사지시코드
# 3 words were added to dic_user.txt.
```

비정형 데이터 처리

5. 단어 추출 사용자 함수 정의 및 단어 추출

```
# (1) 함수 실행 순선 : 단어추출 -> 문자변환 -> 공백으로 합침
exNouns<- function(x) { paste(extractNoun(as.character(x)), collapse=" ")}

# (2) exNouns 함수 이용 단어 추출
# 형식) sapply(적용 데이터, 적용함수) -> 요약 100개를 대상으로 단어 추출
result_nouns <- sapply(result.text, exNouns) # 벡터 타입으로 단어 추출
#Warning messages:

# (3) 단어 추출 결과
result_nouns[1]  # 1번째 백터 요소 보기
# [1] "타인 도움 사람 호 도움 감사 빚 감정 이 보답 ....
```

비정형 데이터 처리

6. 데이터 전처리(부호, 수치, 불용어 제거)

```
# 추출된 단어로 자료집 다시 생성
myCorpus <- Corpus(VectorSource(result_nouns))
myCorpus # <<VCorpus (documents: 100, metadata (corpus/indexed): 0/0)>>

myCorpus <- tm_map(myCorpus, removePunctuation) # 문장부호 제거
myCorpus <- tm_map(myCorpus, removeNumbers) # 수치 제거
myCorpus <- tm_map(myCorpus, tolower) # 소문자 변경
myCorpus <-tm_map(myCorpus, removeWords, stopwords('english'))
# 불용어제거 :  for, very, and, of, are 등

inspect(myCorpus[1:5]) # 데이터 전처리 결과 확인
```

비정형 데이터 처리

7. 단어 선별(단어 길이 2개 이상)

```
# PlainTextDocument 함수를 이용하여 myCorpus를 일반문서로 변경
myCorpus<-tm_map(myCorpus, PlainTextDocument)
myCorpus

# TermDocumentMatrix() : 일반텍스트문서를 대상으로 단어 선별
# 단어길이 2개 이상인 단어만 선별 -> matrix 변경
myTdm <- TermDocumentMatrix(myCorpus, control=list(wordLengths=c(2,Inf)))
myTdm  # (terms: 4791, documents: 100)>> 단어 : 4791,  문서: 100

# matrix -> data.frame 변경
mat <- as.data.frame(as.matrix(myTdm))
mat
dim(mat) # [1] 4791  100
```

비정형 데이터 처리

8. 단어 빈도수 구하기

```
# 단어 빈도수 구하기 및 내림차순 정렬
wordv <- sort(rowSums(mat), decreasing=TRUE) # 빈도수로 내림차순 정렬
wordv[1:5] # 상위 5개 단어 빈도수 보기
#연구 제품 결과 영향 기업
# 303  230  172  152  139
```

9. wordcloud 생성(디자인 전)

```
myName <- names(wordv) # 단어 이름 생성 -> 빈도수의 이름
wordcloud(myName, wordv) # 단어구름 적
myName


