R교재 ch8 고급시각화

1. R의 고급 시각화 도구

주요 패키지: graphics, lattice, ggplot2, ggmap
graphics패키지는 일반적인 시각화 도구 제공. barplot(), dotchart(), pie(), boxplot(), hist(), plot()함수

고급 시각화
lattice 패키지: 서로 상관있는 확률적 반응변수의 시각화에 사용. 특정 변수가 갖는 범주(domain)별로 독립된 패널을 격자(lattice)처럼 배치하여 여러 개의 변수에 대한 범주를 세부적으로 시각화해주는 도구

ggplot2 패키지: 기하학적 객체들(점, 선, 막대 등)에 미적 특성(색상, 모양, 크기)을 적용하여 시각화는 방법을 제공. 그래프와 사용자 간의 상호작용(interaction)기능을 제공하기 때문에 시각화 과정에서 코드의 재사용성이 뛰어나고, 데이터 분석을 위한 시각화에 적합

ggmap: 지도를 기반으로 위치, 영역, 시간과 공간에 따른 차이와 변화에 대한 것을 다루는 공간 시각화에 적합.

2. 격자형 기법 시각화

격자 형태의 그래픽(trellis graphic)을 생성하여 시각화하는 기법.
한번에 여러 개의 차트 생성이 가능하고, 높은 밀도의 차트를 효과적으로 그려주는 이점

lattice패키지

[표 8.1] lattice 패키지의 주요 함수
histogram(): 연속형 변수 대상 히스토그램, cf. hist()
densityplot(): 연속형 변수 대상 밀도 그래프
barchart(): 막대그래프, cf. barplot()
dotplot(): 점그래프, cf. dotchart()
xyplot(): 교차그래프, cf. plot()
equal.count(): 데이터 셋을 지정한 영역만큼 범주화
coplot(): 조건변수 조작으로 조건그래프
cloud(): 3차원 산점도 그래프

https://www.rdocumentation.org/packages/lattice/versions/0.20-44

실습 (lattice 패키지 사용 준비하기)

1단계: lattice 패키지 설치

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

2단계: 실습용 데이터 가져오기

```
install.packages("mlmRev")
library(mlmRev)
data("Chem97")
str(Chem97)
head(Chem97, 30)
Chem97
```

mlmRey 패키지
Chem97 데이터

더 알아보기 (Chem97 데이터 셋)
https://www.rdocumentation.org/packages/mlmRev/versions/1.0-8/topics/Chem97

2.1 히스토그램

lattice패키지, histogram()함수 이용
https://www.rdocumentation.org/packages/lattice/versions/0.12-10/topics/histogram

실습 (histogram()함수를 이용하여 데이터 시각화)

```
histogram(~gcsescore, data = Chem97)
```

실습 (score 변수를 조건변수로 지정하여 데이터 시각화하기)

```
histogram(~gcsescore | score, data = Chem97)
histogram(~gcsescore | factor(score), data = Chem97)
```

2.2 밀도 그래프

밀도 그래프를 그리기 위한 densityplot()함수 이용
형식: densityplot(~x축 컬럼 | 조건, data, group=변수)
https://www.rdocumentation.org/packages/lattice/versions/0.3-1/topics/densityplot

실습 (densityplot()함수를 사용하여 밀도 그래프 그리기)

```
densityplot(~gcsescore | factor(score), data = Chem97,
            groups = gender, plot.Points = T,
            auto.ley = T)
```

densityplot()함수 속성
plot.points = T: 밀도 점 표시 여부(밀도 점을 표시하지 않는 경우의 plotpoints = F)
auto.key = T: 범례 표시 여부(범례는 그래프 상단에 표시)

2.3 막대 그래프

barchart()함수
형식: barchart(y축 컬럼 ~ x축 컬럼 | 조건, data, layout)
https://www.rdocumentation.org/packages/lattice/versions/0.3-1/topics/barchart

실습: barchart() 함수를 사용하여 막대 그래프 그리기

1단계: 기본 데이터 셋 가져오기

```
data(VADeaths)
VADeaths
```

VAdeaths 데이터

2단계: VAdeaths 데이터 셋 구조보기

```
str(VADeaths)
class(VADeaths)
mode(VADeaths)
```

3단계: 데이터 형식 변경(matrix형식을 table 형식으로 변경)

```
dft <- as.data.frame.table(VADeaths)
str(dft)
dft
```

통계처리를 위하여 matrix 자료구조를 table 자료구조로 변환
table구조로 변경하면 가장 왼쪽의 첫번째 컬럼을 기준으로 '넓은 형식'의 자료가 '긴 형식'의 자료로 구조가 변경된다.

as.data.frame.table()함수: 넓은 형식의 데이터를 긴 형식으로 변경
https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/as.data.frame
https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/table

