15.4 SNS 연관어 분석

SNS을 대표하는 Twitter나 Facebook에서 검색한 데이터를 텍스트 자료로 읽어서 단어의 연관성을 분석하는 방법에 대해서 알아본다. '8.3 비정형 데이터 처리'에서는 Facebook에서 검색한 데이터를 텍스트 파일로 미리 저장해서 연관어 분석을 수행했지만, 이절에서는 직접 Twitter에서 앱을 등록하고, 인증과정을 통해서 접근할 수 있는 PIN을 획득하여 데이터를 가져오는 방법으로 연관어 분석을 수행한다. Twitter에서 검색된 데이터를 가져오기(crawling) 위한 절차는 다음과 같다.

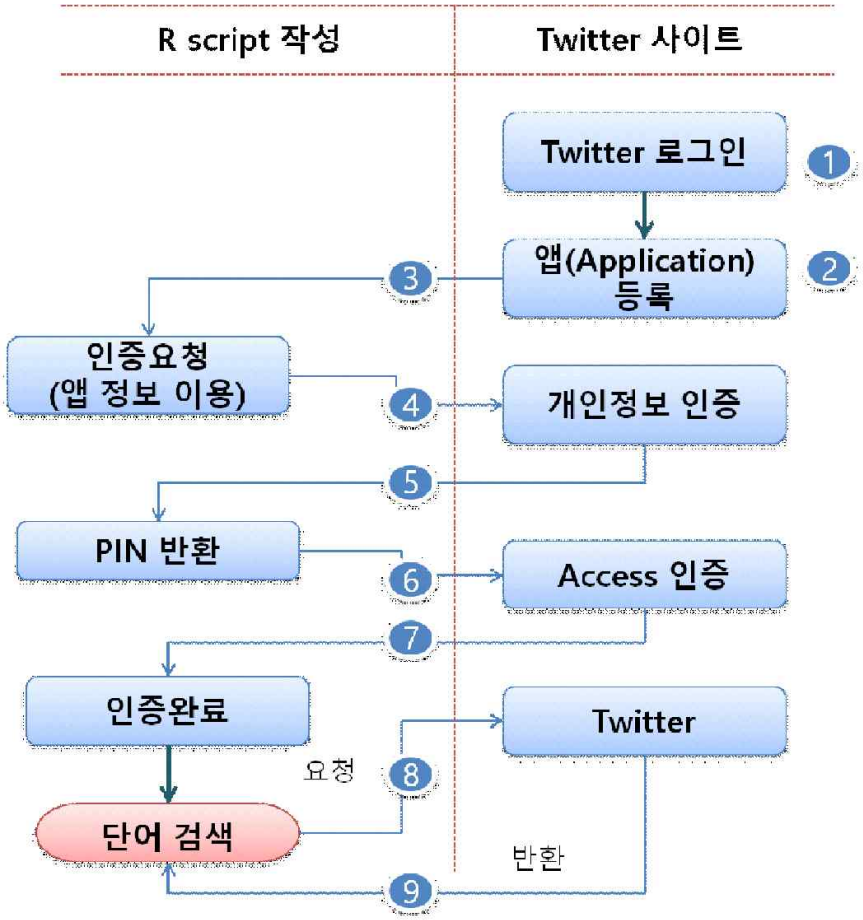


그림 15-4. Twitter에서 검색된 데이터 가져오기 절차

15.4.1 Twitter 앱 등록

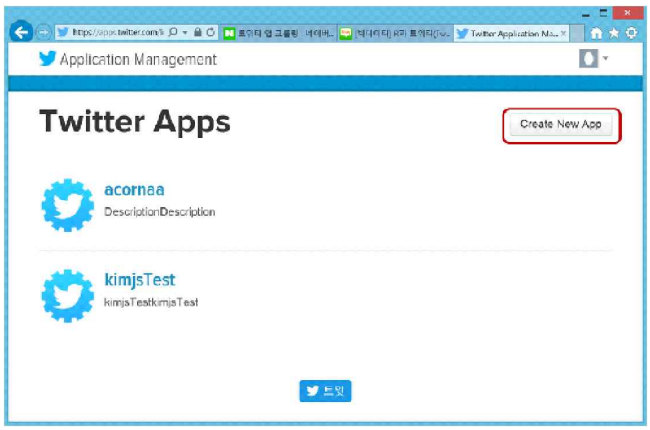