x11( ) # 별도의 창을 띄우는 함수
```

비정형 데이터 처리

- Wordcloud 생성(디자인 전)

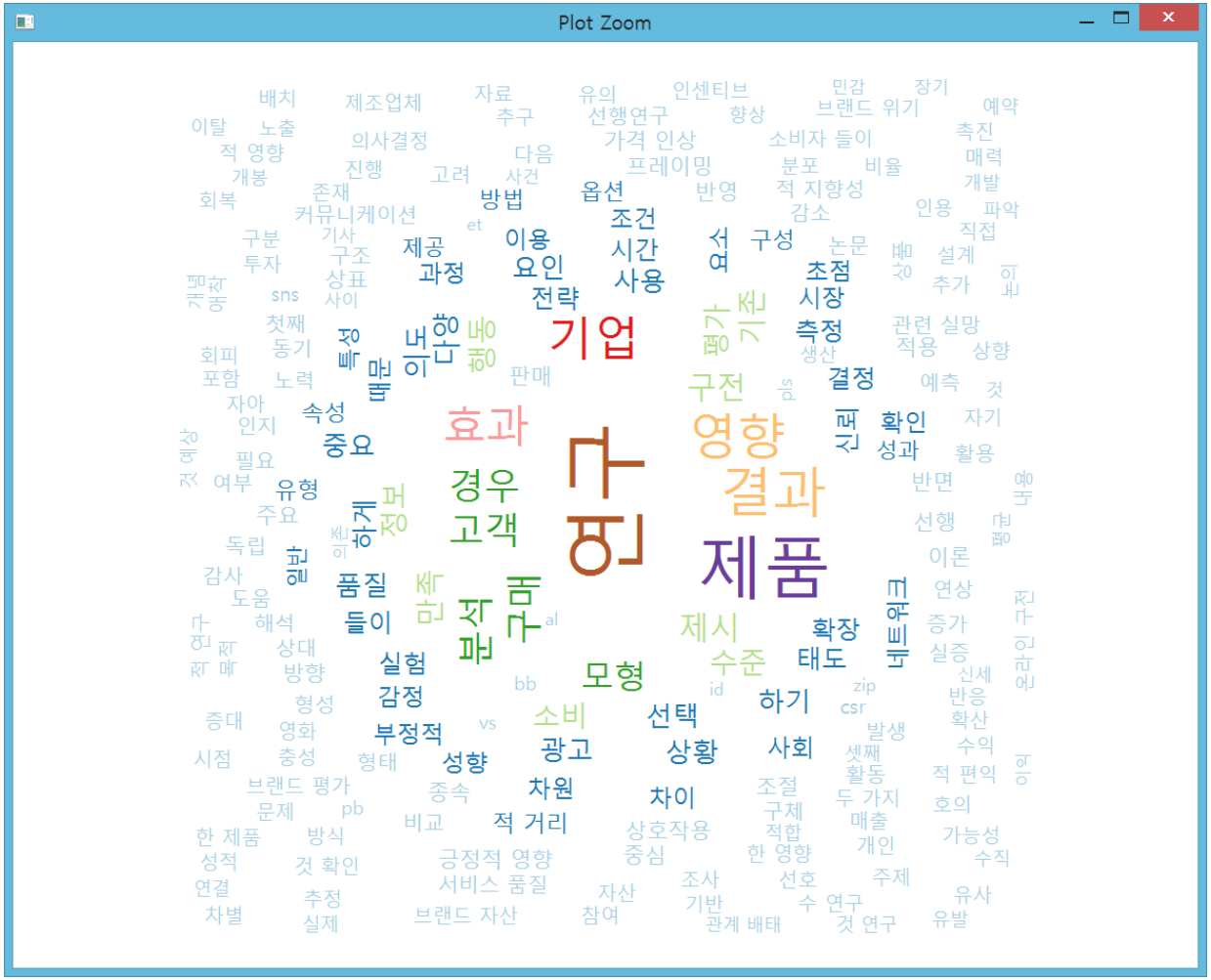
비정형 데이터 처리

9. 단어구름에 디자인 적용(빈도수, 색상, 랜덤, 회전 등)

```
# 단어이름과 빈도수로 data.frame 생성
d <- data.frame(word=myName, freq=wordv)
str(d) # word, freq 변수

