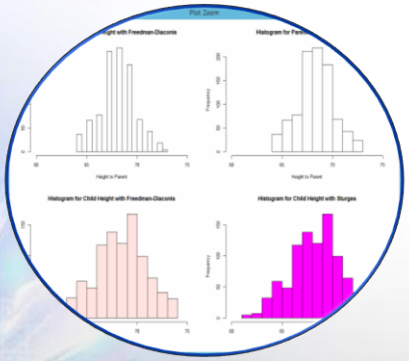
# Part-II. 데이터 특성 분석과 모델링



5. 함수(사용자정의, 내장함수)
6. 데이터 시각화
7. 데이터 전처리
8. 정형과 비정형 데이터 처리

# 5. 함수

## Chap05_Function 수업내용

- 사용자 정의함수 형식
  - ✓ 사용자가 정의한 함수

- R 내장함수
  - ✓ R 설치 시 제공하는 함수

# 5. 함수

- 사용자 정의함수 형식

(형식)

함수명 <- function(매개변수){ }

```
# 매개변수가 없는 함수 예
p <- function( ){
  cat("매개변수가 없는 함수")
}
p() # 함수 호출
```

# 5. 함수

- 매개변수가 있는 함수 예

```
p<- function(x){
   cat("x의 값 = ",x, "₩n") # ₩n 줄바꿈
   print(x) # 변수만 사용
}
p(5+10) # 함수 호출
```

# 5. 함수

- 파타고라스 정의 증명- 식 : a^2+b^2=c^2

```
pytha <- function(s,t){
  a <- s^2 - t^2
  b <- 2*s*t
  c <- s^2 + t^2
  cat("피타고라스의 정리 : 3개의 변수 : ",a,b,c)
}

pytha(2,1) # s,t는 양의 정수 -> 3 4 5
```

# 5. 함수

- # 구구단 출력하기

```
gugu <- function(i,j){
  for(x in i){
    cat("**", x , "단 **₩n")
    for(y in j){
      cat(x, "*", y, "=", x*y, "₩n")
    }
    cat("₩n")
  }
}
i<- c(2:9)
j<- c(1:9)
gugu(i,j)
```

# 5. 함수

<연습문제1> 날짜별 자판기의 코인 수가 2이상이면 "수입금 GOOD"을
그렇지 않으면 "수입금 Bad"를 출력하는 사용자 정의함수를
작성하시오.
조건) 사용 함수 : stringr 패키지 함수 :str_extract(), str_replace()
숫자변환 : as.numeric()

```
# dataset
income <- c("2015-02-05 income1coin", "2015-02-06 income2coin",
            "2015-02-07 income3coin“)

library(stringr) # 패키지 로드
vending <- function(x) {
  for(i in x){
    coin =  ?  # 동전수
    coin =  ?  # 문자열4개 제거
    coin =  ?  # 숫자 변경(연산가능)
    print(coin)
    # 조건
  }
}
vending(income)
```

# 5. 함수

<연습문제2> 함수 y = f(x)에서 x의 값이 a에서 b까지 변할 때

△x = b - a를 x의 증분이라고 한다.

△y = f(b) - f(a)를 y의 증분이라고 한다.

평균변화율 = △y/△x = f(b) - f(a)/ b-a

조건) 함수 $f(x) = x^3 + 4$에서 x의 값이 1에서 3까지 변할 때 평균변화율(mean ratio of change)을 구하는 함수를 작성하시오. (결과값 : mrc = 31-5 / 2 = 13)

# 5. 함수

- 기술통계량 처리 내장함수

```
min(vec) # 벡터 대상 최소값
max(vec) # 벡터 대상 최대값
range(vec) # 벡터 대상 범위 값
mean(vec) # 벡터 대상 평균값
median(vec) # 벡터 대상 사분위수
sum(vec) # 벡터 대상 합계
sort(x)  : 벡터 정렬 (단, 원래의 값을 바꾸지는 않음)
order(x)  : 벡터의 정렬된 값의 인덱스를 보여줌
rank(x)  : 벡터의 각 원소의 순위를 알려줌
sd(x) # 표준편차
summary(x)  : 데이터에 대한 기본적인 통계 정보 요약
table(x)  : 데이터 빈도수
```

# 5. 함수

- 수학관련 내장함수

  abs(x) # 절대값
  sqrt(x) # 제곱근
  ceiling(x), floor(), round() # 값의 올림, 내림, 반올림
  factorial(x) # 팩토리얼 함수
  which.min(x), which.max(x) # 벡터 내의 최소값과 최대값의 인덱스
  pmin(x), pmax(x) # 여러 벡터에서의 원소 단위 최소값과 최대값
  prod() # 벡터의 원소들의 곱
  cumsum(), cumprod() # 벡터의 원소들의 누적합과 누적곱
  cos(x), sin(x), tan(x)  # 삼각함수 (also acos(x), cosh(x), acosh(x), etc)
  log(x) # 자연로그(natural logarithm)
  log10(x) # 10을 밑으로 하는 일반로그 함수(e^x)

# 5. 함수

- 행렬연산 내장함수

ncol(x) # 열의 수
nrow(x) # 행의 수
t(x) # 전치행렬
cbind(...) # 열을 더할 때 이용되는 함수
rbind(...) # 행을 더할 때 이용되는 함수
diag(x) # 대각행렬
det(x) # 행렬식
apply(x, m, fun) # 행 또는 열에 함수 적용
x %*% y # 두 행렬의 곱
solve(x) # 역 행렬
svd(x) # Singular Value Decomposition
qr(x) # QR Decomposition (QR 분해)
eigen(x) # Eigenvalues(고유값)
chol(x) # choleski decomposition(Choleski 분해)

# 5. 함수

- 집합연산 내장함수

  union (x, y) # 집합 x와 y의 합집합

  intersect (x, y) # 집합 x와 y의 교집합

  setdiff (x, y) # x의 모든 원소 중 y에는 없는 x와 y의 차집합

  setequal (x, y) # x와 y의 동일성 테스트

  c %in% y # c가 집합 y의 원소인지 테스트

  choose (n, k) # 크기 n의 집합에서 크기 k의 가능한 부분 집합 개수

# 5. 함수

- 기초 통계량 구하기

```
getwd()
setwd("c:/Rwork/Part-I“)
#excel에서 csv(쉼표로 분리)형식으로 저장한 파일 가져오기
excel <- read.csv("excel.csv", header=TRUE)
# head()함수이용 앞쪽 10줄 출력
head(excel,10) # q1 q2 q3 q4 q5
#colMeans()함수 이용 각 열의 평균 계산
colMeans(excel[1:5])
#q1       q2       q3       q4       q5
#2.733831 2.907960 3.621891 2.509950 3.385572

# summary()함수 이용 각 열단위 기초 통계량
summary(excel)
```

# 6. 데이터 시각화

## chap06_DataVisualization 수업내용

1) 이산변수 시각화
   - ① 막대차트 시각화
   - ② 점 차트 시각화
   - ③ 파이 차트 시각화
2) 연속변수 시각화
   - ① 상자 그래프 시각화
   - ② 히스토그램 시각화
   - ③ 산점도 시각화
   - ④ 변수 간의 비교 시각화

# 6. 데이터 시각화

- **시각화를 위한 데이터 셋 가져오기**

```
install.packages("RSADBE")
library(RSADBE)
data(Severity_Counts) # RSADBE 패키지 제공 데이터셋
str(Severity_Counts) # Named num [1:10]
Severity_Counts # 버그 측정 데이터 셋
```

<<Severity_Counts 데이터 셋에 관한 설명>>

RSADBE 패키지에서 제공하며, 소프트웨어 발표 전과 후의 버그를 측정한 데이터 셋을 제공한다. Severity_Counts는 다음과 같은 변수로 구성되어 있다.

| Bugs.BR/AR | NT.BR/AR | Major.BR/AR | Critical.BR/AR | H.BR/AR |
|---|---|---|---|---|
| 단순버그 | 사소하지않음 | 중대한버그 | 결정적인 버그 | 시급한 버그 |

# 6. 데이터 시각화

## 1) 이산변수(discrete quantitative data) 시각화

- 정수단위로 나누어 측정할 수 있는 변수

- **barplot() 형식 – 막대차트 그리기 함수**

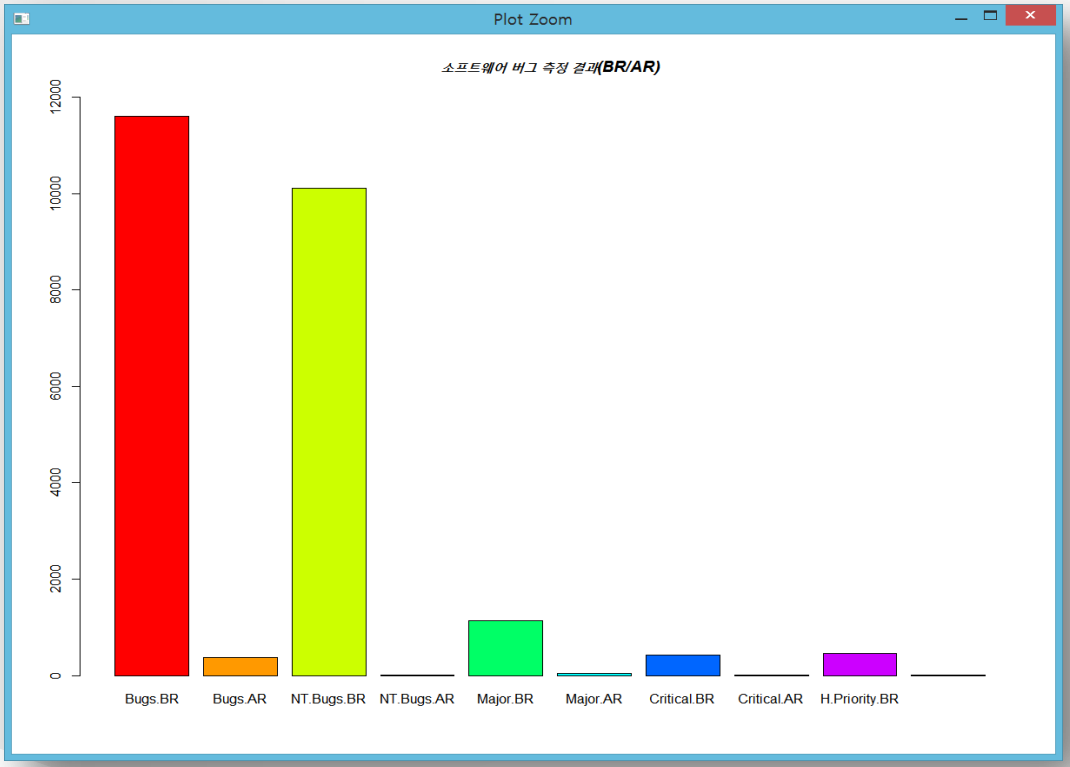
```
help("barplot") # barplot() 함수 형식 보기

barplot(height, width = 1, space = NULL,
        names.arg = NULL, legend.text = NULL, beside = FALSE,
        horiz = FALSE, density = NULL, angle = 45,
        col = NULL, border = par("fg"),
        main = NULL, sub = NULL, xlab = NULL, ylab = NULL,
        xlim = NULL, ylim = NULL, xpd = TRUE, log = "",
        axes = TRUE, axisnames = TRUE,
        cex.axis = par("cex.axis"), cex.names = par("cex.axis"),
        inside = TRUE, plot = TRUE, axis.lty = 0, offset = 0,
        add = FALSE, args.legend = NULL, ...)
```

# 6. 데이터 시각화

## ① 막대차트 시각화

# ylim=c(0,12000) : y축 값 범위, col=rainbow(10) : 10가지 무지개 색상,
# main : 제목, font.main=4 : 글꼴 유형

```
barplot(Severity_Counts, ylim=c(0,12000),
    col=rainbow(10), main ="소프트웨어 버그 측정 결과(BR/AR)",font.main=4)
```

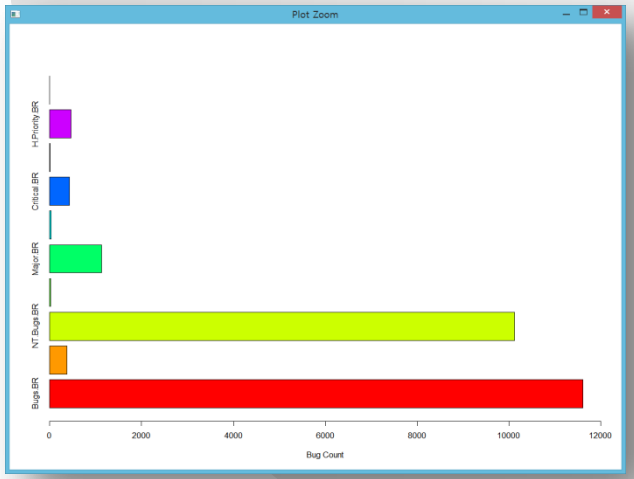


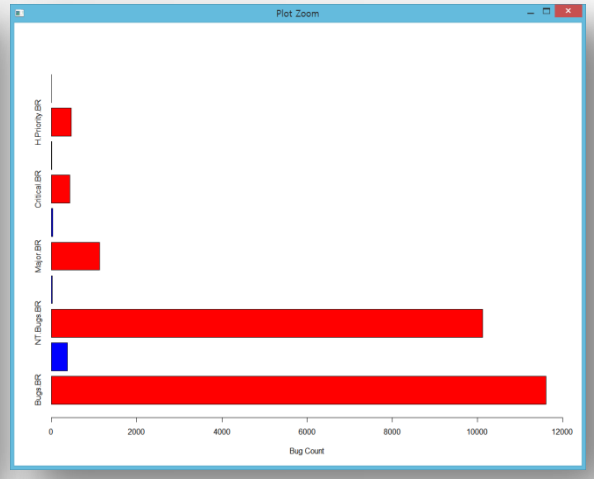
# 6. 데이터 시각화

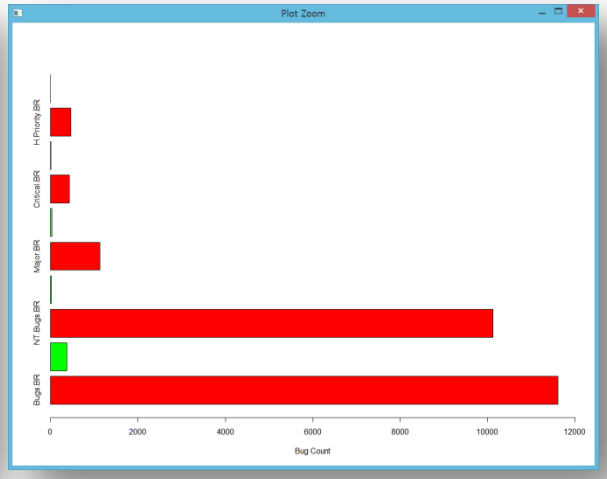
● **가로막대 차트 시각화**

# xlab : x축 이름, xlim : x축 값 범위, horiz=T : 가로막대

```
barplot(Severity_Counts,xlab="Bug Count", xlim=c(0,12000),
        horiz=T, col=rainbow(10)) # 10가지 무지개 색
barplot(Severity_Counts,xlab="Bug Count", xlim=c(0,12000),
        horiz=T, col=rep(c(2, 4),5)) # red와 blue 색상 5회 반복
barplot(Severity_Counts,xlab="Bug Count", xlim=c(0,12000),
        horiz=T, col=rep(c("red","green"),5)) # red와 green 색상 5회 반복
```



# 6. 데이터 시각화

- **1행 2열 그래프 보기**

**# 차트 데이터 가져오기**

**data(Bug_Metrics_Software)** # RSADBE 패키지 제공 데이터 셋

**Bug_Metrics_Software** # 행렬 데이터 구조 - 1면(Before)과 2면(After) 구성

<<Bug_Metrics_Software 데이터 셋에 관한 설명>>

RSADBE 패키지에서 제공하며, 5개의 소프트웨어 별로 발표 전과 후 버그 측정 결과를 3차원 배열구조로 데이터 셋을 제공한다. 1면에는 소프트웨어 발표 전(Before) 버그 측정 결과, 2면에는 소프트웨어 발표 후(After) 버그 측정 결과를 제공한다.