Twitter 공식 사이트에서 회원 가입 후 로그인 하면 Twitter 앱(Application)을 만들 수 있는 사이트에서 앱을 등록할 수 있다. 이 절의 내용은 그림 15-4에서 ① ~ ② 단계에 해당된다.

단계 1. Twitter 로그인

- https://twitter.com에서 회원 가입 후 email과 암호를 입력하여 로그인 한다.

단계 2. Twitter 앱 만들기

- https://apps.twitter.com/ko에서 [Create New App] 버튼을 클릭한다.

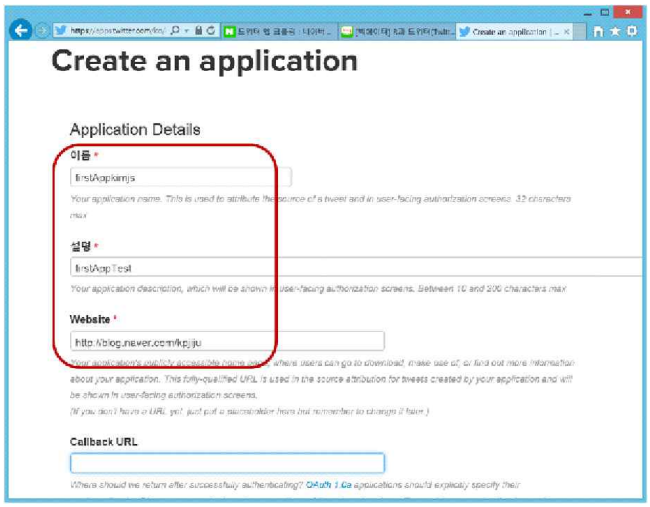


- 이름, 설명, 웹 사이트를 순서대로 입력한다.

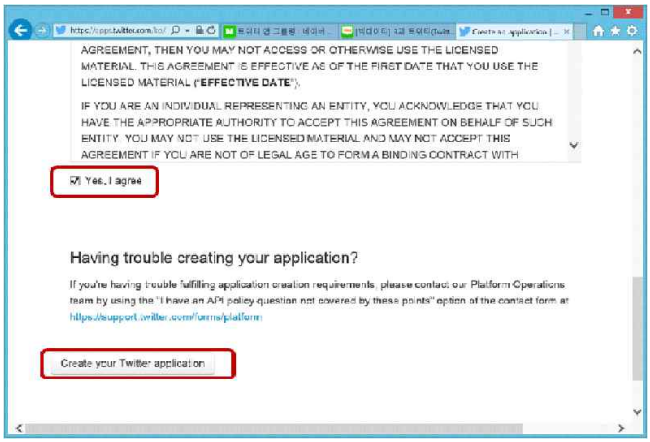
✓ 이름 : 앱의 이름으로 다른 사람이 이미 사용 중인 이름을 지정할 수 없다.

✓ 설명 : 앱에 대한 설명문으로 10자 이상 입력할 수 있다.