# 색상 지정
pal <- brewer.pal(12,"Paired") # Set1~Set3

# 폰트 설정세팅 : "맑은 고딕", "서울남산체 B"
windowsFonts(malgun=windowsFont("맑은 고딕"))  #windows

# 색상, 빈도수, 글꼴, 회전 등 적용
wordcloud(d$word, d$freq, scale=c(5,1), min.freq=3, random.order=F, rot.per=.1,
colors=pal, family="malgun")
#wordcloud(단어, 빈도수, 5:1비율 크기,최소빈도수,랜덤순서,회전비율, 색상(파렛트),컬러,글꼴 )
```

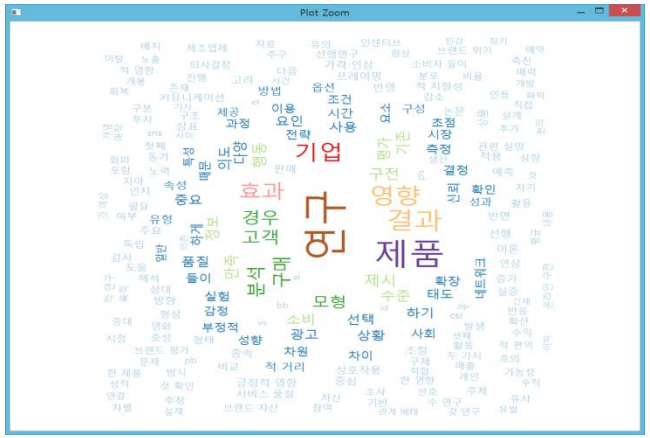
비정형 데이터 처리

- Wordcloud 생성(디자인 후)

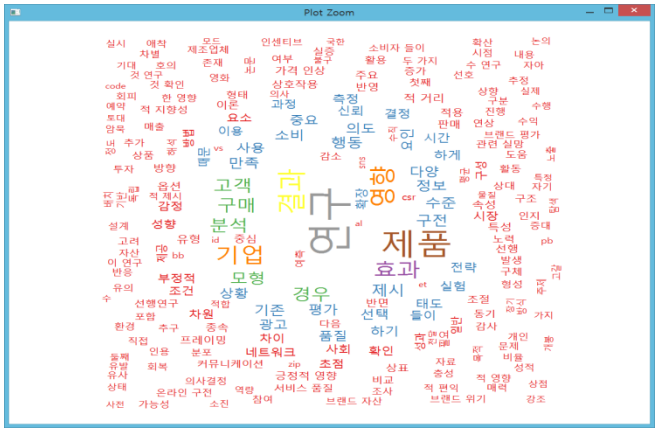


비정형 데이터 처리

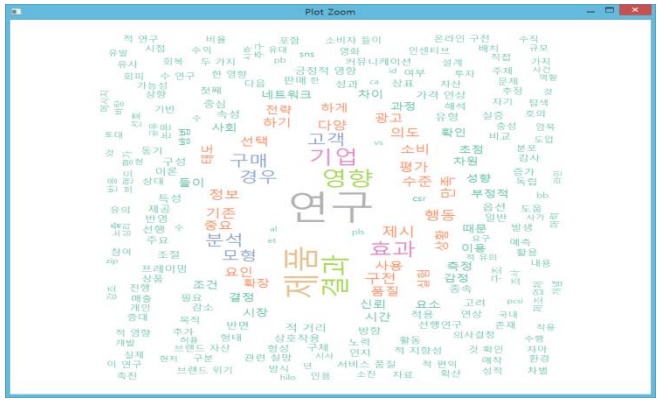
- brewer.pal() 색상 지정



pal <- brewer.pal(12,"Paired")



pal <- brewer.pal(12,"Set1")



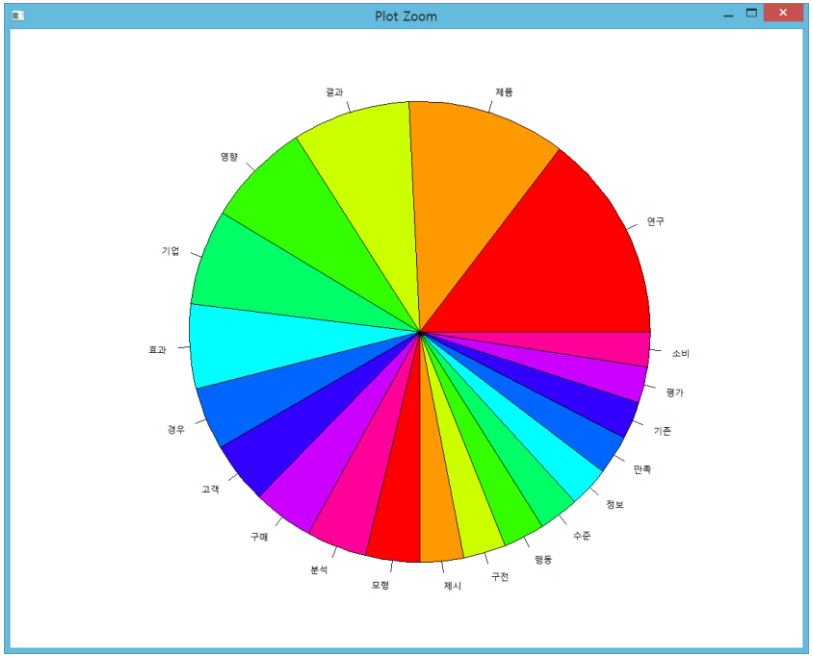
pal <- brewer.pal(12,"Set2")

pal <- brewer.pal(12,"Set3")

비정형 데이터 처리

10. 차트 시각화