# 6. 데이터 시각화

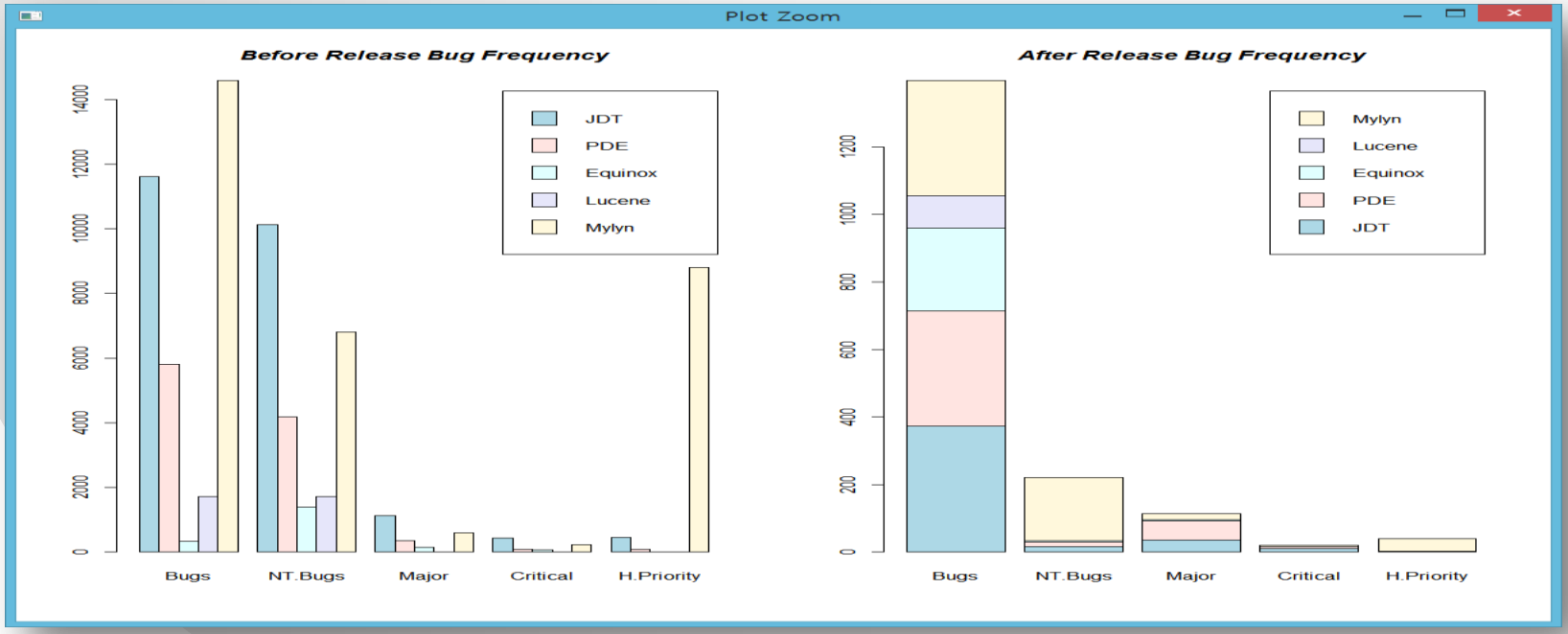
- **Before Bug(1면) 차트 그리기**

```
par(mfrow=c(1,2)) # 1행 2열 그래프 보기(2개 차트 동시에 보이기)
barplot(Bug_Metrics_Software[,,1], beside=T,
        col=c("lightblue","mistyrose","lightcyan","lavender","cornsilk"),
        legend=c("JDT","PDE","Equinox","Lucene","Mylyn"))
title(main ="Before Release Bug Frequency",font.main=4)

# beside = T : X축 값이 가로로 배열, F이면 하나의 막대로 누적
# col =c() : 5가지 막대색상
# legend : 범례 이름(S/W이름)
# title : 차트 제목, font.main=4 : 기울림체
```

# 6. 데이터 시각화

- After Bug(2면) 차트 그리기

```
barplot(Bug_Metrics_Software[,,2], beside=F,
        col=c("lightblue","mistyrose","lightcyan","lavender","cornsilk"),
        legend=c("JDT","PDE","Equinox","Lucene","Mylyn"))
title(main ="After Release Bug Frequency", font.main=4)
```

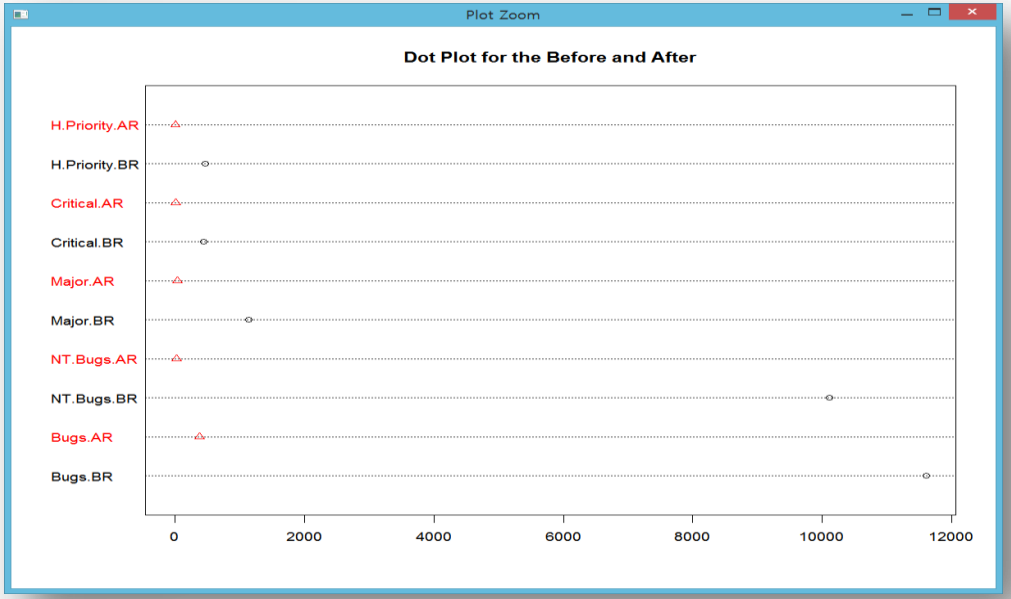



1행 2열 그래프 결과 화면

# 6. 데이터 시각화

## ② 점 차트 시각화

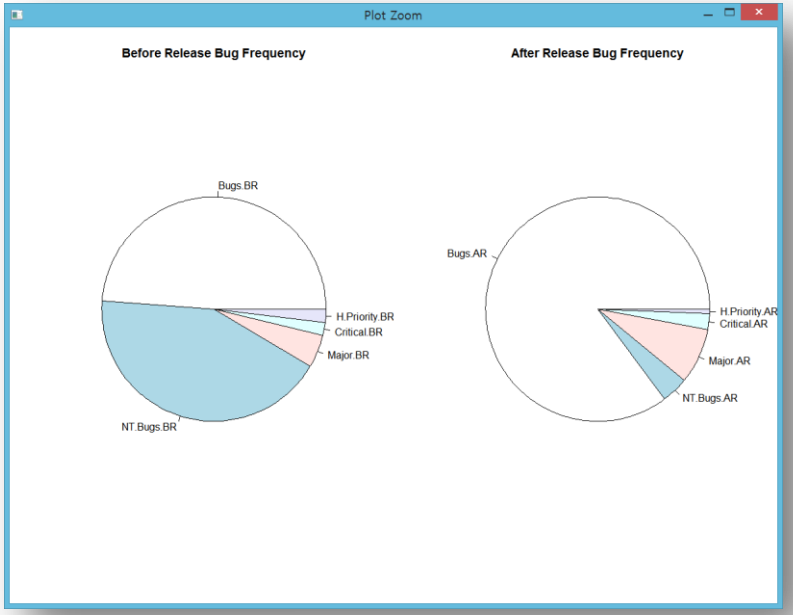
```
par(mfrow=c(1,1)) # 1행 1열 그래프 보기
dotchart(Severity_Counts, col=9:10, lcolor="black", pch=1:2,
         labels=names(Severity_Counts),
         main="Dot Plot for the Before and After", cex=1.2)
# col=9:10 -> BR(검정), AR(빨강)
# lcolor="black" -> 구분선(line) 검정색
# pch=1:2 -> 점 모양 : 원(1), 삼각형(2), +(3)
# labels=names(Severity_Counts) : y축 이름(컬럼명으로 지정)
# cex=1.2 -> 1.2배 확대
```



# 6. 데이터 시각화

## ③ 파이 차트 시각화

```
class(Severity_Counts) # "numeric"
Severity_Counts

par(mfrow=c(1,2)) # 1행 2열 그래프 보기
pie(Severity_Counts[c(1,3,5,7,9)]) # Bugs.BR
title("Before Release Bug Frequency")
pie(Severity_Counts[c(2,4,6,8,10)]) # Bugs.AR
title("After Release Bug Frequency")
```



# 6. 데이터 시각화

## 2) 연속변수(Continuous quantitative data)

- 시간, 길이 등과 같이 연속성을 가진 실수 단위 변수값

● 데이터 셋 가져오기

```
data(resistivity) # RSADBE패키지에서 제공하는 데이터셋
class(resistivity) # data.frame
resistivity # Process.1 Process.2
# Process.2 데이터는 Process.1 보다 0.004 정도 높은 값
```

# 6. 데이터 시각화

① 상자 그래프 그래프 시각화
- ✓ 상자 그래프는 요약정보를 시각화한다.
- ✓ 데이터의 퍼짐 정도와 이상치 발견이 목적

**summary(resistivity)**

#Process.1 Process.2

#Min. :0.1380 Min. :0.1420 맨하위 실선

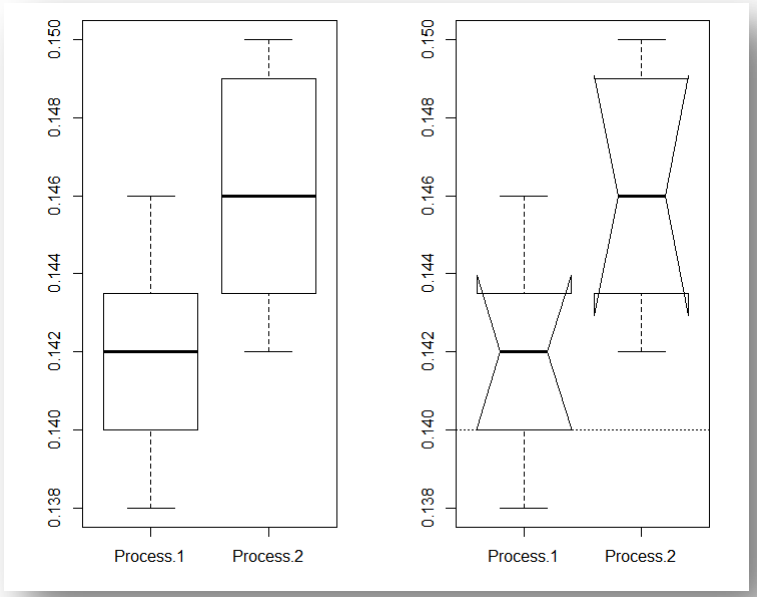
#1st Qu.:0.1405 1st Qu.:0.1437 박스 상단

#Median :0.1420 Median :0.1460 중간실선

#Mean :0.1419 Mean :0.1461

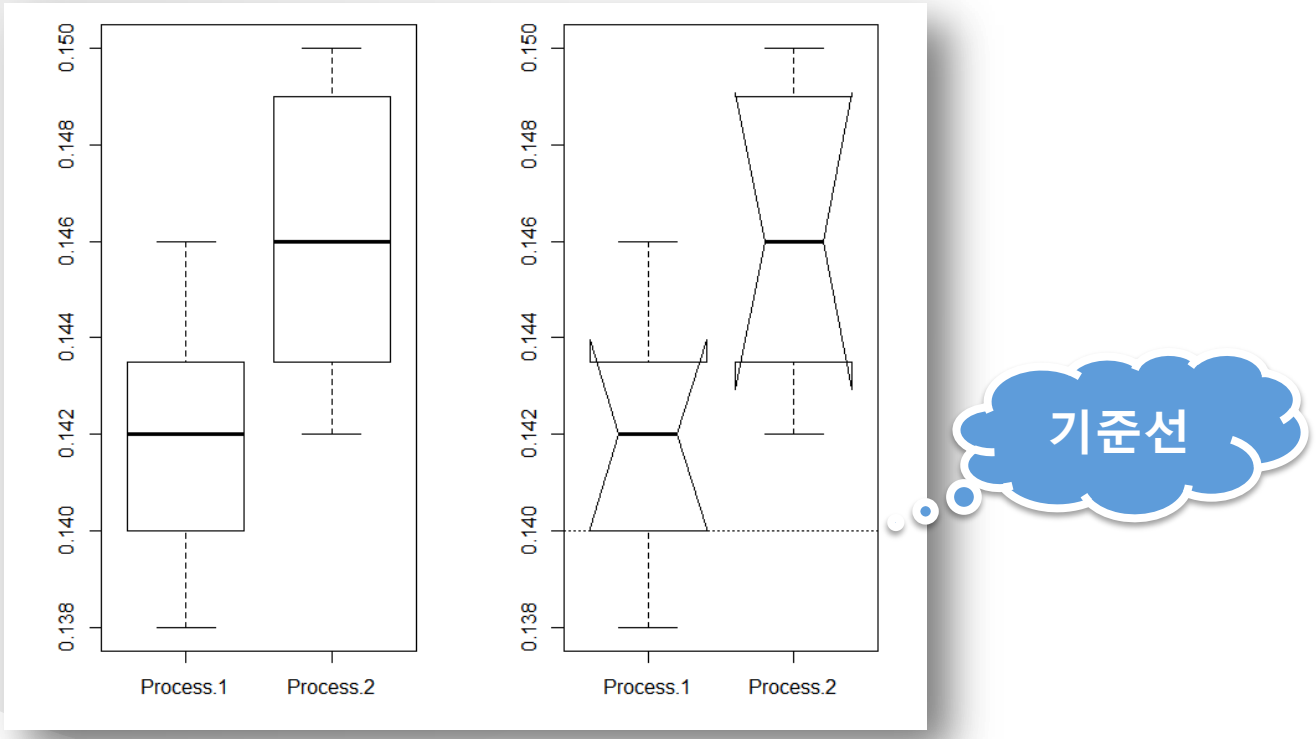
#3rd Qu.:0.1432 3rd Qu.:0.1485 박스 하단

#Max. :0.1460 Max. :0.1500 맨상위 실선



# 6. 데이터 시각화

**boxplot(resistivity, range=0)** # 두 Process 상자 그래프 시각화
# range=0 : 최소값과 최대값을 점선으로 연결하는 역할
**boxplot(resistivity, range=0, notch=T)** # 두 Process 상자 그래프 시각화
# notch=T : 중위수 비교시 사용되는 옵션 <- 허리선
**abline(h=0.140, lty=3)** # 기준선 추가(lty=3 : 선 스타일-점선)



# 6. 데이터 시각화

## ② 히스토그램 시각화

```
# 데이터 셋 가져오기
par(mfrow=c(2,2)) # 2행 2열 차트 표현
data(galton) # 자식과 부모의 키 사이의 관계
names(galton) #"child"  "parent"
dim(galton) # 928   2
head(galton,20)
```

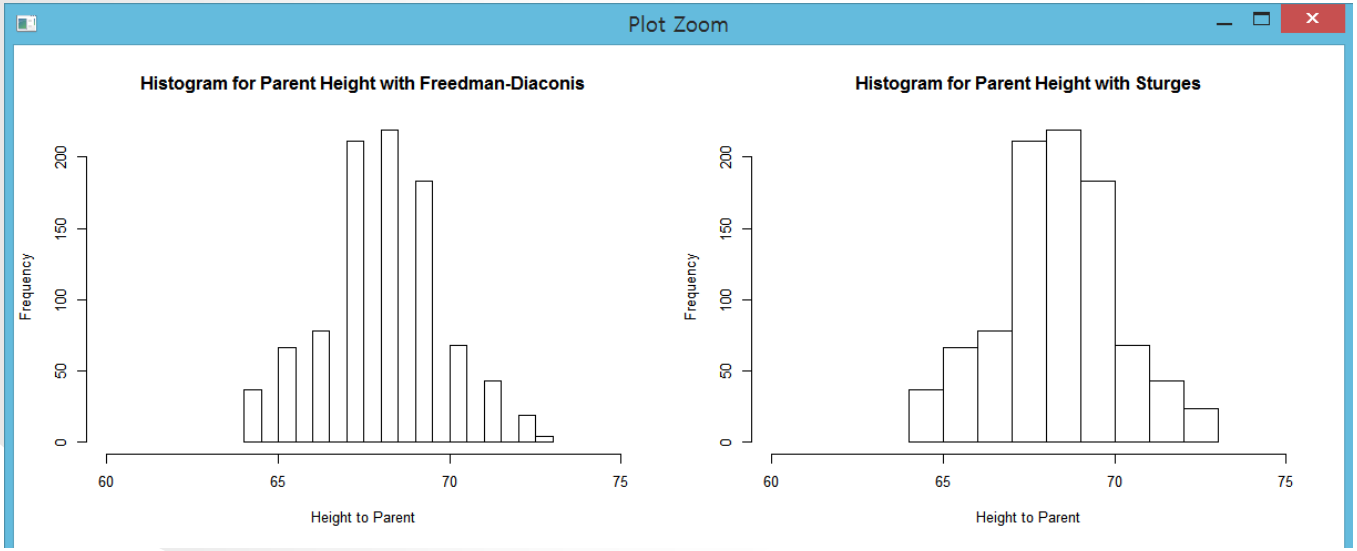
<<galton 데이터 셋에 관한 설명>>

psych 패키지에서 제공하며, Galton에 의해서 작성된 부모와 자식의 키 사이의 관계를 나타낸 데이터 셋으로 변수는 child와 parent를 제공하며, 928개 관측치를 제공한다. 프랜시스 칼턴(Francis Galton)은 영국 유전학자로 우생학 창시자, 종의 기원를 저술한 찰스 다윈(Darwin)의 사촌이다. 우생학이란 유전학, 의학, 통계학을 기초로 우수 유전자 증대를 목적으로 한 학문이다.