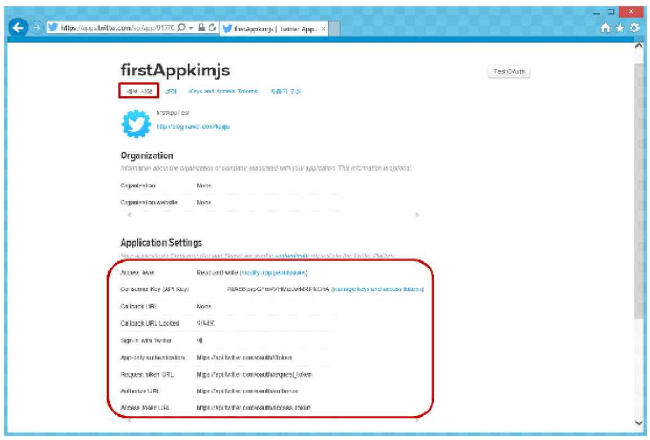
✓ 웹 사이트는 자신이 운영 중인 웹 사이트나 블로그 주소를 입력한다.



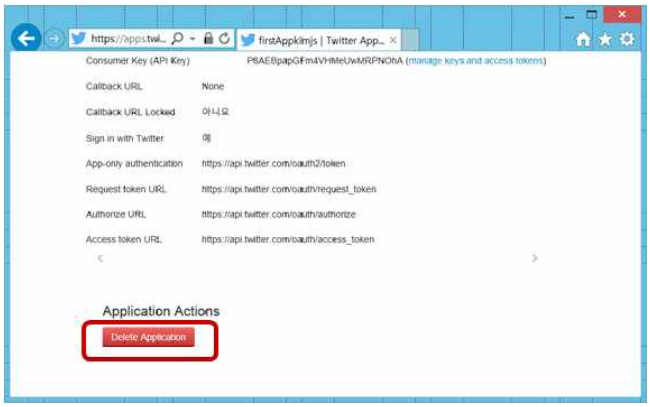
- 페이지 아래쪽으로 이동하여 동의 부분을 체크하고, [Create your Twitter Application] 버튼을 클릭하여 앱을 생성한다.



단계 3. Twitter 앱 세부사항

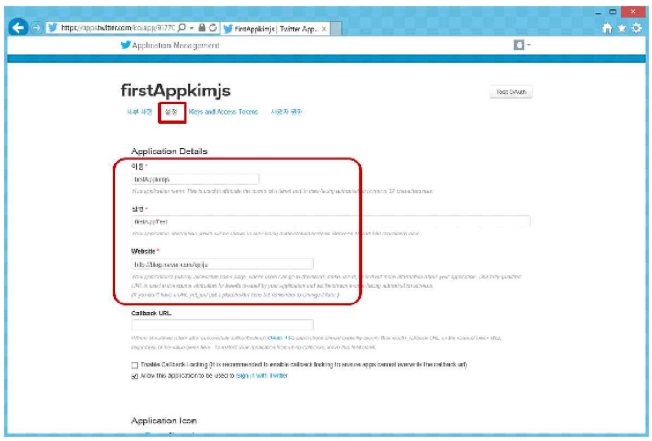


- 생성된 Twitter의 앱에 관한 세부사항을 볼 수 있는 탭이다.
- Request token URL, Authorize URL, Access token URL은 앱 인증 요청 시 사용된다.



- 페이지 아래쪽에 [Delete Application] 버튼을 클릭하면 해당 앱을 삭제할 수 있다.

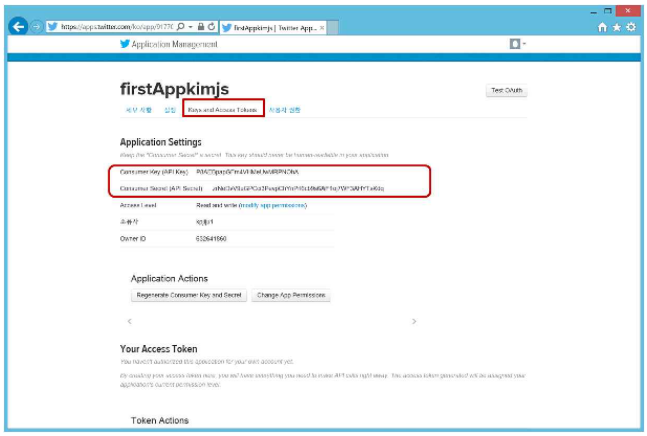
단계 4. Twitter 앱 설정



- 앱을 생성하는 과정에서 입력한 이름, 설명, 웹 사이트를 확인할 수 있는 탭이다.