```
word <- head(sort(wordv, decreasing=T), 20) # 상위 20개 토픽추출
pie(word, col=rainbow(10), radius=1) # 파이 차트 - 무지개색, 원크기
pct <- round(word/sum(word)*100, 1) # 백분율
names(word)
# 키워드와 백분율 적용
lab <- paste(names(word), "₩n", pct, "%")
```

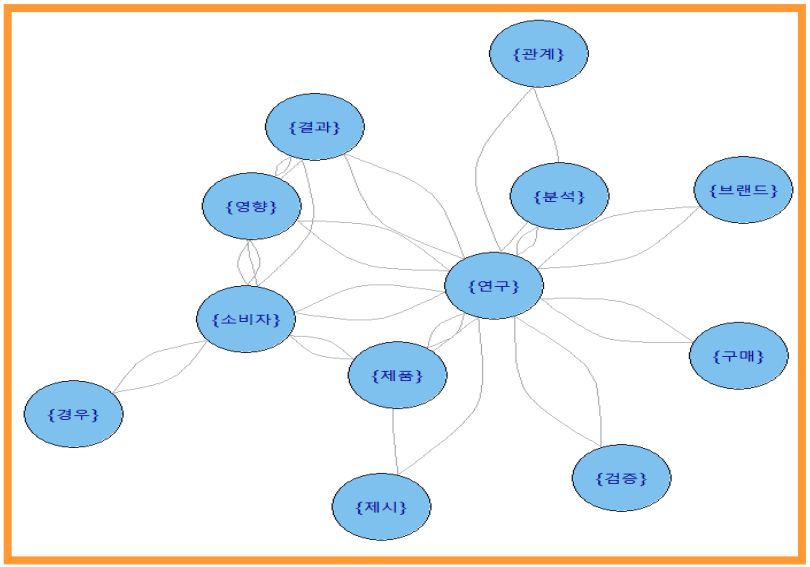


비정형 데이터 처리

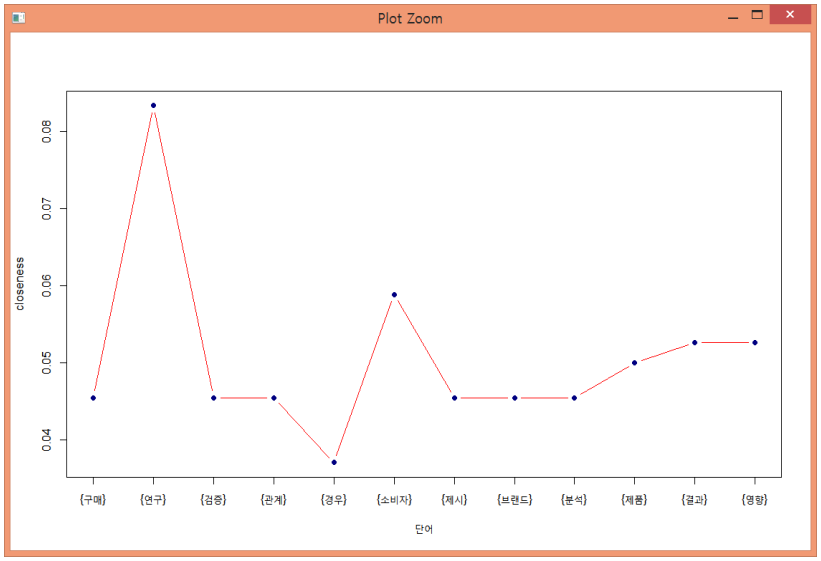
```
##############################################
#    단계2 – 연관어 분석(단어 연관성)                    #
#          √ 시각화 : 연관어 네트워크 시각화와 근접중심성   #
##############################################
```

비정형 데이터 처리

단계2 : 연관어 분석(결과물)



연관어 분석 시각화



연관어 중요도(중심성) 시각화

비정형 데이터 처리

- 연관어 분석을 위한 패키지 설치

```
install.packages("rJava")
Sys.setenv(JAVA_HOME='C:/Program Files/Java/jre1.8.0_161')
library(rJava) # 아래와 같은 Error 발생 시 Sys.setenv()함수로 java 경로 지정

install.packages(c("KoNLP", "arules", "igraph"))
library(KoNLP) # rJava 라이브러리가 필요함

library(arules) # 연관규칙 라이브러리
library(igraph)
```

비정형 데이터 처리

1. 데이터셋(abstract.txt) 가져오기

```
f <- file("C:/workspaces/Rwork/data/abstractClean.txt", encoding="UTF-8")
fl <- readLines(f)
# incomplete final line found on - Error 발생 시 UTF-8 인코딩 방식으로 재 저장
close(f)
head(fl, 10)
```

[1] "타인의 도움을 받은 사람은 받은 호의나 도움에 대해 감사를 느끼기도 하지만 빚을 졌다는 감정과 이에 보답을 해야 한다는 의무감을 느낀다. ”

[2] "그리고 감사는 받은 도움에 대한 고마움과 이에 대한 호의의 차원에서 친사회적 결과를, 신세는 받은 혜택을 되갚아야 된다는 의무감에 의해 친사회적 행동을 보이게 된다. ”

:

생략

비정형 데이터 처리

2. 단어추출 및 단어 트랜잭션 생성