# 6. 데이터 시각화

- **히스토그램 시각화(parent)**

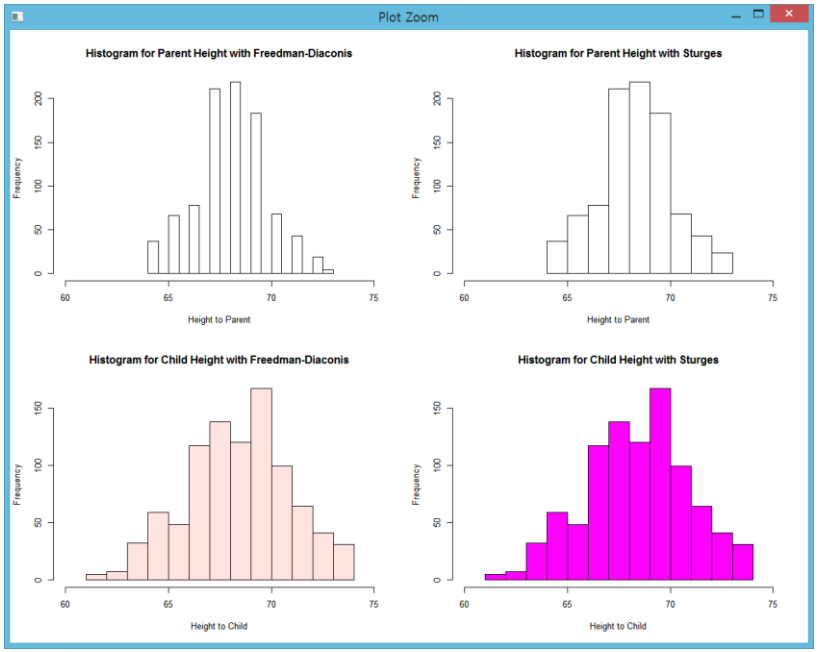
```
hist(galton$parent,breaks="FD", xlab="Height to Parent",
     main="Histogram for Parent Height with Freedman-Diaconis“, xlim=c(60,75))
# breaks="FD" : Freedman-Diaconis, 구간 너비
# xlab : x축 이름, main : 제목, xlim : x축 범위
hist(galton$parent,breaks="Sturges", xlab="Height to Parent",
     main="Histogram for Parent Height with Sturges“, xlim=c(60,75))
#breaks="Sturges"  : 구간 너비
```



# 6. 데이터 시각화

● **히스토그램 시각화(child)**

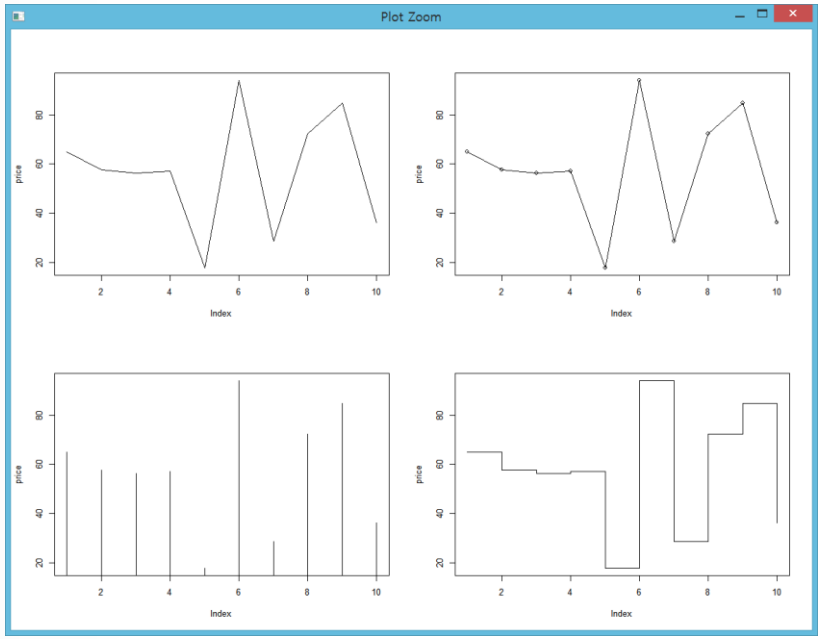
```
hist(galton$child,breaks="FD", xlab="Height to Child",
     main="Histogram for Child Height with Freedman-Diaconis",
     xlim=c(60,75), col="mistyrose")
# col="mistyrose" : 색상(흐릿한 장미) 적용
hist(galton$child,breaks="Sturges",
   xlab="Height to Child“, main="Histogram for Child Height with Sturges",
   xlim=c(60,75), col="magenta")
# col="magenta" : 색상(진홍색) 적용
```



# 6. 데이터 시각화

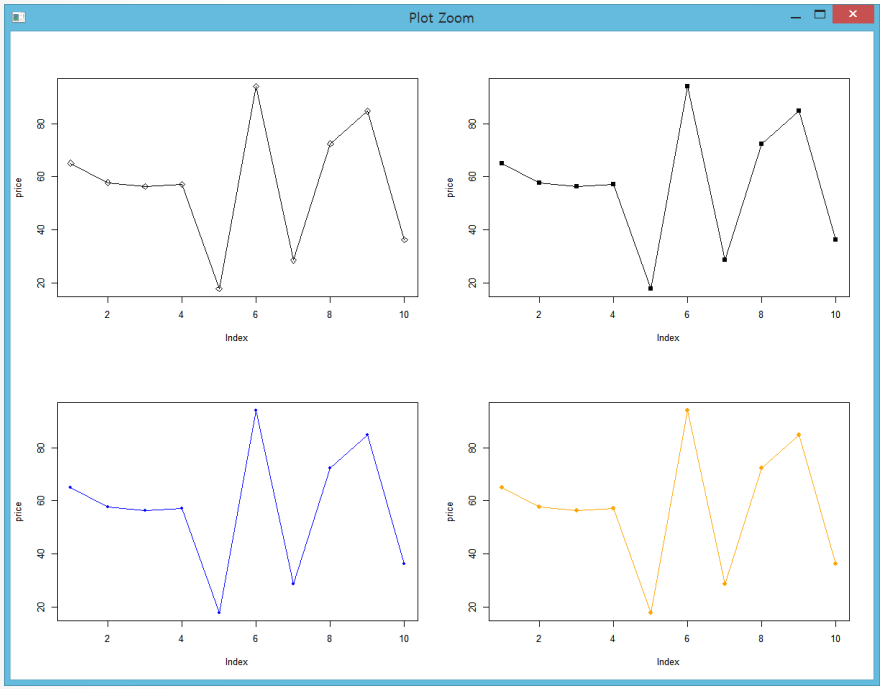
③ 산점도 시각화

```
price<- runif(10, min=1, max=100) # 1~100사이 10개 난수 발생
price #price <-c(1:10)
par(mfrow=c(2,2)) # 2행 2열 차트 그리기
plot(price, type="l") # 유형 : 실선
plot(price, type="o") # 유형 : 원형과 실선(원형 통과)
plot(price, type="h") # 직선
plot(price, type="s") # 꺾은선
```



# 6. 데이터 시각화

```
# plot() 함수 속성 : pch : 연결점 문자타입-> plotting characher-번호(1~30)
plot(price, type="o", pch=5) # 빈 사각형
plot(price, type="o", pch=15)# 채워진 마름모
plot(price, type="o", pch=20, col="blue") #color 지정
plot(price, type="o", pch=20, col="orange", cex=1.5) #character expension(확대)
plot(price, type="o", pch=20, col="green", cex=2.0, lwd=3) #lwd : line width
```



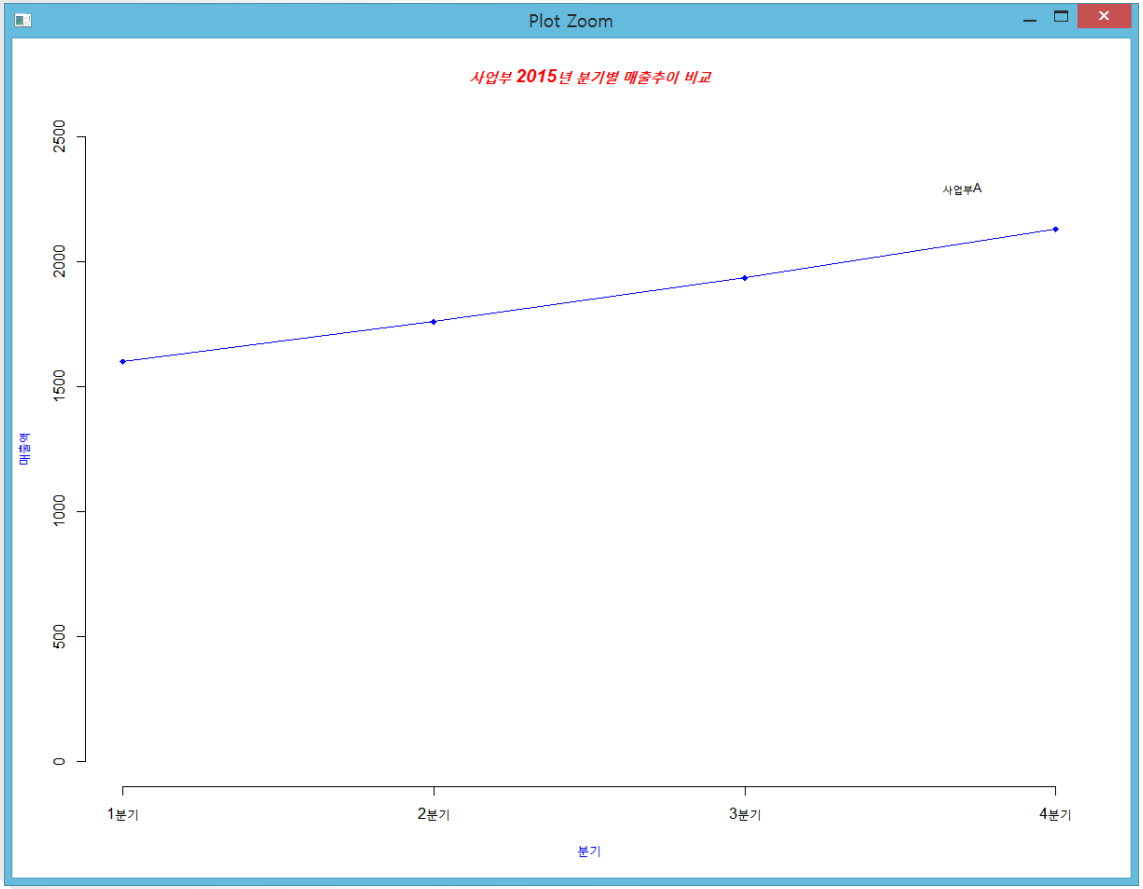
# 6. 데이터 시각화

● 산점도를 이용한 추세 그래프

```
# 데이터 파일 가져오기
bizUnit <- read.csv("C:/Rwork/Part-II/bizUnit.csv", header=TRUE)
bizUnit

par(mfrow=c(1,1)) # 1개 차트 그리기
attach(bizUnit) # bizUnit 생략
plot(Quarter, BU_A, type="o", pch=18, col="blue", ylim=c(0, 2500), axes=T,ann=T)
# axes=F : x축/y축 눈금 제거, ann=F : x축/y축 이름 제거
plot(Quarter, BU_A, type="o", pch=18, col="blue", ylim=c(0, 2500), axes=F,ann=F)

# X축 범위와 이름
axis(1, at=1:4, lab=c("1분기", "2분기", "3분기", "4분기"))
# y축 이름과 범위
axis(2, ylim=c(0, 2500))
text(3.7, 2300, "사업부A", cex=0.8) # 특정 위치에 텍스트 추가
# 제목 추가
title(main="사업부 2015년 분기별 매출추이 비교", col.main="red", font.main=4)
title(xlab="분기", col.lab="blue") # x축 이름
title(ylab="매출액", col.lab="blue") # y축 이름
```

# 6. 데이터 시각화

- x와 y축 이름과 범위, 차트 제목 직접 지정



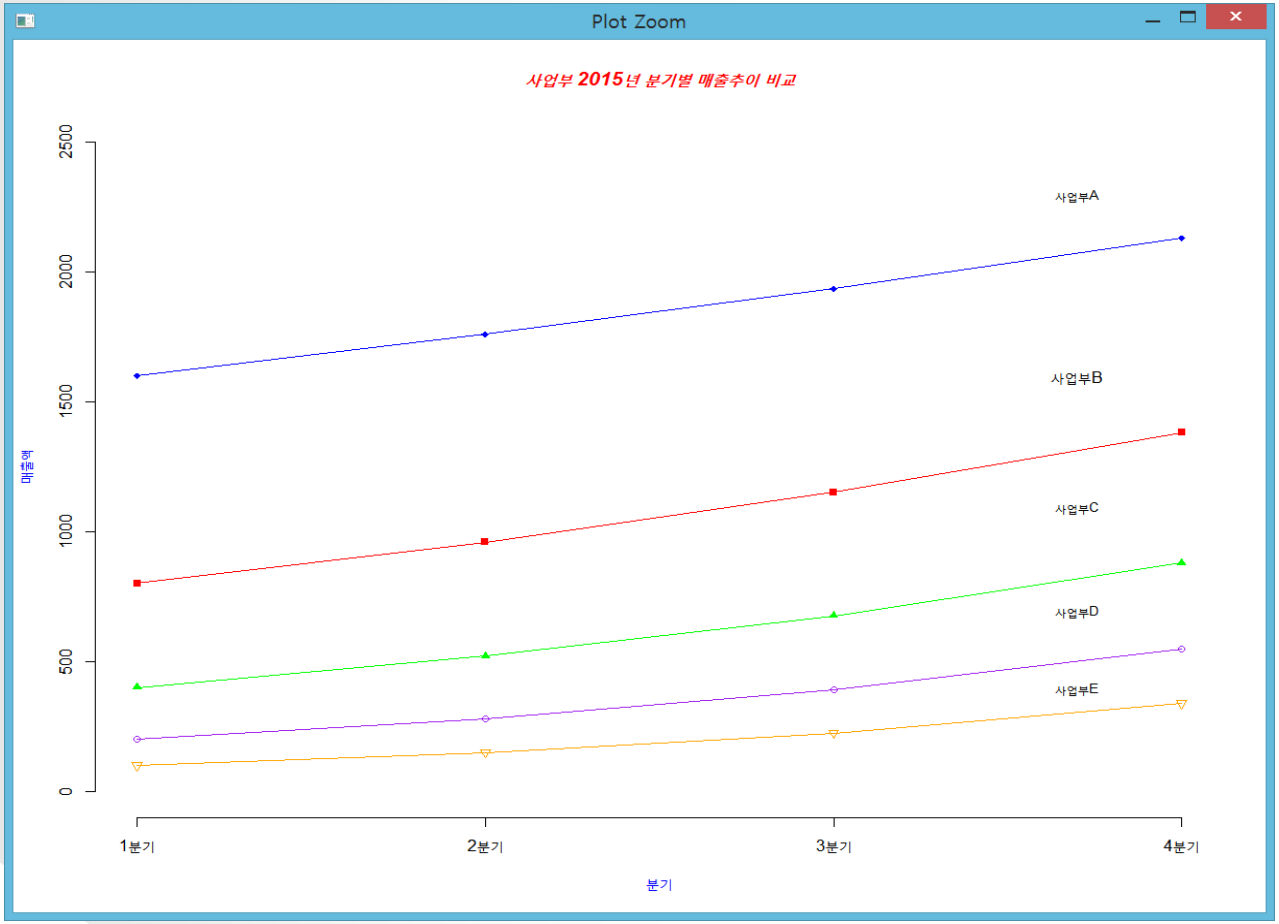
# 6. 데이터 시각화

- 그래프 추가

```
par(new=T) # 그래프 추가
plot(Quarter, BU_B, type="o", pch=15, col="red", ylim=c(0, 2500), axes=F, ann=F)
text(3.7, 1600, "사업부B")

par(new=T)
plot(Quarter, BU_C, type="o", pch=17, col="green", ylim=c(0, 2500), axes=F,
ann=F)
text(3.7, 1100, "사업부C", cex=0.8)

par(new=T)
plot(Quarter, BU_D, type="o", pch=21, col="purple", ylim=c(0, 2500), axes=F,
ann=F)
text(3.7, 700, "사업부D", cex=0.8)

par(new=T)
plot(Quarter, BU_E, type="o", pch=25, col="orange", ylim=c(0, 2500), axes=F,
ann=F)
text(3.7, 400, "사업부E", cex=0.8)
detach(bizUnit) # attach 해제
```