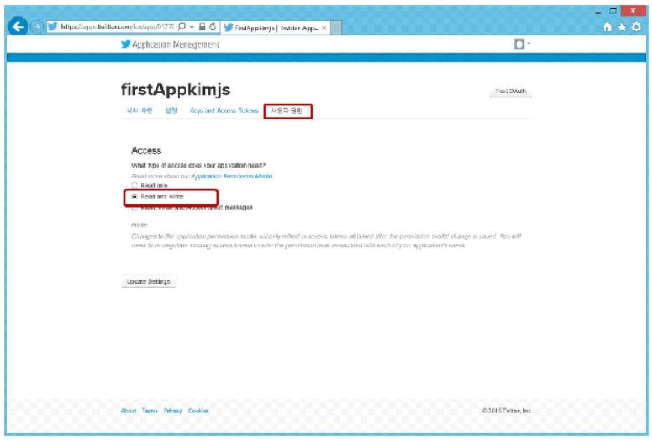
단계 5. Twitter 앱 키와 접근 토큰

- Twitter 인증 수행과정에서 필요한 키와 접근할 토큰을 생성하는 탭이다.



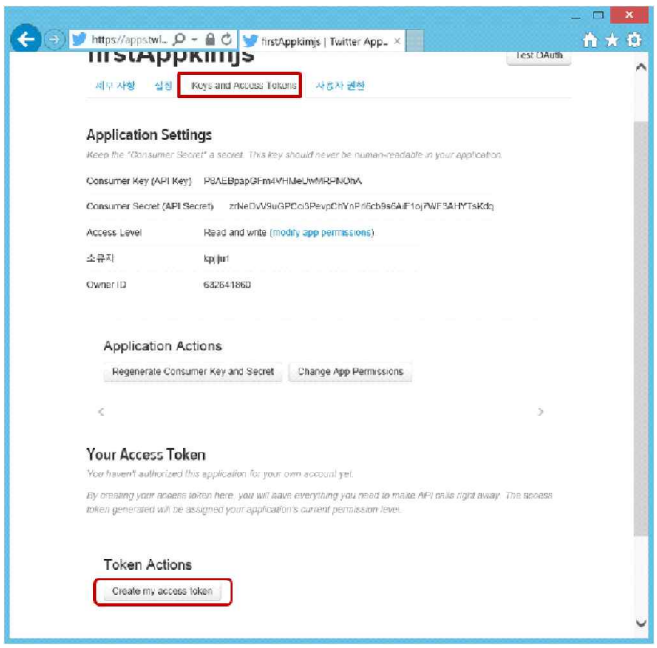
단계 6. Twitter 앱 사용자 권한

- Twitter 앱에 관한 사용자의 권한 설정을 확인하거나 수정할 수 있는 탭이다.
- Read and Write 사용자 권한 설정을 권장사항이다.

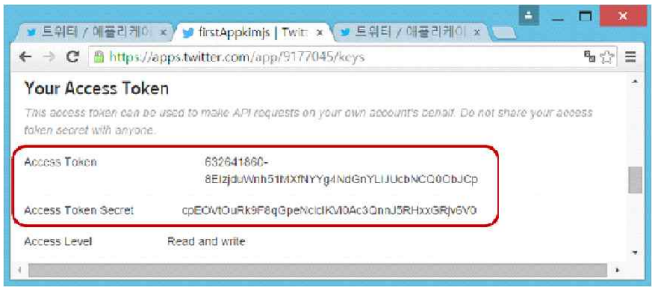


단계 7. Twitter 앱 토큰 생성

① [Keys and Access Tokens] 탭에서 [Create my access token] 버튼을 클릭하여 접근 토큰을 생성한다.



② Access Token과 Access Token Secret가 생성된다.