```
tran <- Map(extractNoun, fl) # 단어 추출 - KoNLP 제공 함수
tran <- unique(tran) # 중복제거
# Warning messages:
tran <- sapply(tran, unique) # 중복제거
# 데이터 전처리
tran <- sapply(tran, function(x) {Filter(function(y) + {nchar(y) <= 4 &&
nchar(y) > 1 && is.hangul(y)},x)} )
tran <- Filter(function(x){length(x) >= 2}, tran) # 2자 이상 단어 필터링
names(tran) <- paste("Tr", 1:length(tran), sep="") # 앞쪽에 Tr 문자열 붙임
names(tran)
wordtran <- as(tran, "transactions")
wordtran
wordtab <- crossTable(wordtran) # 교차표 작성
wordtab
```

비정형 데이터 처리

3. 단어간 연관규칙 산출

```
ares <- apriori(wordtran, parameter=list(supp=0.07, conf=0.05))
inspect(ares)
rules <- labels(ares, ruleSep=" ")
rules <- sapply(rules, strsplit, " ",  USE.NAMES=F)
rulemat <- do.call("rbind", rules)
rulemat
```

비정형 데이터 처리

- 연관규칙에 의한 연관어 결과

```
      [,1]       [,2]
 [1,] "{}"       "{사회}"
 [2,] "{}"       "{상호작용}"
 [3,] "{}"       "{감정}"
 [4,] "{}"       "{선택}"
 [5,] "{}"       "{하기}"
 [6,] "{}"       "{지각}"
 [7,] "{}"       "{고객}"
 [8,] "{}"       "{정보}"
 [9,] "{}"       "{확인}"
[10,] "{}"       "{이용}"
[11,] "{}"       "{만족}"
[12,] "{}"       "{특성}"
[13,] "{}"       "{시사점}"
[14,] "{}"       "{기존}"
[15,] "{}"       "{요인}"
[16,] "{}"       "{다양}"
[17,] "{}"       "{행동}"
[18,] "{}"       "{들이}"
[19,] "{}"       "{수준}"
[20,] "{}"       "{구매}"
[21,] "{}"       "{검증}"
[22,] "{}"       "{관계}"
```

```
[23,] "{}"       "{기업}"
[24,] "{}"       "{효과}"
[25,] "{}"       "{경우}"
[26,] "{}"       "{제시}"
[27,] "{}"       "{브랜드}"
[28,] "{}"       "{분석}"
[29,] "{}"       "{제품}"
[30,] "{}"       "{결과}"
[31,] "{}"       "{영향}"
[32,] "{}"       "{소비자}"
[33,] "{}"       "{연구}"
[34,] "{구매}"   "{연구}"
[35,] "{연구}"   "{구매}"
[36,] "{검증}"   "{연구}"
[37,] "{연구}"   "{검증}"
[38,] "{관계}"   "{연구}"
[39,] "{연구}"   "{관계}"
[40,] "{경우}"   "{소비자}"
[41,] "{소비자}" "{경우}"
[42,] "{제시}"   "{연구}"
[43,] "{연구}"   "{제시}"
[44,] "{브랜드}" "{연구}"
[45,] "{연구}"   "{브랜드}"
```

```
[46,] "{분석}"   "{연구}"
[47,] "{연구}"   "{분석}"
[48,] "{제품}"   "{소비자}"
[49,] "{소비자}" "{제품}"
[50,] "{제품}"   "{연구
[50,] "{제품}"   "{연구}"
[51,] "{연구}"   "{제품}"
[52,] "{결과}"   "{영향}"
[53,] "{영향}"   "{결과}"
[54,] "{결과}"   "{소비자}"
[55,] "{소비자}" "{결과}"
[56,] "{결과}"   "{연구}"
[57,] "{연구}"   "{결과}"
[58,] "{영향}"   "{소비자}"
[59,] "{소비자}" "{영향}"
[60,] "{영향}"   "{연구}"
[61,] "{연구}"   "{영향}"
[62,] "{소비자}" "{연구}"
[63,] "{연구}"   "{소비자}"
```

비정형 데이터 처리

4. 연관어 시각화

```
# 연관어 시각화 - rulemat[c(34:63),] # 연관규칙 결과- {} 제거(1~33)
ruleg <- graph.edgelist(rulemat[c(34:63),], directed=F)
ruleg
plot.igraph(ruleg, vertex.label=V(ruleg)$name,vertex.label.cex=1.2,
            vertex.size=30,layout=layout.fruchterman.reingold.grid)
```

정점(타원) 크기 속성 : vertex.label.cex
레이블 크기, vertext.size

[연구] 단어 중심 네트워크 형성