# 6. 데이터 시각화

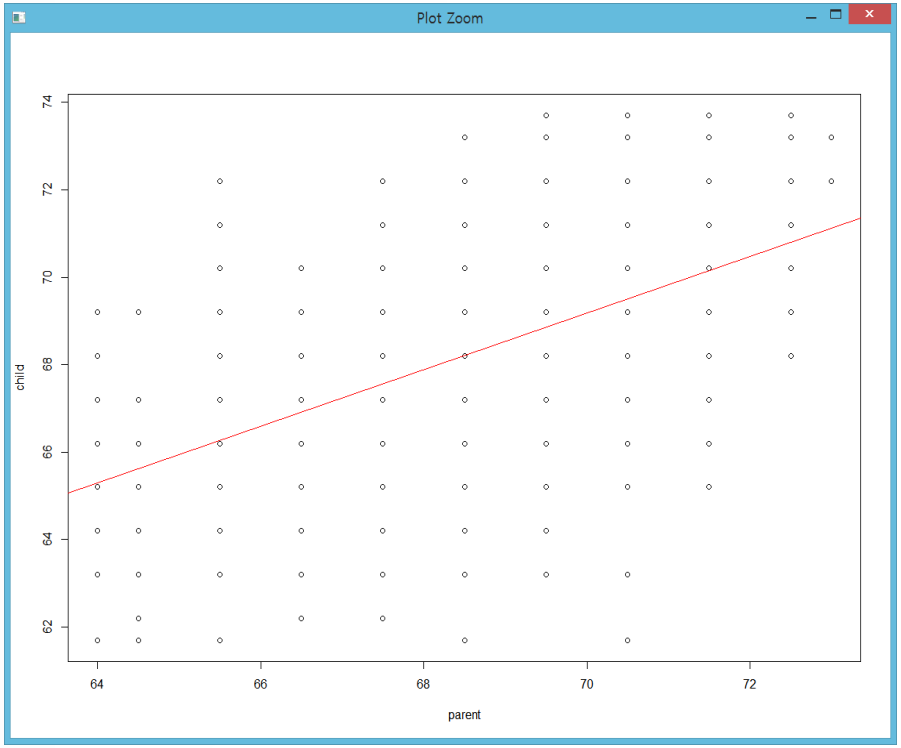
- 추세선 추가



# 6. 데이터 시각화

- 변수 간 비교 시각화

```
# galton 데이터 셋을 이용한 변수 간 상관관계
# parent와 child 변수 대상
par(mfrow=c(1,1))
plot(child~parent, data=galton)
out = lm(child~parent, data=galton)
abline(out, col="red")
```




부모와 자식 간 변수 비교

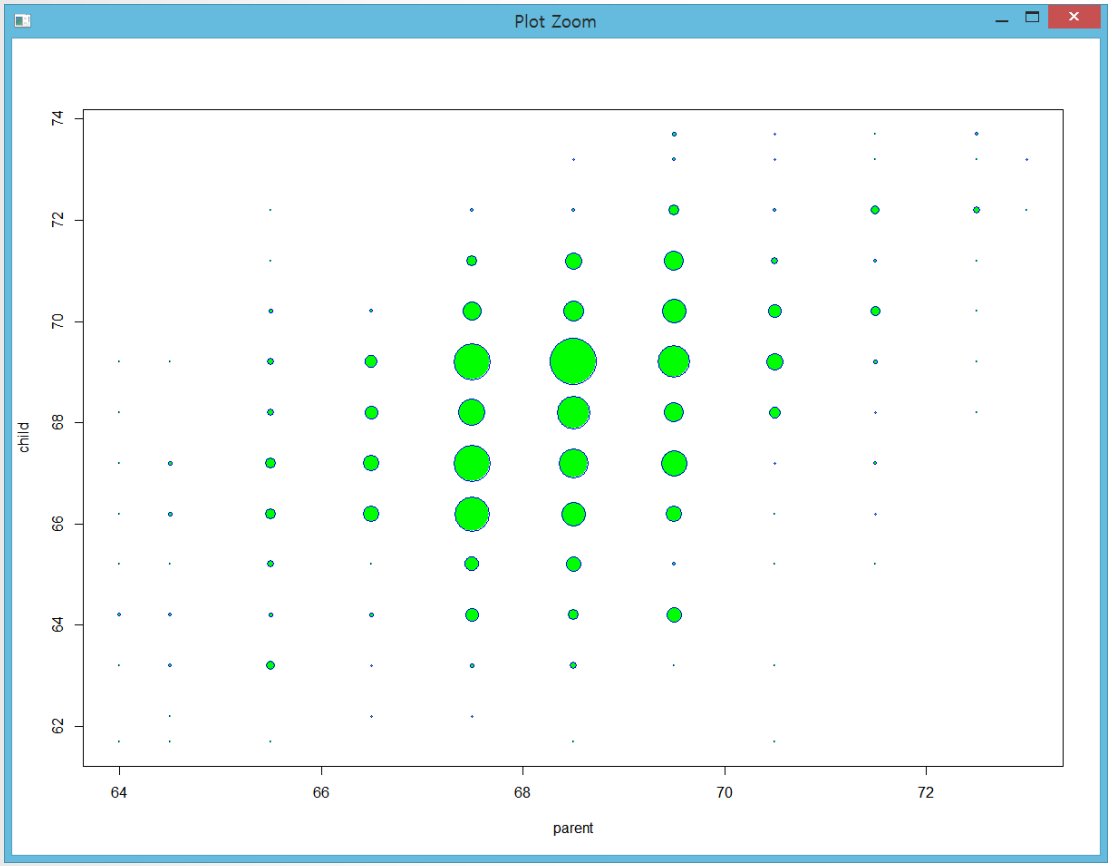
# 6. 데이터 시각화

- 중복 데이터 시각화

```
# 1) 데이터프레임으로 변환 : 컬럼 단위의 데이터 활용을 위해서
freqData <- as.data.frame(table(galton$child, galton$parent))
freqData # Var1 Var2 Freq(중복 수)
str(freqData) # 154 obs(928 관측치가 중복 제외한 154개 관측치 생성 )
names(freqData)=c("child","parent", "freq") # 컬럼에 이름 지정
# 2) 프레임 -> 벡터 -> 수치데이터변환, cex : 빈도수에 0.15 곱(가중치 적)
parent <- as.numeric(as.vector(freqData$parent))
child <- as.numeric(as.vector(freqData$child))
plot(child~parent, pch=21, col="blue", bg="green",
     cex=0.15*freqData$freq, xlab="parent", ylab="child")
```

# 6. 데이터 시각화

- 빈도수를 적용한 가중치 적용




부모와 자식 간 변수 비교

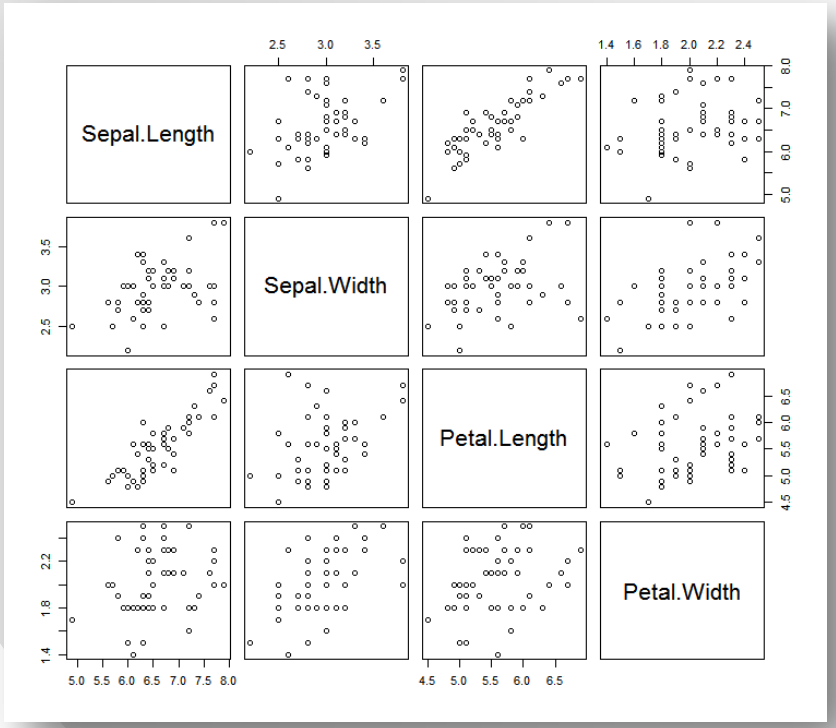
# 6. 데이터 시각화

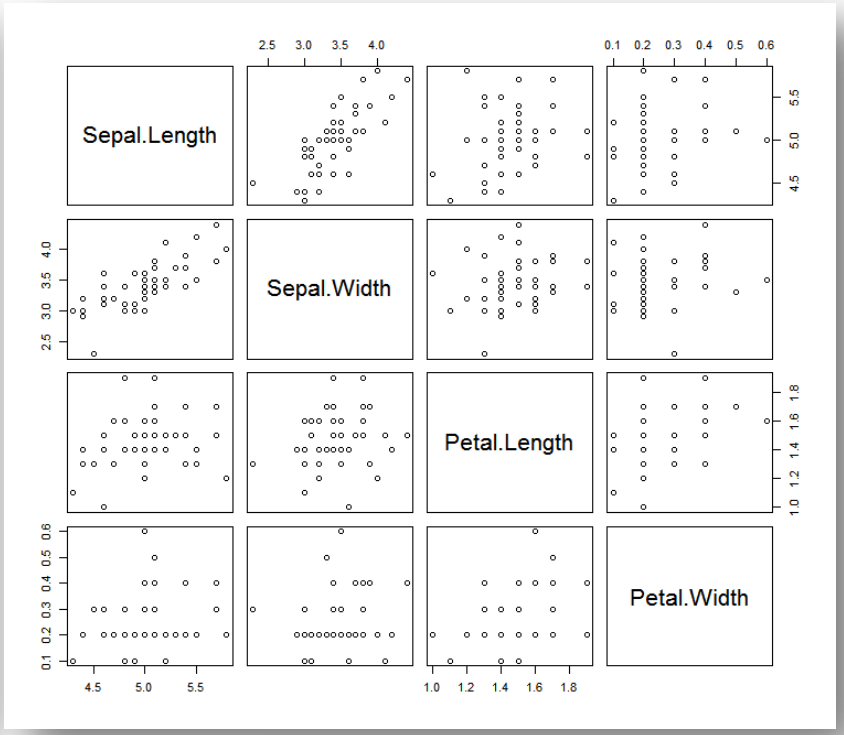
- 데이터 셋 가져오기

```
data(iris)
iris
################## iris 데이터셋 ##########################
#  R에서 제공되는 기본 데이터 셋으로 3가지 꽃의 종류별로 50개의
#  케이스를 제공하여 전체 150개의 관측치와 5개 변수로 구성된다.
#    총 5개의 변수로 구성
# - Sepal.Length(꽃받침 길이), Sepal.Width(꽃받침 너비),
# - Petal.Length(꽃잎 길이), Petal.Width(꽃잎 너비),
# - Species(꽃의 종류) : 3가지 종류별 50개 -> 150개
#########################################################
# 4개 변수 상호비교
pairs(iris[,1:4]) # Sepal.Length Sepal.Width Petal.Length Petal.Width

# Species=="virginica"인 경우만 4개 변수 상호비교
iris[iris$Species=="virginica", 1:4]
# 101~150 레코드

pairs(iris[iris$Species=="virginica", 1:4])
pairs(iris[iris$Species=="setosa", 1:4])
```

# 6. 데이터 시각화

- 변수간 비교 시각화 결과



# 6. 데이터 시각화

<연습문제> iris 데이터 테이블을 대상으로 다음 조건에 맞게 plot을 작성하시오.

조건1) 1번 컬럼이 x축, 3번 컬럼이 y축

조건2) 5번 컬럼으로 색상 지정 - 형식) plot( ,col= )

조건3) "iris 데이터 테이블 산포도 차트" 제목 추가

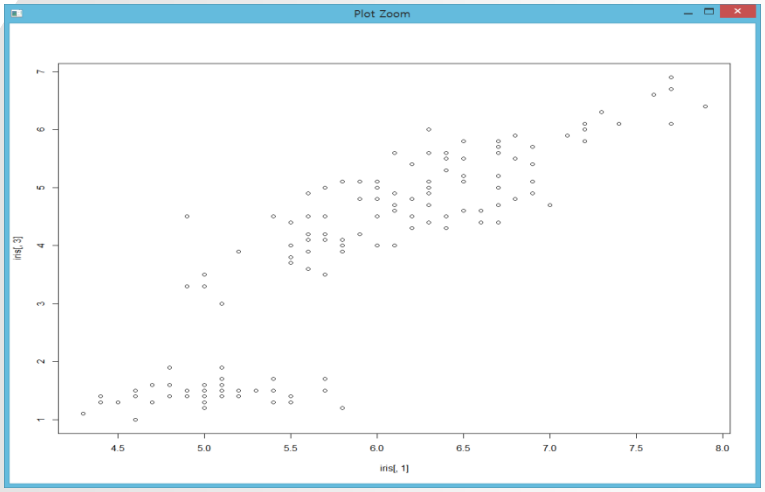
```
iris
names(iris)
```

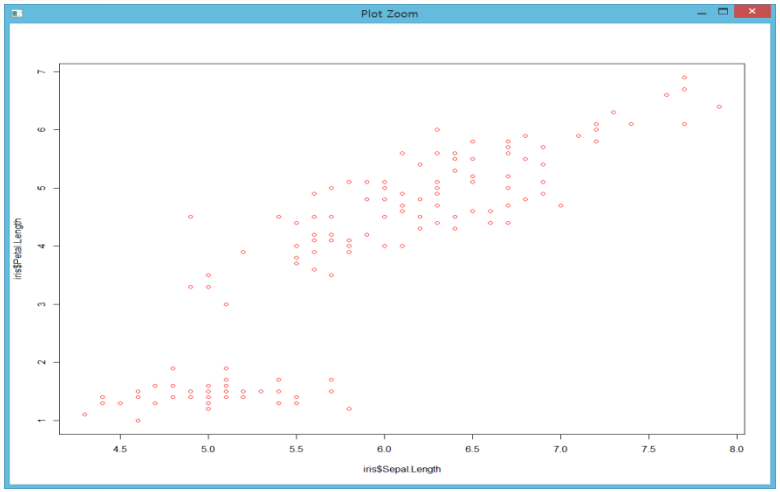
조건1) 1번 컬럼이 x축, 3번 컬럼이 y축

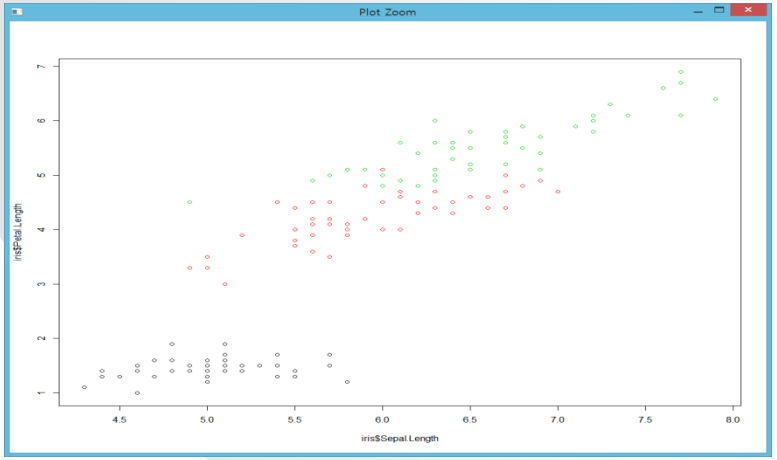
조건2) 5번 컬럼으로 색상을 지정하시오.