15.4.2 Twitter 앱 인증 요청 및 인증 완료

Twitter 공식 사이트에서 만들어진 앱의 URL과 Token 정보를 이용하여 R 스크립트를 작성하고, 작성된 스크립트를 이용하여 인증을 요청하면, Twitter 서버에서 PIN을 제공한다. PIN은 해당 웹으로 접근을 승인하는 역할을 제공한다. 이 절의 내용은 그림 15-4에서 ③ ~ ⑦ 단계에 해당된다.

단계 1. Twitter 앱 인증 요청 및 인증 수행을 위한 패키지 설치 및 로딩

```
install.packages("twitteR") # twitter 인증관련 함수 제공
install.packages("ROAuth") # OAuthFactory 객체 제공
library(twitteR)
library(ROAuth)
```

단계 2. Twitter 앱 요청 URL과 API 설정

```
# Twitter 앱 [세부사항] 탭을 참고하여 3개 URL 변수 생성
reqURL <- "https://api.twitter.com/oauth/request_token"
authURL <- "https://api.twitter.com/oauth/authorize"
accessURL <- "https://api.twitter.com/oauth/access_token"

# Twitter 앱 [Keys and Access Tokens] 탭을 참고하여 4개 변수 생성
apiKey <-  "P8AEBpapGFm4VHMeUwMRPNOhA"
apiSecret <- "zrNeDvV9uGPCci3PevpChYnPrl6cb9s6AiF1oj7WF3AHYTsKdq"
accessToken="632641860-8EIzjduWnh51MXfNYYg4NdGnYLIJUcbNCQ0ObJCp"
accessTokenSecret="cpEOVtOuRk9F8qGpeNclcIKVi0Ac3QnnJ5RHxxGRjv6V0"
```

단계 3. Twitter 앱 인증 요청

```
# API key, API Secret, URL 정보를 이용하여 Twitter 접근 객체 생성
twitCred <- OAuthFactory$new(
  consumerKey = apiKey,
  consumerSecret = apiSecret,
  requestURL = reqURL,
```

```
    accessURL = accessURL,
    authURL = authURL
  )
```

단계 4. Twitter 앱 인증 수행

인증 요청 결과를 이용하여 Twitter에 접근할 수 있는 PIN을 받기 위한 과정이다. RCurl 패키지는 HTTP/FTP, SSL/HTTPS 프로토콜을 기반으로 요청과 응답 형식으로 인증을 수행하는 패키지이다.

```
  twitCred$handshake(
    cainfo = system.file("CurlSSL", "cacert.pem", package = "RCurl")
  )
```

단계 5. Twitter 앱 인증을 위한 PIN 받기

단계 4의 수행 결과 Twitter 서버에서 에서 앱 접근을 승인하기 위한 PIN을 제공한다. PIN(Personal Identification Number)[1]은 앱 사용자를 식별해주는 역할을 제공한다.

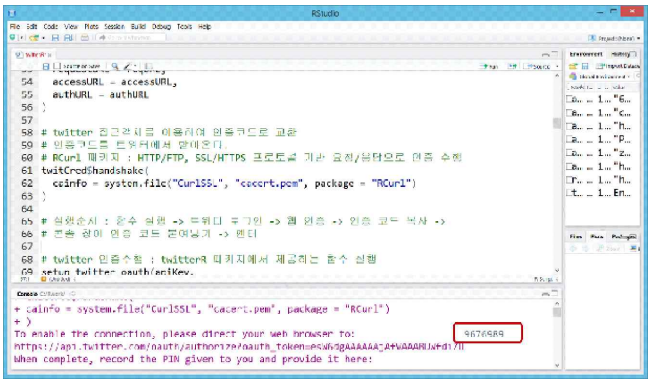
- Twitter 계정의 로그인 상태가 아니면 로그인 과정을 거친다.

- 만약 이미 로그인 상태이면 앱의 계정 사용을 위한 승인을 위해서 [앱 인증] 버튼을 클릭한다.