4단계: 막대 그래프 그리기

```
barchart(Var1 ~ Freq | Var2, data = dft, layout = c(4, 1))
```

"layout_c(4,1)" 속성: 4개의 패널을 한줄에 표시

5단계: origin 속성을 사용하여 막대 그래프 그리기

barchart(Var1 ~ Freq | Var2, data = dft, layout = c(4, 1), origin = 0)

"origin = 0" 속성: x축의 구간을 0부터 표시해주는 역할

2.4 점 그래프

dotplot()함수
형식: dotplot(y축 컬럼 ~ x축 컬럼 |조건, data. layout)
https://www.rdocumentation.org/packages/lattice/versions/0.3-1/topics/dotplot

실습 (dotplot()함수를 사용하여 점 그래프 그리기)

1단계: layout 속성이 없는 경우

dotplot(Var1 ~ Freq | Var2, dft)

2단계: layout 속성을 적용한 경우

dotplot(Var1 ~ Freq | Var2, dft, layout = c(4, 1))

실습 (점을 선으로 연결하여 시각화히기)

```
dotplot(Var1 ~ Freq, data = dft,
        groups = Var2, type = "o",
        auto.key = list(space = "right", points = T, lines = T))
```

dotplot() 함수의 속성
type='0': 점(point)타입으로 원형에 실선이 통과하는 유형으로 그래프의 타입을 지정
auto.key=list(space="right", points=T, lines=T): 범례를 타나내는 속성. 범례의 위치는 오른쪽으로 지정, 점과 선을 범례에 표시

2.5 산점도 그래프

xyplot()함수
형식: xyplot(y축 컬럼 ~ x축 컬럼|조건변수, data=data.frame 또는 list, layout)

실습 (airquality 데이터 셋으로 산점도 그래프 그리기)

1단계: airquality 데이터 셋 가져오기

```
library(datasets)
str(airquality)
```

더 알아보기 (airquality 데이터 셋)
https://www.rdocumentation.org/packages/datasets/versions/3.6.2/topics/airquality

2단계: xyplot()함수를 사용하여 산점도 그리기

```
xyplot(Ozone ~ Wind, data = airquality)
```

https://www.rdocumentation.org/packages/lattice/versions/0.10-10/topics/xyplot

3단계: 조건변수를 사용하는 xyplot()함수로 산점도 그리기

```
xyplot(Ozone ~ Wind | Month, data = airquality)
```

Month가 조건변수로 지정

4단계: 조건변수와 layout속성을 사용하는 xyplot()함수로 산점도 그리기

```
xyplot(Ozone ~ Wind | Month, data = airquality, layout = c(5, 1))
```

5단계: Month변수를 factor타입으로 변환하여 산점도 그리기 (패널 제목에는 factor값을 표시)

```
convert <- transform(airquality, Month = factor(Month))
```

```
str(convert)

xyplot(Ozone ~ Wind | Month, data = convert)
#xyplot(Ozone ~ Wind | Month, data = convert, layout = c(5, 1))
```

실습 (quakes 데이터 셋으로 산점도 그래프 그리기)

1단계: quakes 데이터 셋 보기

```
head(quakes)
str(quakes)
```

더 알아보기 (quakes 데이터 셋)
https://www.rdocumentation.org/packages/datasets/versions/3.6.2/topics/quakes

2단계: 지진 발생 진앙지(위도와 경도) 산점도 그리기

```
xyplot(lat ~ long, data = quakes, pch = ".")
```

3단계: 산점도 그래프를 변수에 저장하고, 제목 문자열 추가하기

```
tplot <- xyplot(lat ~ long, data = quakes, pch = ".")
tplot <- update(tplot, main = "1964년 이후 태평양에서 발생한 지진 위치")
print(tplot)
```

실습 (이산형 변수를 조건으로 지정하여 산점도 그리기)
수심별로 진앙지를 파악하기 위해서 depth변수(이산형 변수)를 조건으로 지정

1단계: depth 변수의 범위 확인

```
range(quakes$depth)
```

2단계: depth 변수 리코딩: 6개의 범주(100단위)로 코딩 변경

```
quakes$depth2[quakes$depth >= 40 & quakes$depth <= 150] <- 1
quakes$depth2[quakes$depth >= 151 & quakes$depth <= 250] <- 2
quakes$depth2[quakes$depth >= 251 & quakes$depth <= 350] <- 3
quakes$depth2[quakes$depth >= 351 & quakes$depth <= 450] <- 4
quakes$depth2[quakes$depth >= 451 & quakes$depth <= 550] <- 5
quakes$depth2[quakes$depth >= 551 & quakes$depth <= 680] <- 6
```