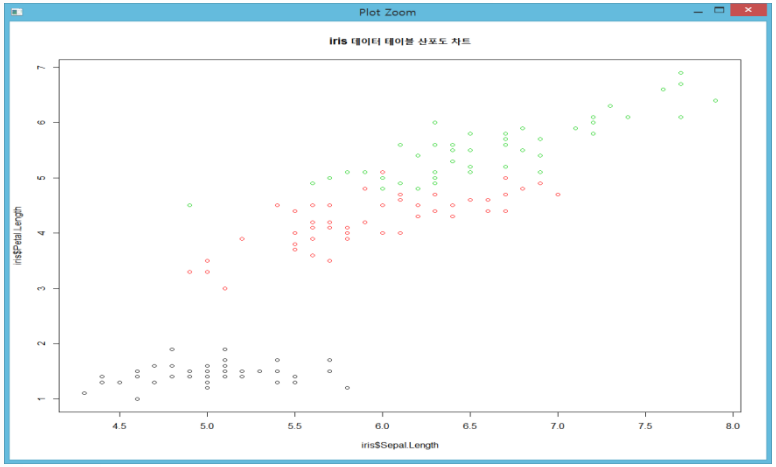
조건3) 제목 추가

# 6. 데이터 시각화


Plot Zoom
iris[, 3]
iris[, 1]

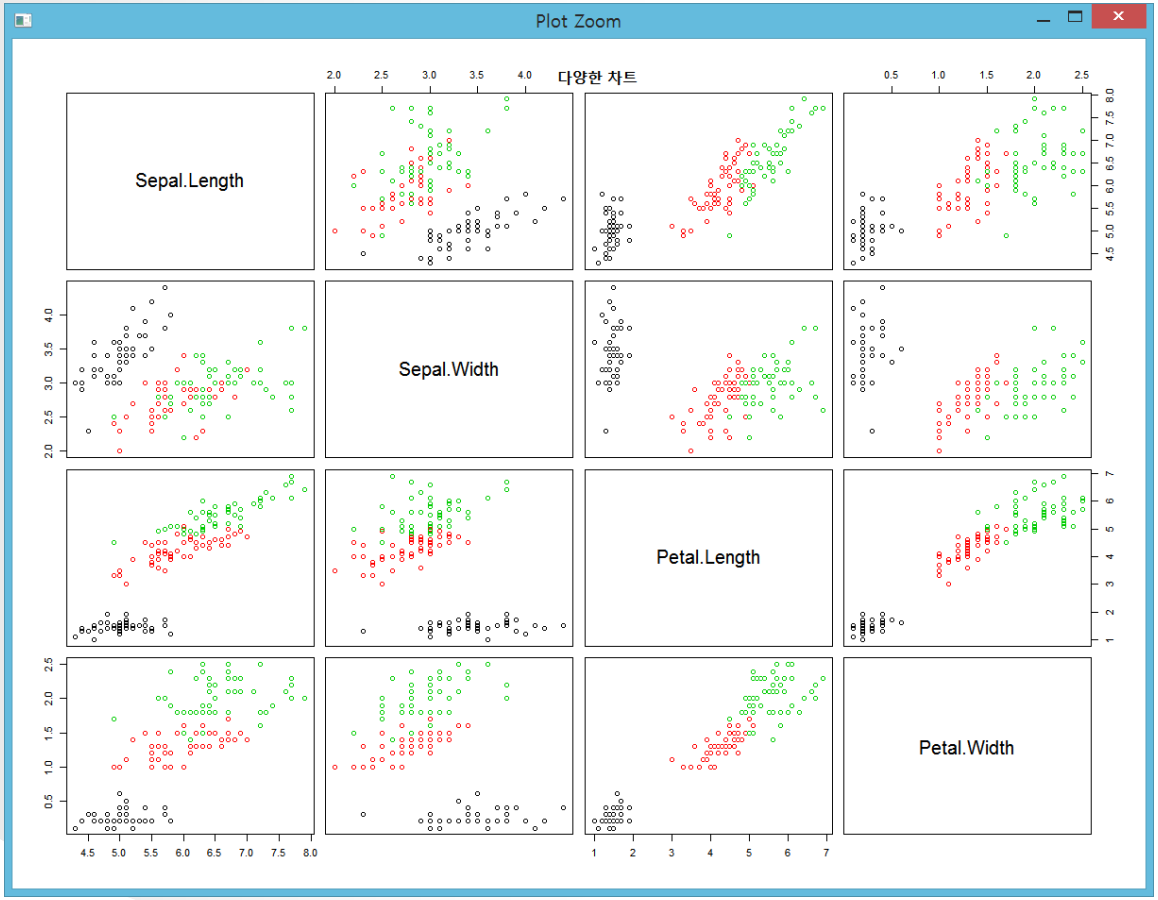

Plot Zoom
iris$Petal.Length
iris$Sepal.Length


Plot Zoom
iris$Petal.Length
iris$Sepal.Length


Plot Zoom
iris 데이터 데이블 산포도 차트
iris$Petal.Length
iris$Sepal.Length

# 6. 데이터 시각화

```
plot(iris) # iris 데이터를 대상으로 제공되는 모든 차트 그려줌
plot(iris[, -5], col=iris[,5]) # 5번컬럼 제거, 색지정으로 사용
title(main="다양한 차트")
```



# 6. 데이터 시각화

```
# 파일로 차트 저장하기
setwd("C:/Rwork/Part-II") # 폴더 지정
jpeg("iris.jpg", width=720, height=480) # 픽셀 지정 가능
plot(iris$Sepal.Length, iris$Petal.Length, col=iris$Species)
title(main="iris 데이터 테이블 산포도 차트")
dev.off() # 장치 종료
# "c:/Rwork/Part-II" <- 이미지 파일 확인
```

# 7. 데이터 전처리

## chap07_DataPreprocessing 수업내용

1) 탐색적 데이터 분석
2) 결측값(NA) 처리
3) 극단치(이상치) 처리
4) 역 코딩/디코딩
5) 파생변수 생성
6) 표본 셈플링

# 7. 데이터 전처리

- 실습 데이터 읽어오기

```
getwd()
setwd("C:/Rwork/Part-II")
dataset <- read.csv("dataset.csv", header=TRUE) # 헤더가 있는 경우
#dataset 변수에 저장 : 메모리 로딩
dataset # resident gender job age position  price survey
```

# 7. 데이터 전처리

## 1) 탐색적 데이터 분석

- 데이터 셋 보기

  print(dataset)

  View(dataset) # 별도의 데이터 뷰어창에서 출력됨, 컬럼 정렬

  #간단이 앞쪽/뒤쪽 조회

  head(dataset) # 앞부분 데이터 셋 6개

  tail(dataset) # 끝부분 데이터 셋 6개

  head(dataset, 10) # 앞부분 10개

data Mart(dataset.csv)

| # resident | gender | job | age | position | price | survey |
|---|---|---|---|---|---|---|
| # 명목 | 명목 | 명목 비율 | 서열 | 비율 | 등간 | |

# 7. 데이터 전처리

- **데이터 셋 구조보기**

  **#dataset에 들어 있는 세부 정보항목 조회**

  **names(dataset)** # 변수명(컬럼)

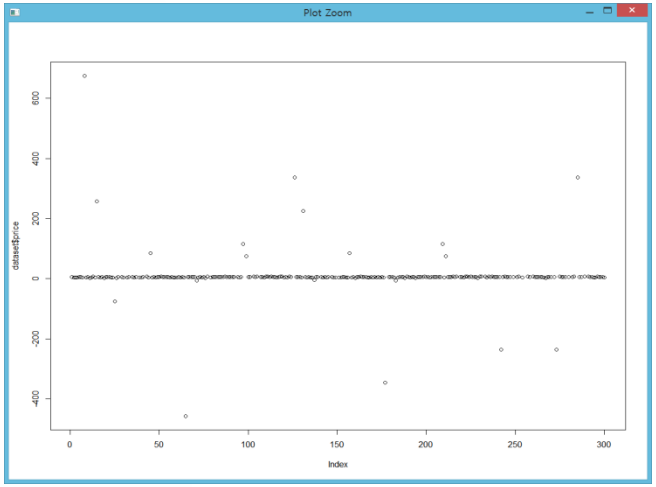
  **attributes(dataset)** # names(열이름), class, row.names(행이름)

  **str(dataset)** # 데이터 구조보기

# 7. 데이터 전처리

- 데이터 셋 조회

```
#dataset 데이터 중 특정변수 조회
dataset$age # [1] [27] [148] age값의 색인
dataset$resident
length(dataset$age) # data 수-300개

x <- dataset$gender # 결과를 변수에 저장
y <- dataset$price
x;y

plot(dataset$price) # 산점도 형태 전반적인 가격분포 보기

# $기호 대신 [""]기호를 이용한 특정변수 조회
dataset["gender"]
dataset["price"]
```



# 7. 데이터 전처리

```
# 색인(index)으로 변수의 위치값 조회
dataset[2] #두번째 컬럼(gender)
dataset[6] #여섯번째 컬럼(price)
dataset[3,] #3번째 관찰치(행) 전체 -> 열 공통
dataset[,3] # 전체행의 3번째 변수(열) -> 행 공통
```

# 7. 데이터 전처리

```
# dataset 데이터 중 변수를 2개 이상 조회하는 경우
dataset[c("job","price")]
dataset[c(2,6)] # gender, price

dataset[c(1,2,3)] #resident,gender,age
dataset[c(1:3)] #resident,gender,age
dataset[c(2,4:6,3,1)] #gender,age,position,price,job,resident

# dataset 데이터 중 특정 행/열을 지정해 조회
dataset[,c(2:4)] #2~4열(gender job age) 전체 -> test[c(2:4)]과 동일
dataset[c(2:4),] #2~4행 전체
dataset[-c(1:100),] # 1~100행 제외
```

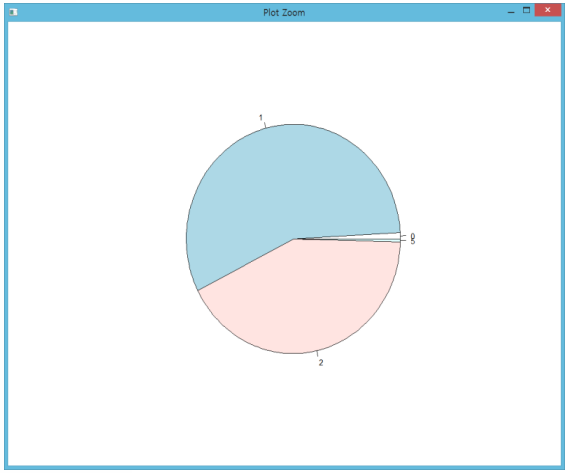
# 7. 데이터 전처리

2) 결측값(NA) 처리

```
summary(dataset$price) # 결측치 확인 -> NA's - 30개
sum(dataset$price) # NA 출력

# 결측데이터 제거 방법1
sum(dataset$price, na.rm=T) # 2362.9

# 결측데이터 제거 방법2
price2 <- na.omit(dataset$price) # price에 있는 모든 NA 제거
sum(price2) # 2362.9
length(price2) # 270 -> 30개 제거
```

# 7. 데이터 전처리

### 3) 극단치 발견과 처리



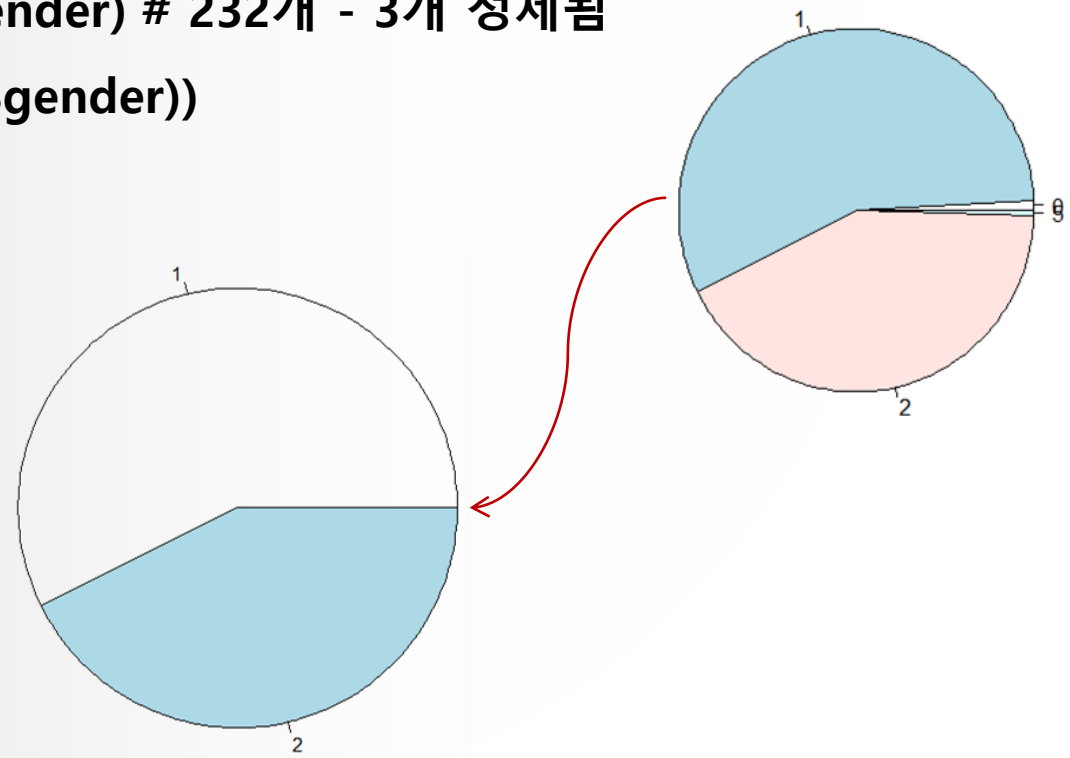

(1) 범주형 변수 극단치 처리

```
# gender 변수 outlier 확인
gender <- dataset$gender
hist(gender) # 히스토그램으로 outlier 확인
table(gender) # 빈도수로 outlier 확인
pie(table(gender)) # 파이 차트로 outlier 확인
```

# 7. 데이터 전처리

- 성별 데이터 정제 - subset() 함수 이용

```
data <- subset(data,data$gender == 1 | data$gender == 2)
data # gender변수 데이터 정제
length(data$gender) # 232개 - 3개 정제됨
pie(table(data$gender))
```



# 7. 데이터 전처리

2) 연속형 변수 극단치 처리

```
# price outlier 확인
dataset$price # 세부 데이터 보기
length(dataset$price) #300개(NA포함)
plot(dataset$price) # 산점도 형태 전반적인 가격분포 보기
summary(dataset$price) # -457~675 범위확인
```

# 7. 데이터 전처리

**# price변수 정제(2~8)**

**data <- subset(dataset, dataset$price >= 2 & dataset$price <= 8)**

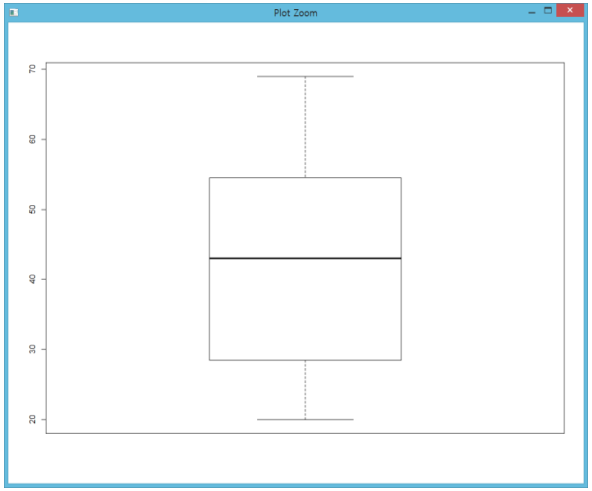
**length(data$price) #251개(49개 정제)**

**stem(data$price) # 줄기와 잎 도표보기**

```
The decimal point is at the |

  2 | 133
  2 |
  3 | 000003344
  3 | 555589
  4 | 00000000000000111111111222333334444
  4 | 566666777889999
  5 | 000000000000000000111111111122222222233333444444
  5 | 5555555556677777788899
  6 | 000000000011111122222222222223333333333333344444444
  6 | 5555777777788889999
  7 | 000111122
  7 | 7799
```