1) 인터넷상에서 제공하는 개인 식별 번호

▪ 앱 접근을 승인하기 위한 PIN을 제공한다.

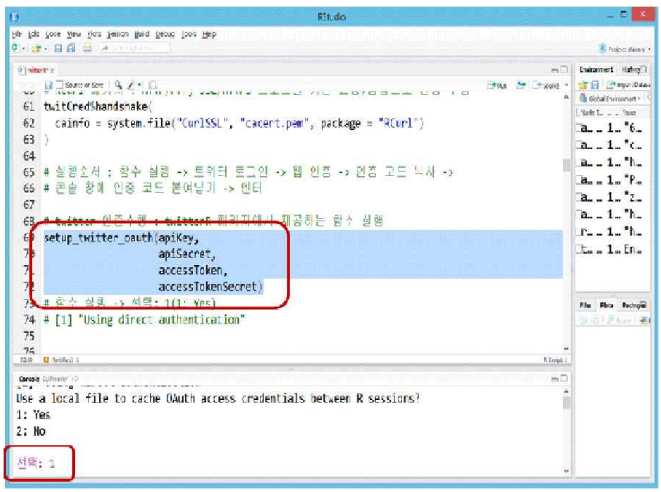
▪ R Studio의 콘솔 창에 PIN 번호를 입력한 후 Enter↵ 키를 눌러서 앱 접근을 위한 승인을 받는다.



① API와 Access Token 정보를 인수로 setup_twitter_oauth()함수를 작성하여 Twitter 인증을 수행한다.

```
setup_twitter_oauth(apiKey,
                    apiSecret,
                    accessToken,
                    accessTokenSecret)
```

② 선택 : 1 을 입력한 Enter↵ 키를 누른다.



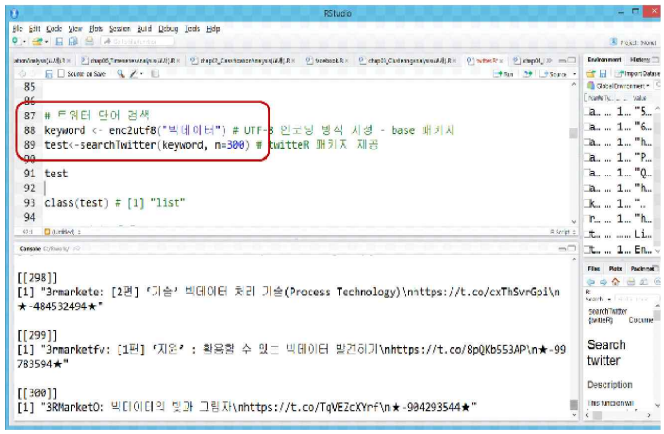
※ 위 과정까지 완료되면 Twitter 앱에 접근하여 데이터를 검색할 수 있는 인증 과정이 모두 완료된다.

15.4.3 Twitter 앱에 접근하여 데이터 가져오기

Twitter 앱에 접근할 수 있는 승인을 받았다면 이제 부터는 Twitter 앱을 통해서 데이터를 검색하고 검색 결과를 받을 수 있다. 이 절의 내용은 그림 15-4에서 ⑧ ~ ⑨ 단계에 해당된다.

단계 1. 검색어 입력과 검색결과 받기

▪ 다음과 같이 R 스크립트 작성하여 해당 검색어(“빅 데이터)를 이용하여 Twitter의 정보 검색 결과를 받는다.




```
keyword <- enc2utf8("빅 데이터")
test <- searchTwitter
        (keyword, n=300) #300개
```

※ 위 실행 결과는 “빅 데이터”를 검색어로 하여 최대 300개 까지 데이터를 가져와서 test 변수에 저장한다.