# 7. 데이터 전처리

```
# age 변수 NA 발견
summary(data$age) # Min(20), Max(69), NA(16)
length(data$age) # 251

# age 변수 정제(20~69)
data <- subset(data,data$age >= 20 & data$age <= 69)
length(data$age) # 235개(16 정제)
# box 플로팅으로 평균연령 분석
boxplot(data$age) # 45대 중반 평균 연령
```



# 7. 데이터 전처리

4) 역코딩 : 긍정순서(1 -> 5, 5 -> 1 )

```
data$survey
survey <- data$survey
csurvey <- 6-survey
csurvey
survey  # 역코딩 결과와 비교
data$survey <- csurvey # data set에 survery 변수 수정
head(data) # survey 결과 확인
```

| | resident | gender | job | age | position | price | survey |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 26 | 2 | 5.1 | 1 |
| 2 | 2 | 1 | 2 | 54 | 5 | 4.2 | 2 |
| 3 | NA | 1 | 2 | 41 | 4 | 4.7 | 4 |
| 4 | 4 | 2 | NA | 45 | 4 | 3.5 | 2 |
| 5 | 5 | 1 | 3 | 62 | 5 | 5.0 | 1 |
| 6 | 3 | 1 | 2 | 57 | NA | 5.4 | 2 |

# 7. 데이터 전처리

## 5) 코딩변경 - 변수변환 : 리코딩 하기

```
# 문자열로 리코딩(청년층, 중년층, 장년층)
data$age2[data$age <= 30] <-"청년층"
data$age2[data$age > 30 & data$age <=45] <-"중년층"
data$age2[data$age > 45] <-"장년층"

head(data) # data 테이블 전체 - age와 age2 비교
head(data[c("age","age2")]) # 2개만 지정
# age  age2
# 26  청년층
head(data) # dataset 테이블 전체 - age2 컬럼 생성
head(data[c("age","age2")]) # 2개만 지정
# age  age2  age3
# 26  청년층  1
```

# 7. 데이터 전처리

6) 파생변수 생성 : 기존 데이터로 새로운 변수 생성

```
data$resident2[data$resident == 1] <-"특별시"
data$resident2[data$resident >=2 & data$resident <=4] <-"광역시"
data$resident2[data$resident == 5] <-"시구군"
head(data) # data 테이블 전체 - age2 컬럼 생성
head(data[c("resident","resident2")]) # 2개만 지정
```

| | resident | resident2 |
|---|---|---|
| 1 | 1 | 특별시 |
| 2 | 2 | 광역시 |
| 3 | NA | <NA> |
| 4 | 4 | 광역시 |
| 5 | 5 | 시구군 |
| 6 | 3 | 광역시 |

# 7. 데이터 전처리

## 7) 표본 추출

(1) 정제된 데이터 파일 저장(cleanData.csv)

```
print(data) # 정제 데이터 확인
getwd() # 작업 디렉터리 확인
setwd("c:/Rwork/Part-II") # 저장 디렉터리 지정

# 따옴표와 행 이름 제거하여 저장
write.csv(data,"cleanData.csv", quote=F, row.names=F)

# 저장된 파일 불러오기/확인
data <- read.csv("cleanData.csv", header=TRUE)
data # 저장된 파일 불러오기/확인
length(data$age)# 235개 정제 데이터 확인
```

# 7. 데이터 전처리

2) 정제 데이터 대상 셈플링하기
nrow(data) # 235개 : 행수 구하기 -> Number of Rows

```
# 235개 중 30개 무작위 추출
choice1 <- sample(nrow(data), 30)
choice1 # 추출된 행 번호 출력

# 50~235 사이에서 30개 무작위 추출
choice2 <- sample(50:nrow(data), 30)
choice2

# 50~100 사이에서 30개 무작위 추출
choice3 <- sample(c(50:100), 30)
choice3
```

# Sampling 결과는 데이터 셋에서 선택된 레코드 번호를 의미

# 7. 데이터 전처리

```
#다양한 범위를 지정해서 무작위 셈플링
choice4 <- sample(c(10:50, 70:150, 160:190),30)
choice4

# 마지막 행수 직접 입력
choicePrice <- sample(1:235,30)
choicePrice # 셈플링 결과
length(choicePrice) #30개 생성

# 특정 변수 대상 셈플링 불가
data2 <- data$gender
data2
choice <- sample(1:nrow(data2),30)
# Error in 1:nrow(data2) : argument of length 0
```

# 7. 데이터 전처리

<연습문제1> cleanData.csv 파일을 불러와서 data 변수에 저장하시오.

<연습문제2> 직급(position) 변수를 대상으로 1급 -> 5급, 5급 -> 1급 형식으로 역코딩하여 position변수를 수정 하시오.

<연습문제3> resident변수의 NA 값을 제거한 후 data 변수에 저장하시오.

<연습문제4> gender변수를 대상으로 1 -> "남자", 2 -> "여자" 형태로 리코딩 하여 gender2변수에 추가한 후 파이 차트로 결과를 확인하시오.

# 7. 데이터 전처리

<연습문제5> 나이를 30세 이하-> 1, 30~45-> 2, 45이상-> 3 으로 리코딩하여 age3변수에 추가한 후 age, age2, age3 변수 3개만 확인하시오.

<연습문제6> 정제된 데이터를 "c:/Rwork/Part-II/cleanData.csv" 파일에 따옴표와 행 이름 제거하여 저장하시오.

# 8. 정형과 비정형 데이터 처리

## Chap08_Formal_InFormal 수업내용

1) 정형 데이터 처리 - Oracle DB 데이터 처리
   - ✓ DB(RDB) 연결 - ODBC, JDBC, DBI
   - ✓ Oracle/MySql 실습
2) 비정형 데이터 처리 - SNS 데이터 분석(텍스트 마이닝)

   1단계 : 토픽분석(단어의 빈도수)

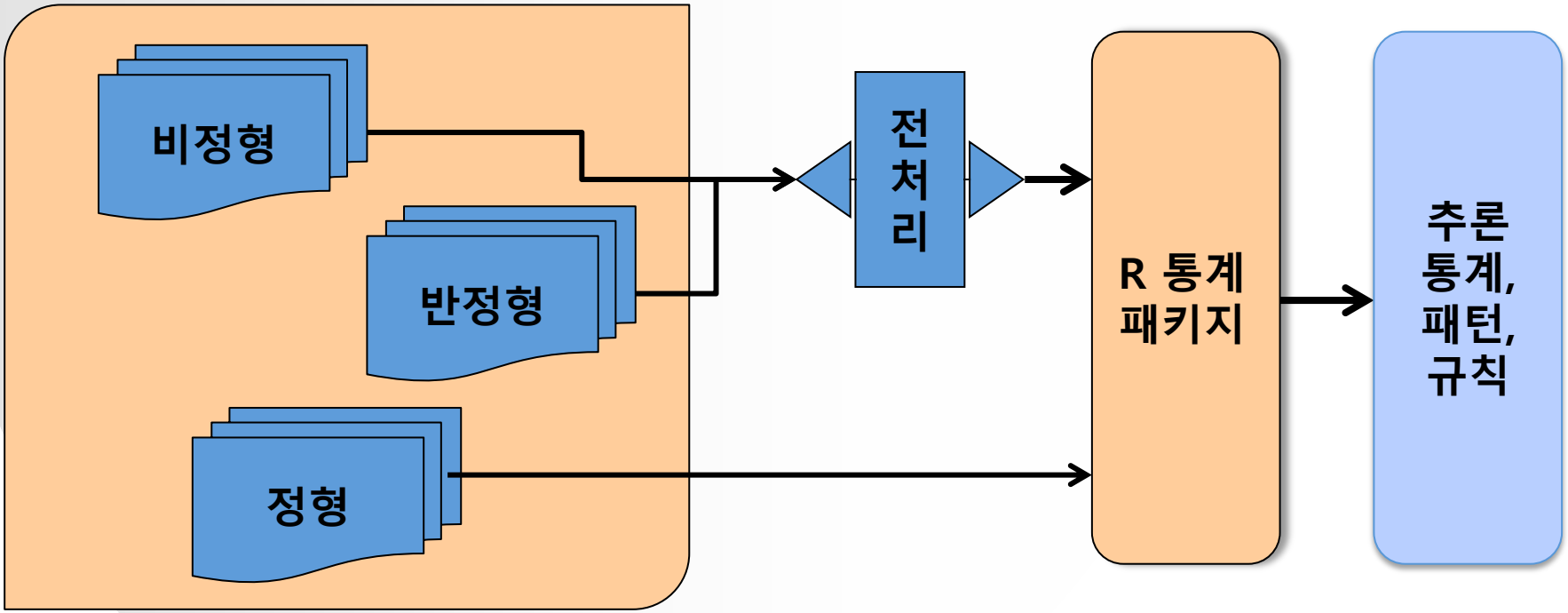
   2단계 : 연관어 분석(관련 단어 분석)

   3단계 : 감성 분석(단어의 긍정/부정 분석)

   4단계 : 소셜 네트워크 분석

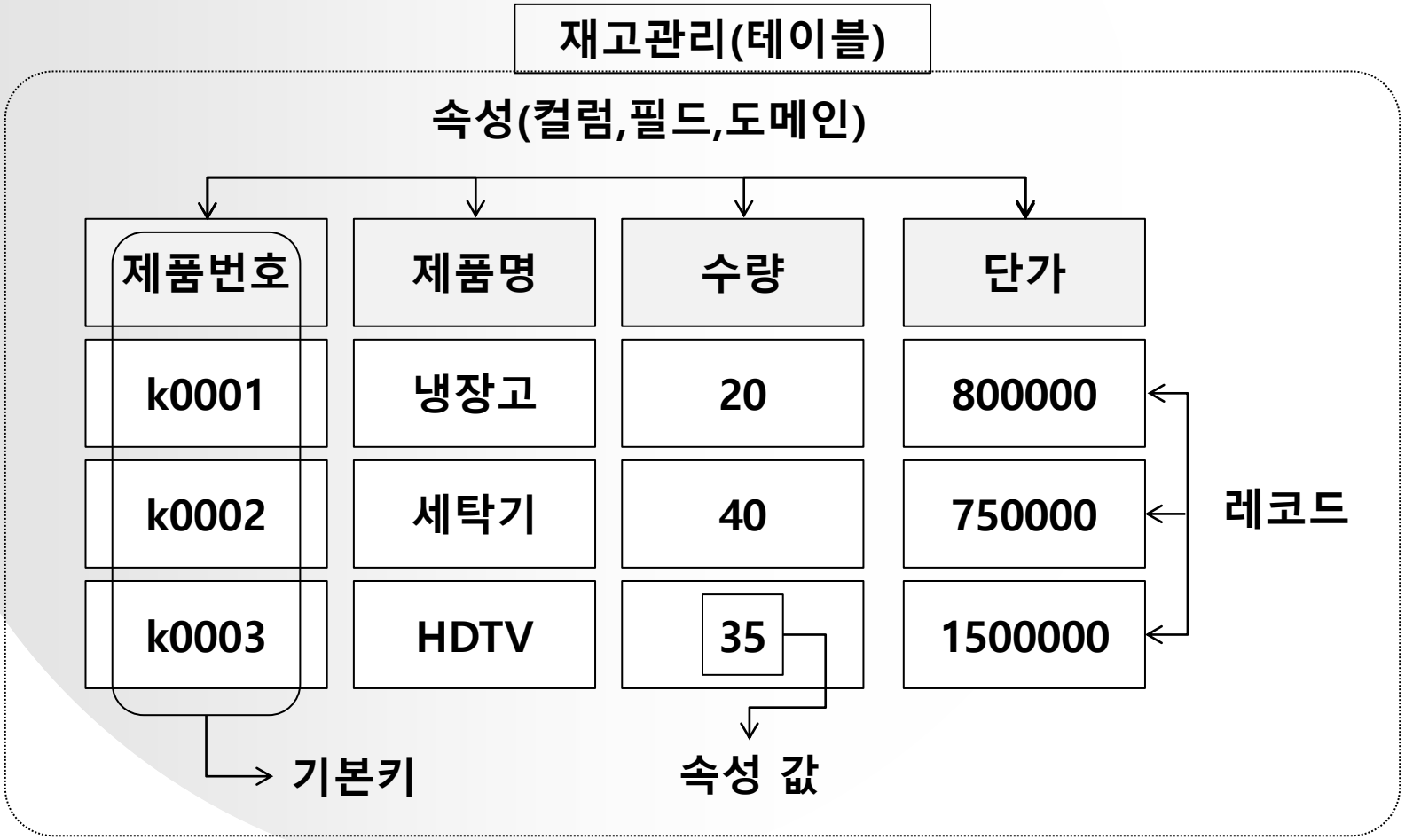
# 8. 정형과 비정형 데이터 처리

- 정형과 비정형 데이터 처리 과정



# 8. 정형과 비정형 데이터 처리

- 관계형데이터베이스의 테이블 구조



# 8. 정형과 비정형 데이터 처리

## 1) 정형 데이터(RDB-Oracle)

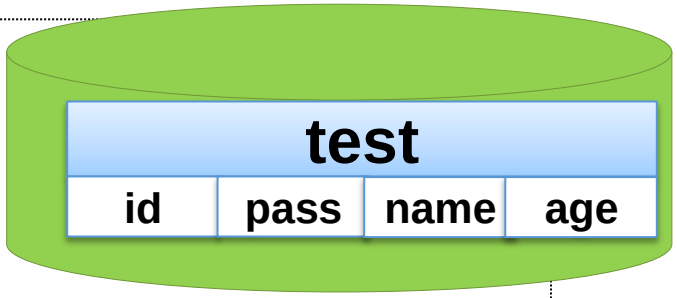
① 패키지 설치

```
# 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_31')
library(rJava)
library(RJDBC) # rJava에 의존적이다.(rJava 먼저 로딩)
```

# 8. 정형과 비정형 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);
```



# 8. 정형과 비정형 데이터 처리

④ **Oracle 연동**

```
# driver
drv<-JDBC("oracle.jdbc.driver.OracleDriver",
  "C:/app/jinsung/product/11.2.0/dbhome_1/jdbc/lib/ojdbc6.jar")
# db연동(driver, url,uid,upwd)
conn<-dbConnect(drv,
        jdbc:oracle:thin:@//127.0.0.1:1521/orcl","scott","tiger")
query = "SELECT * FROM test"
dbGetQuery(conn, query)
#     ID   PASS NAME  AGE
#1 hong 1234  홍길동  35
#2  lee   1234  이순신  45
```

# 8. 정형과 비정형 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
```

# 8. 정형과 비정형 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
###########################################################
```

# 8. 정형과 비정형 데이터 처리

## 2) 비정형 데이터 – 텍스트 마이닝 분석


# 8. 정형과 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

# 8. 정형과 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

  단계4 : 소셜 네트워크 분석
  - 영향자(inplancer)

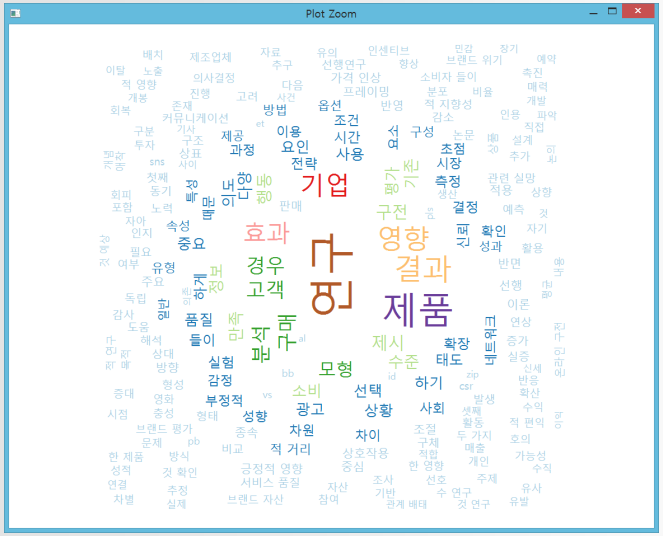
❖ 형태소 분석 : 문장을 분해 가능한 최소한의 단위로 분리하는 작업

# 8. 정형과 비정형 데이터 처리

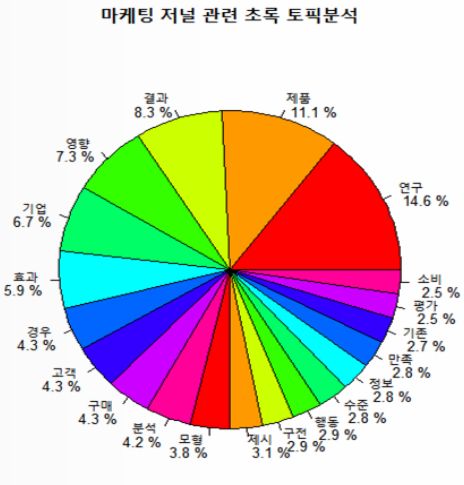
```
##################################################
#    단계1 - 토픽분석(텍스트 마이닝)                    #
#          √ 시각화 : 단어 빈도수에 따른 WordCloud      #
##################################################
```

# 8. 정형과 비정형 데이터 처리

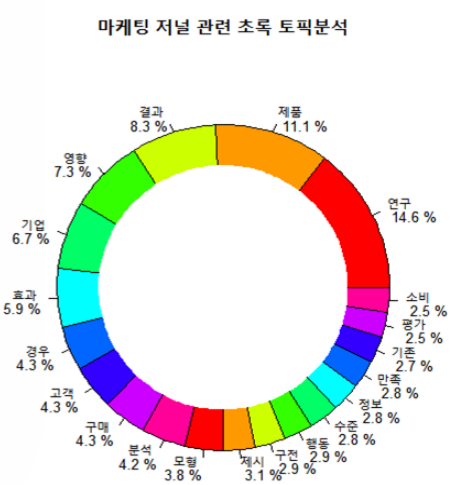
## 단계1 : 토픽분석(결과물)




단어구름(Wordcloud)




Pie 차트 시각화




도넛 차트 시각화

# 8. 정형과 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_31')**
**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)

# 8. 정형과 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"

# 8. 정형과 비정형 데이터 처리

## 1. 데이터셋(abstract.txt) 가져오기

```
data = read.csv("C:/Rwork/Part-II/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

# 8. 정형과 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
```

# 8. 정형과 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.

# 8. 정형과 비정형 데이터 처리

## 5. 단어추출 사용자 함수 정의 및 단어추출

```
# (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 ....
```

# 8. 정형과 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
```

# 8. 정형과 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
```

# 8. 정형과 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
```

# 8. 정형과 비정형 데이터 처리

- **Wordcloud 생성(디자인 전)**

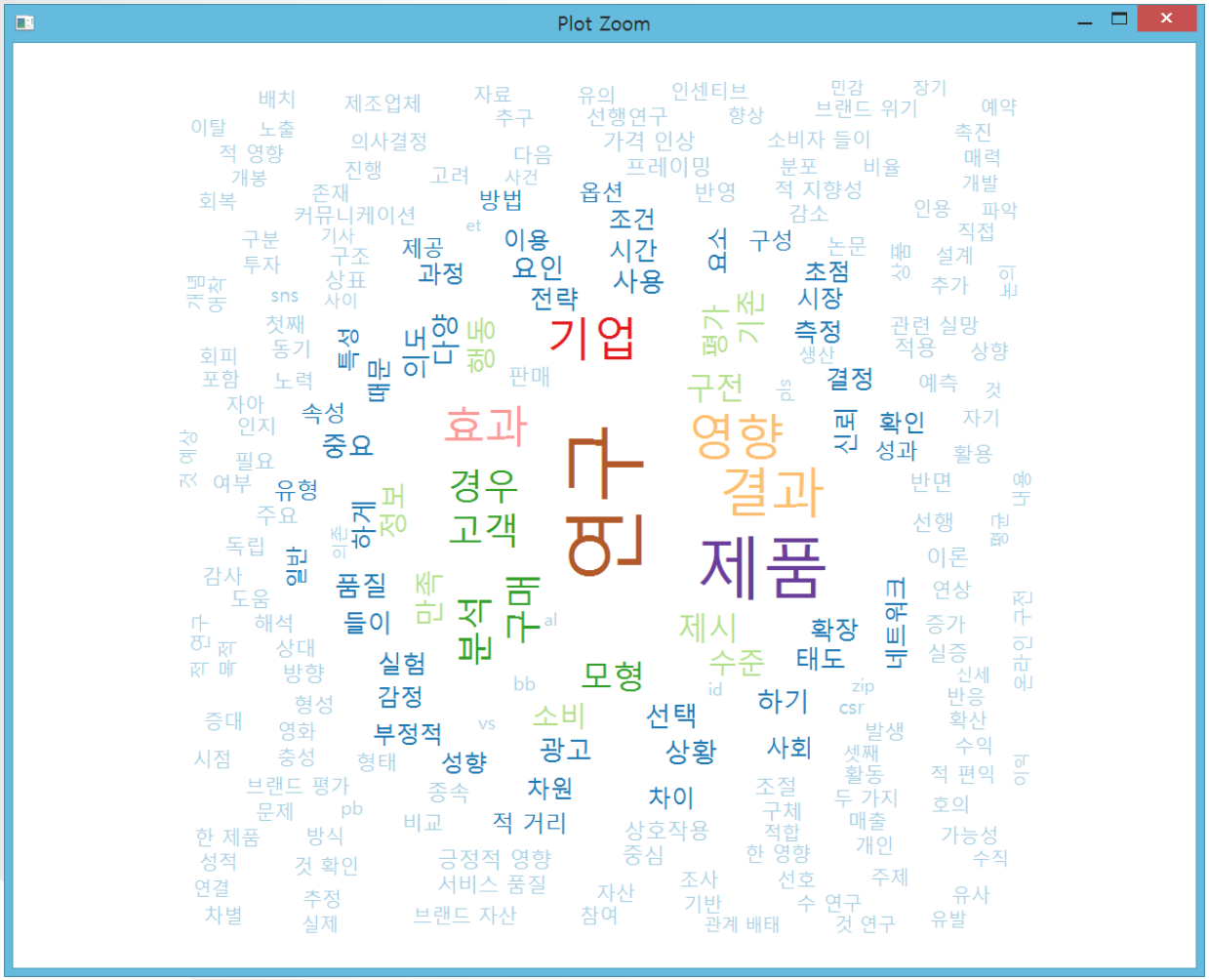
# 8. 정형과 비정형 데이터 처리

## 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 )
```

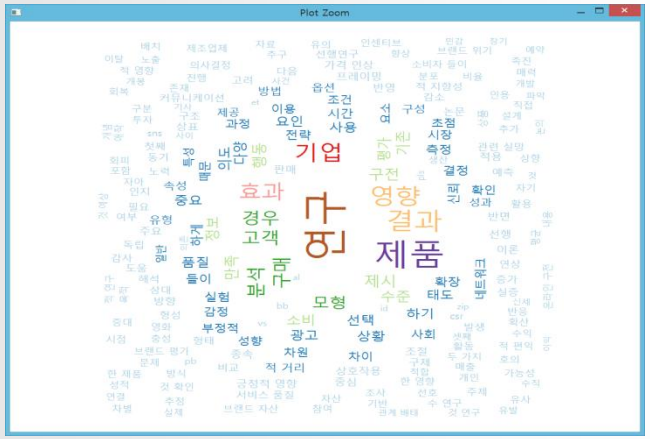
# 8. 정형과 비정형 데이터 처리

- **Wordcloud 생성(디자인 후)**

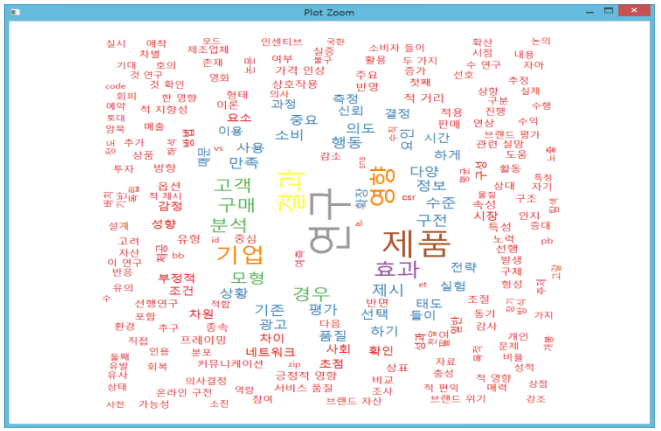


# 8. 정형과 비정형 데이터 처리

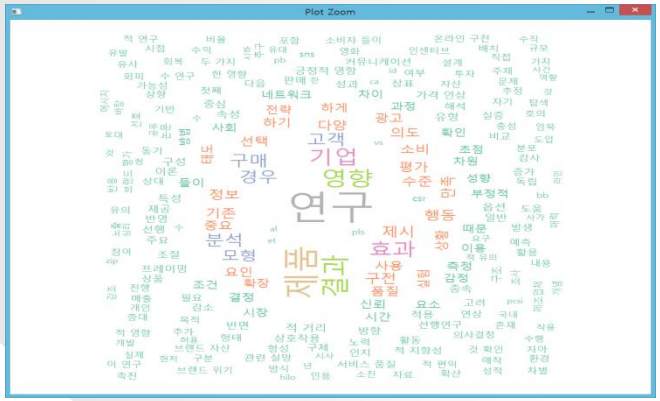
- **brewer.pal() 색상 지정**




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




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



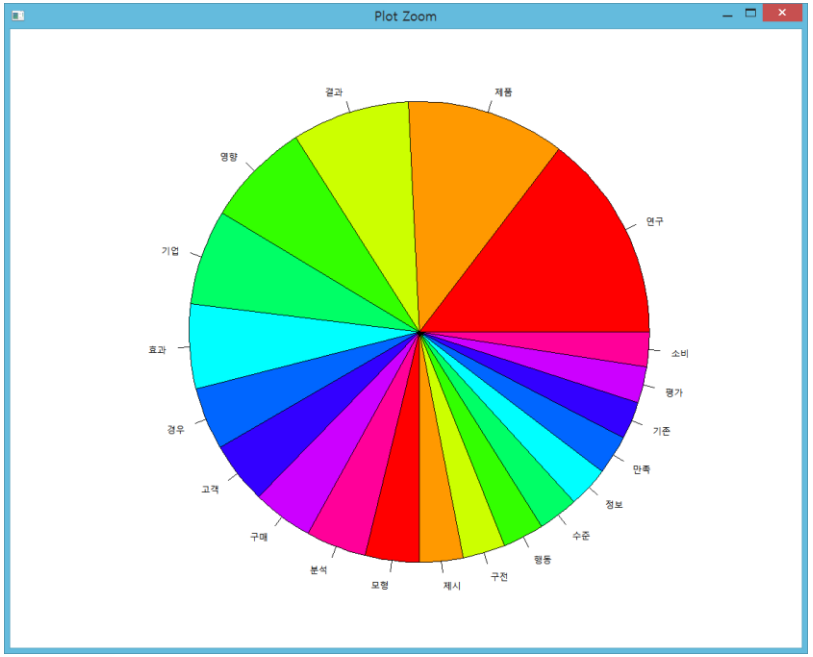

**pal <- brewer.pal(12,"Set2")**


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

# 8. 정형과 비정형 데이터 처리

## 10. 차트 시각화

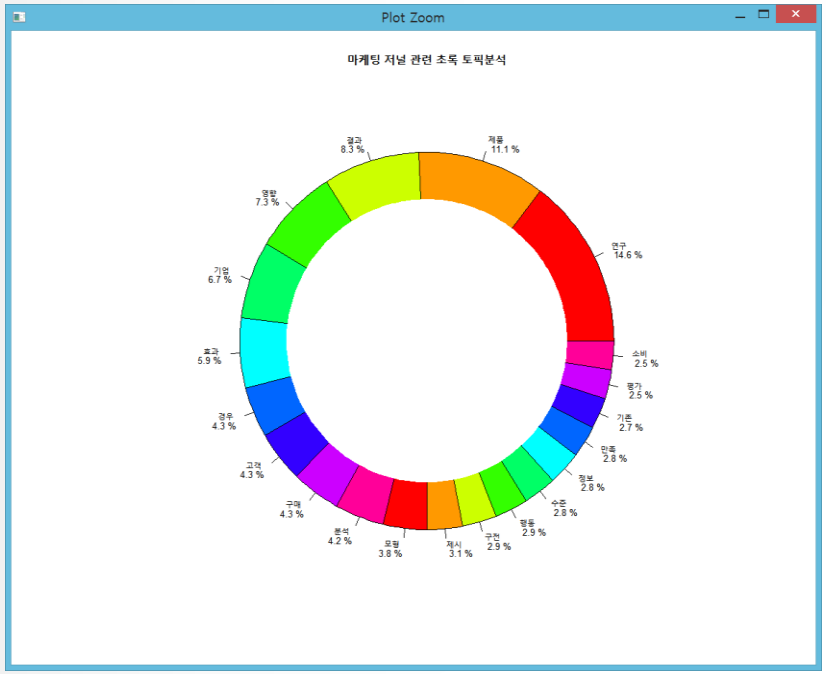
```
word <- head(sort(wordv, decreasing=T), 20) # 상위 20개 토픽추출
pie(word, col=rainbow(10), radius=1) # 파이 차트-무지개색, 원크기
pct <- round(word/sum(word)*100, 1) # 백분율
names(word)
# 키워드와 백분율 적용
lab <- paste(names(word), "₩n", pct, "%")
```



# 8. 정형과 비정형 데이터 처리

● **제목과 백분율 적용 도넛 차트**

```
pie(word, main="마케팅 저널 초록 토픽분석", col=rainbow(10), cex=0.8, labels=lab)
# 링 모양 파이 차트 <- 파이차트 위에 공백 원형 추가
par(new=T) # 차트 추가
pie(word, radius=0.6, col="white", labels=NA, border=NA)
```



# 8. 정형과 비정형 데이터 처리

**##################################################**

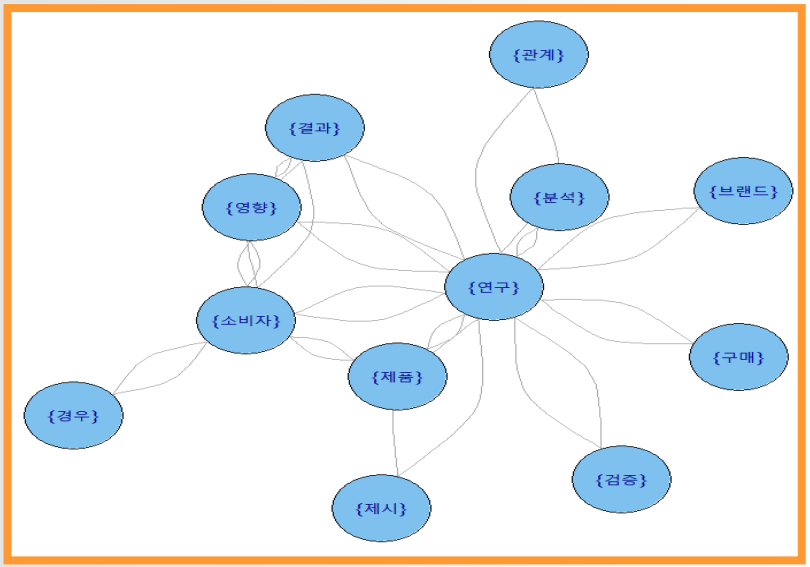
**#  단계2 – 연관어 분석(단어 연관성)  #**

**#  √ 시각화 : 연관어 네트워크 시각화와 근접중심성  #**

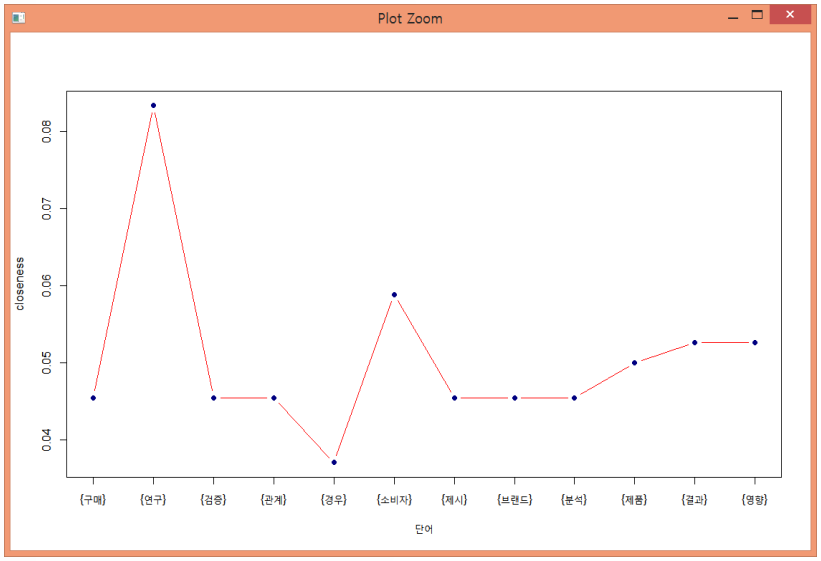
**##################################################**

# 8. 정형과 비정형 데이터 처리

## 단계2 : 연관어 분석(결과물)




연관어 분석 시각화




연관어 중요도(중심성) 시각화

# 8. 정형과 비정형 데이터 처리

- **연관어 분석을 위한 패키지 설치**