단계 2. list 자료구조를 vector 자료구조로 변경

Twitter에서 가져온 데이터는 list구조로 되어 있기 때문에 파일에 보관하기 위해서 vector형으로 변환해야 한다.

```
# vector 객체 생성
test_vec <- vector()
# list 객체 전체의 text 값을 가져와서 벡터 객체에 저장
n <- 1: length(test)
for(i in n){
  test_vec[i] <- test[[i]]$getText()
}
# 자료구조 보기
class(test_vec); str(test_vec)
```

단계 3. 파일 저장 및 가져오기

vector 자료구조의 데이터를 파일로 저장한다.

```
write.csv(test_vec, "c:/Rwork/output/test.txt", quote = F, row.names = F)

# 저장된 파일을 가져와서 줄 단위로 읽어온다.
test_txt <- file("C:/Rwork/output/test.txt")
twitter <- readLines(test_txt)
str(twitter)
```

15.4.4 Twitter 검색 데이터 대상 연관어 분석

Twitter를 통해서 가져온 데이터를 대상으로 연관어를 분석하여 연관 네트워크 형태로 시각화한다. 이 절의 수행 과정은 8.3.2 연관어 분석의 내용과 동일하다.

단계 1. 한글 처리를 위한 패키지 설치

```
install.packages("rJava")
Sys.setenv(JAVA_HOME='C:\\Program Files\\Java\\jre1.8.0_60')
library(rJava) # 아래와 같은 Error 발생 시 Sys.setenv()함수로 java 경로 지정
install.packages("KoNLP")
library(KoNLP) # rJava 라이브러리가 필요함
```

단계 2. 줄 단위 단어 추출

```
lword <- Map(extractNoun, twitter)
length(lword)
lword <- unique(lword) # 중복제거1(전체 대상)
length(lword)
lword <- sapply(lword, unique) # 중복제거2(줄 단위 대상)
length(lword)
lword # 추출 단어 확인
```

단계 3. 데이터 전처리

```
(1) 길이가 2~4 사이의 단어 필터링 함수 정의
filter1 <- function(x){
  nchar(x) <= 4 && nchar(x) >= 2 && is.hangul(x)
}
(2) Filter(f,x) -> filter1() 함수를 적용하여 x 벡터 단위 필터링
filter2 <- function(x){
  Filter(filter1, x)
}
(3) 줄 단어 대상 필터링
lword <- sapply(lword, filter2)
lword # 추출 단어 확인(길이 1개 단어 삭제됨)
```

단계 4. 트랜잭션 생성 : 연관분석을 위해서 단어를 트랜잭션으로 변환

```
install.packages("arules")
library(arules)
wordtran <- as(lword, "transactions") # lword에 중복데이터가 있으면 error발생
```

```
wordtran
# 트랜잭션 내용 보기 -> 각 트랜잭션의 단어 보기
inspect(wordtran)

# 동일한 단어끼리 교차테이블 작성
wordtable <- crossTable(wordtran) # 교차표 작성
length(wordtable)
str(wordtable)
```

단계 5.단어 간 연관규칙 산출

```
tranrules <- apriori(wordtran, parameter=list(supp=0.03, conf=0.8))
inspect(tranrules) # 연관규칙 54개 생성 결과 보기
```

단계 6. 연관어 시각화

```
(1) 데이터 구조 변경 : 행렬구조 변경
rules <- labels(tranrules, ruleSep=" ") # 연관규칙 레이블을 " "으로 변경
rules # 예) 59 {경영,마케팅}   => {자금} -> 59 "{경영,마케팅} {자금}"
class(rules)
[1] "character"

rules <- sapply(rules, strsplit, " ",  USE.NAMES=F)
rules
class(rules)
[1] "list"

# 행 단위로 묶어서 matrix로 반환
rulemat <- do.call("rbind", rules)
rulemat
class(rulemat)
[1] "matrix"

(2) 연관어 시각화를 위한 igraph 패키지 설치
install.packages("igraph")# graph.edgelist(), plot.igraph(), closeness() 함수 제공
library(igraph)

(3) edgelist보기 - 연관단어를 정점 형태의 목록 제공
ruleg <- graph.edgelist(rulemat[c(1:54),], directed=F) # [1,]~[11,] "{}" 제외
```