```
install.packages("rJava")
Sys.setenv(JAVA_HOME='C:₩₩Program Files₩₩Java₩₩jre1.8.0_31')
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)
```

# 8. 정형과 비정형 데이터 처리

## 1. 데이터셋(abstract.txt) 가져오기

```
f <- file("C:/Rwork/Part-II/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

# 8. 정형과 비정형 데이터 처리

## 2. 단어추출 및 단어트랜잭션생성

```
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
```

# 8. 정형과 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
```

# 8. 정형과 비정형 데이터 처리

- **연관규칙에 의한 연관어 결과**

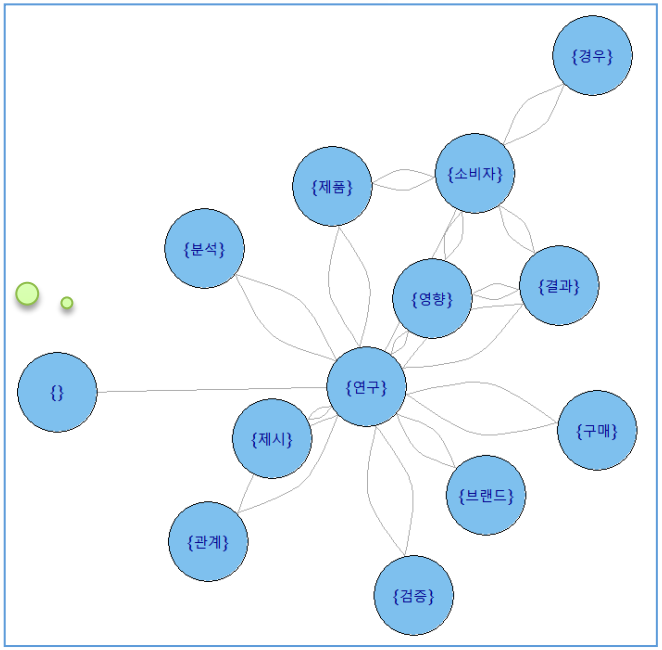
```
      [,1]       [,2]
 [1,] "{}"       "{사회}"
 [2,] "{}"       "{상호작용}"
 [3,] "{}"       "{감정}"
 [4,] "{}"       "{선택}"
 [5,] "{}"       "{하기}"
 [6,] "{}"       "{지각}"
 [7,] "{}"       "{고객}"
 [8,] "{}"       "{정보}"
 [9,] "{}"       "{확인}"
[10,] "{}"       "{이용}"
[11,] "{}"       "{만족}"
[12,] "{}"       "{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}"
```

# 8. 정형과 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**



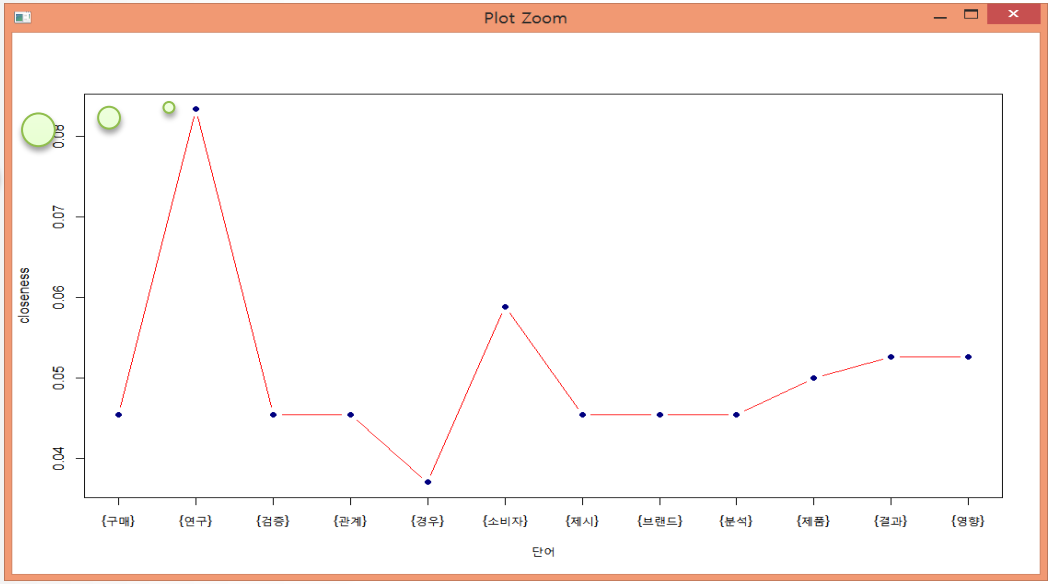
# 8. 정형과 비정형 데이터 처리

## 5. 단어 근접 중심성(closeness centrality) 분석

```
closen <- closeness(ruleg) #closen <- closen[-1] # {} 항목제거
closen
plot(closen, col="red",xaxt="n", lty="solid", type="b", xlab="단어", ylab="closeness")
points(closen, pch=16, col="navy")
axis(1, seq(1, length(closen)), V(ruleg)$name, cex=5)
```

**# 중심성 : 노드(node)의 상대적 중요성을 나타내는 척도**

**[연구] 단어 중심성 높음**



# 8. 정형과 비정형 데이터 처리

<연습문제> hometex.txt 데이터셋을 대상으로 단어 근접 중심성 분석을 수행하시오.

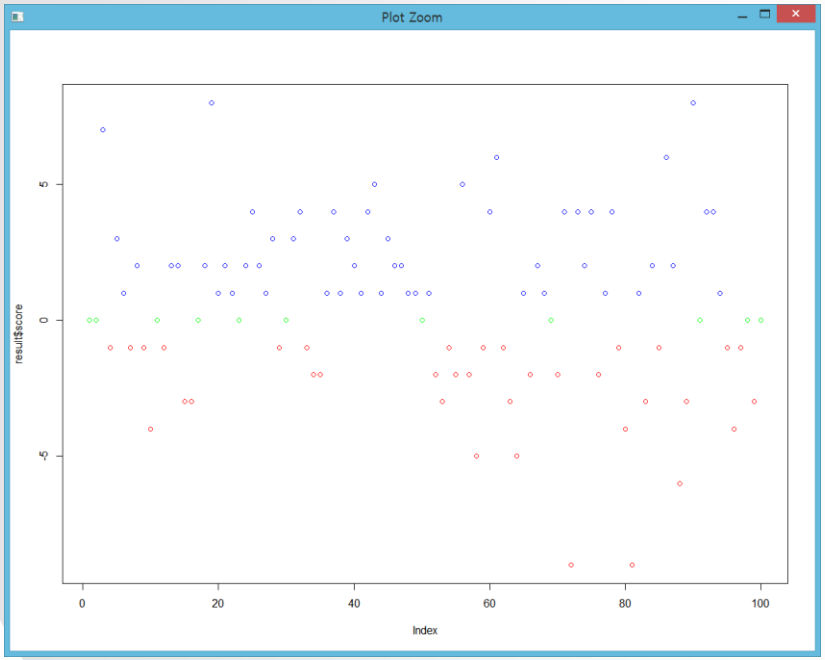
```
# hometex.txt 데이터 셋 가져오기
f <- file("c:/Rwork/Part-II/hometax.txt", encoding="UTF-8")
fl <- readLines(f)
close(f)
head(fl, 10)
```

# 8. 정형과 비정형 데이터 처리

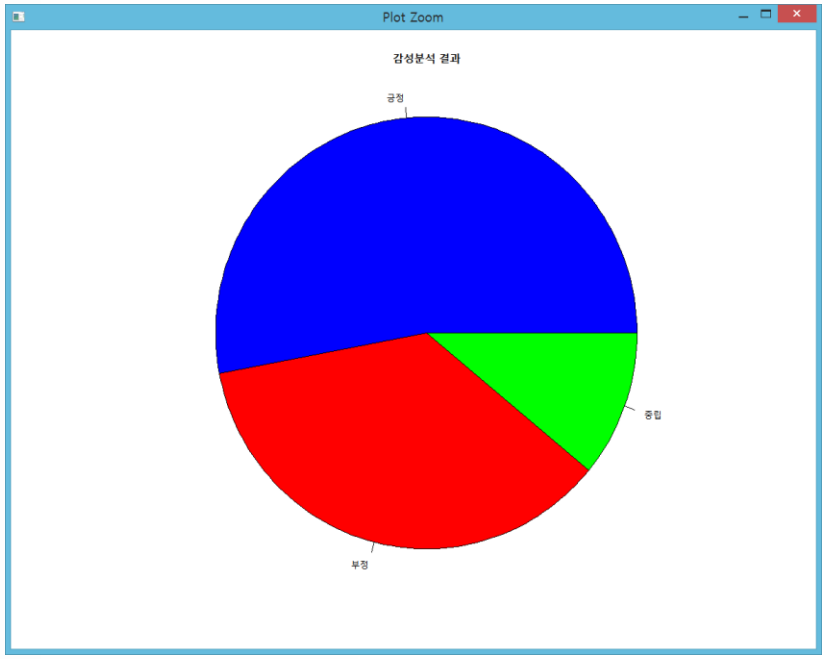
```
##################################################
#     단계3 - 감성 분석(단어의 긍정/부정 분석)                  #
#          √ 시각화 : 긍정(파랑)/ 부정(빨강) -> 불만고객 시각화  #
##################################################
```

# 8. 정형과 비정형 데이터 처리

## 단계3 : 감성 분석(결과물)




긍정/부정 고객 산포도 시각화




긍정/부정(불만) 파이 차트 시각화

# 8. 정형과 비정형 데이터 처리

## 1. 데이터 가져오기

```
data<-read.csv(file.choose()) # file.choose() 파일 선택
head(data,2)
dim(data) # 100    2
str(data) # company, review

# reviews-1행 2열 문장만 추출
sentence <- as.character(data[1,2])
sentence
```

# 8. 정형과 비정형 데이터 처리

## 2. 단어 사전에 단어추가

```
# 긍정단어와 부정단어를 카운터하여 긍정/부정 테스트로 결론
# 긍정단어 - 부정단어 = 0
# 감성분석 - 한글은 공개된 것이 없음

# 긍정어와 부정어 사전 가져오기
setwd("C:/Rwork/Part-II")
pos <- readLines("pos.txt") #pos.txt : 긍정어 사전
neg <- readLines("neg.txt") #neg.txt : 부정어 사전
length(pos) # 2006
length(neg) # 4783

# 사전에 단어 추가 -> 새로운 객체 생성
pos.final<-c(pos, 'upgrade')
neg.final<-c(neg, 'wait')
pos.final; neg.final # 마지막에 단어 추가 됨
```

```
[4777] "zap"                "zapped"
[4779] "zaps"               "zealot"
[4781] "zealous"            "zealously"
[4783] "zombie"             "wait"
```

# 8. 정형과 비정형 데이터 처리

### 3. 감성 분석 처리 함수 정의

```
score.sentiment = function(sentences, pos.words, neg.words, .progress='none'){
  require(plyr)
  require(stringr)

  scores = laply(sentences, function(sentence, pos.words, neg.words) {
    sentence = gsub('[[:punct:]]', '', sentence) #문장부호 제거
    sentence = gsub('[[:cntrl:]]', '', sentence) #특수문자 제거
    sentence = gsub('₩₩d+', '', sentence) # 숫자 제거
    sentence = tolower(sentence) # 소문자 변경

    word.list = str_split(sentence, '₩₩s+') # ₩₩s+ : 공백 정규식, +(1개 이상)
    words = unlist(word.list) # unlist() : 벡터로 변경
    pos.matches = match(words, pos.words) # word의 단어를 pos.words에서 match
    neg.matches = match(words, neg.words)
    pos.matches = !is.na(pos.matches)
    neg.matches = !is.na(neg.matches)
    score = sum(pos.matches) - sum(neg.matches) # 긍정 - 부정
    return(score)
  }, pos.words, neg.words, .progress=.progress )
  scores.df = data.frame(score=scores, text=sentences)
  return(scores.df)
}
```

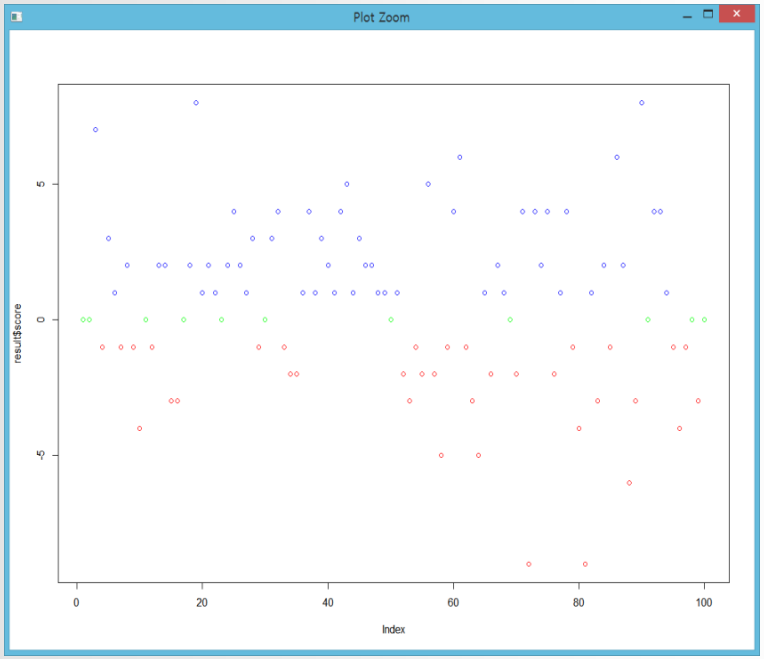
# 8. 정형과 비정형 데이터 처리

## 4. 감성 분석 및 시각화

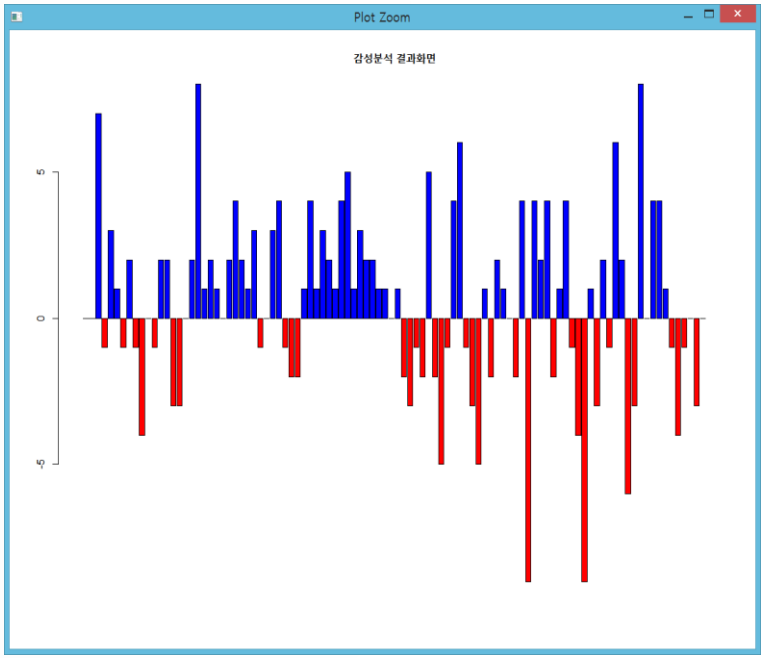
```
#감성 분석 : 두번째 변수(review) 전체 레코드 대상 감성분석
result<-score.sentiment(data[,2], pos.final, neg.final)
names(result) # "score"  "text"
dim(result) # 100    2
result$score

# score값을 대상으로 색 지정
result$color[result$score >=1] <- "blue"
result$color[result$score ==0] <- "green"
result$color[result$score < 0] <- "red"

# 감성분석 결과 차트보기
plot(result$score, col=result$color) # 산포도 색생 적용
barplot(result$score, col=result$color, main ="감성분석 결과화면") # 막대차트
```

# 8. 정형과 비정형 데이터 처리

- 감성 분석 및 시각화




#산포도 시각화




# 막대차트 시각화

# 8. 정형과 비정형 데이터 처리

## 5. 단어의 긍정/부정 분석

```
# 감성분석 빈도수
table(result$color)
#blue green   red
#53    11    36

result$remark[result$score >=1] <- "긍정"
result$remark[result$score ==0] <- "중립"
result$remark[result$score < 0] <- "부정"

sentiment_result<- table(result$remark)
# 제목, 색상, 원크기
pie(sentiment_result, main="감성분석 결과",
    col=c("blue","red","green"), radius=0.8)
```