(4) edgelist 시각화

```
X11()
plot.igraph(ruleg, vertex.label=V(ruleg)$name,
            vertex.label.cex=1.0, vertex.label.color='black',
            vertex.size=20, vertex.color='green', vertex.frame.color='blue')
```

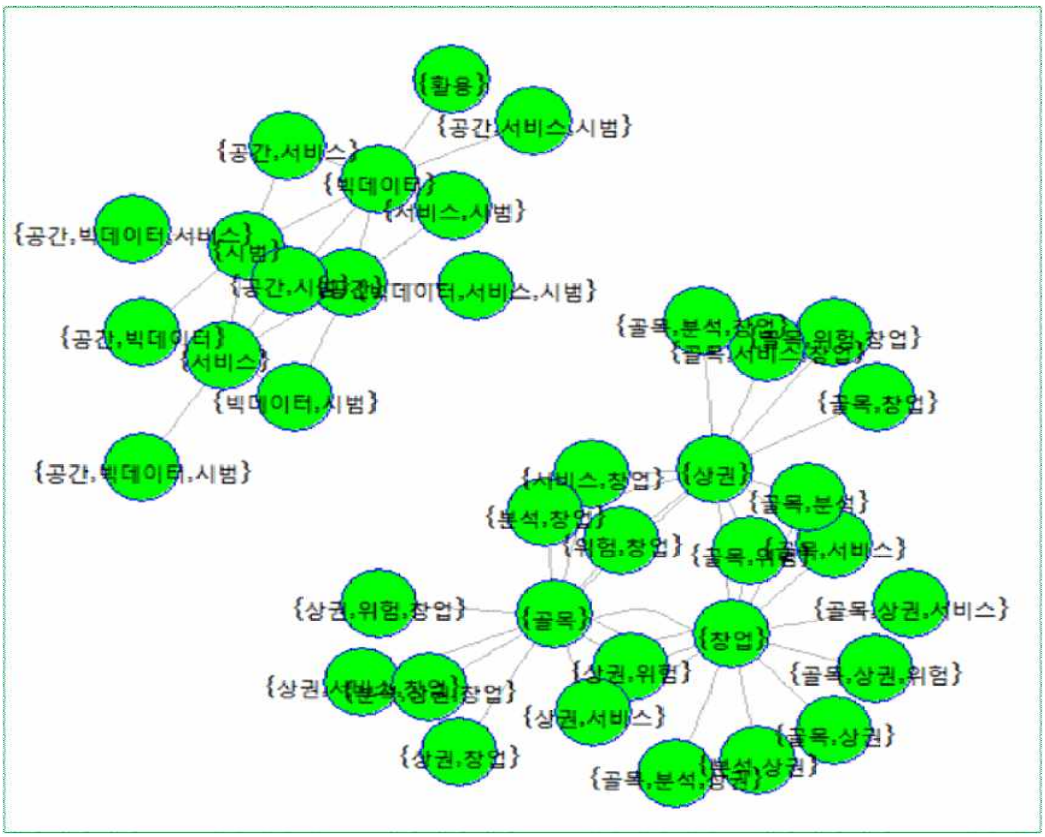


그림 15-5. Twitter 데이터 연관어 네트워크 시각화 결과

<해설> '빅 데이터' 검색어로 Twitter에서 가져온 텍스트 자료를 대상으로 지지도 3%, 신뢰도 80%(supp=0.03, conf=0.8)을 적용하여 발견된 54개의 규칙을 토대로 연관어를 분석하고, 결과를 연관어 네트워크 형태로 시각화한 결과이다. Twitter에서 가져온 데이터는 파일에 보관되어 있기 때문에 토픽분석을 통해서 단어의 빈도분석도 수행이 가능하다.