3단계: 리코딩된 변수(depth2)를 조건으로 산점도 그리기

```
convert <- transform(quakes, depth2 = factor(depth2))
xyplot(lat ~ long | depth2, data = convert)
```

실습 (동일한 패널에 두 개의 변수값 표현하기)

```
xyplot(Ozone + Solar.R ~ Wind | factor(Month),
        data = airquality,
        col = c("blue", "red"),
        layout = c(5, 1))
```

두개의 변수값을 표현하기 위한 xyplot()함수
형식: xyplot(y1축 + y2축 ~ x축 | 조건, data, type, layout)

2.6 데이터 범주화

데이터를 일정구간으로 범주화 작업

lattice 패키지의 equal.count()함수 이용
형식: equal.count(data, number=n, overlap =0)

실습 (equal.count()함수를 사용하여 이산형 변수 범주화하기)

1단계: 1~150 을 대상으로 겹치지 않게 4개 영역으로 범주화

```
numgroup <- equal.count(1:150, number = 4, overlap = 0)
numgroup
```

2단계: 지진의 깊이를 5개 영역으로 범주화

```
depthgroup <- equal.count(quakes$depth, number = 5, overlap = 0)
depthgroup
```

3단계: 범주화된 변수(depthgroup)를 조건으로 산점도 그리기

```
xyplot(lat ~ long | depthgroup, data = quakes,
       main = "Fiji Earthquakes(depthgroup)",
       ylab = "latitude", xlab = "longitude",
       pch = "@", col = "red")
```

xyplot()함수의 주요 속성
main: 차트 제목
ylab: y축 이름
xlab: x축 이름
pch="@":: 점 표현문자
col='red': 점 색상

실습 (수심과 리히터 규모 변수를 동시에 적용하여 산점도 그리기

1단계: 리히터 규모를 2개 영역으로 구분

```
magnitudegroup <- equal.count(quakes$mag, number = 2, overlap =0)
magnitudegroup
```

2단계: margintudegroup변수를 기준으로 산점도 그리기

```
xyplot(lat ~ long | magnitudegroup, data = quakes,
        main = "Fiji Earthquakes(magnitude)",
        ylab = "latitude", xlab = "longitude",
        pch = "@", col = "blue")
```

3단계: 수심과 리히터 규모를 동시에 표현(2행 5열 패널 구현)

```
xyplot(lat ~ long | depthgroup * magnitudegroup, data = quakes,
        main = "Fiji Earthquakes",
        ylab = "latitude", xlab = "longitude",
        pch = "@", col = c("red", "blue"))
```

같은 패널에 수심과 리히터 규모가 동시에 표현되기 때문에 이를 구분하기 위해 2개의 색상으로 구분

[표 8.2] 수심과 리히터 규모에 의해서 생성된 각 패널의 범주 값

		1열	2열	3열	4열	5열
1행	수심	39.5-139.5	79.5-186.5	185.5-397.5	396.5-562.5	562.5-680.5
	리히터	3.95-4.65				
2행	수심	39.5-139.5	79.5-186.5	185.5-397.5	396.5-562.5	562.5-680.5
	리히터	4.55-6.45				

각 패널에 수심(depth)변수와 mag(리히터 규모)변수의 범주 값을 factor형으로 변환하여 산점도 그래프를 그리면 각 패널에 그려진 진앙의 수심과 리히터 규모의 범주 값을 쉽게 판단할 수 있다. 하지만 수심(depth)변수와 mag(리히터 규모)변수는 연속형 변수이기 때문에 이들을 이산형 변수로 변경한후 factor형으로 변환하여 산점도를 그린다.

실습 (이산형 변수를 리코딩한 뒤에 factor형으로 변환하여 산점도 그리기)

1단계: depth변수 리코딩

```
quakes$depth3[quakes$depth >= 39.5 & quakes$depth <= 80.5] <- 'd1'
quakes$depth3[quakes$depth >= 79.5 & quakes$depth <= 186.5] <- 'd2'
quakes$depth3[quakes$depth >= 185.5 & quakes$depth <= 397.5] <- 'd3'
quakes$depth3[quakes$depth >= 396.5 & quakes$depth <= 562.5] <- 'd4'
quakes$depth3[quakes$depth >= 562.5 & quakes$depth <= 680.5] <- 'd5'
```

2단계: mag변수 리코딩

```
quakes$mag3[quakes$mag >= 3.95 & quakes$mag <= 4.65] <- 'm1'
quakes$mag3[quakes$mag >= 4.55 & quakes$mag <= 6.65] <- 'm2'
```

3단계: factor형 변환

```
convert <- transform(quakes,
                     depth3 = factor(depth3),
                     mag3 = factor(mag3))
```

transform()함수
https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/transform

4단계: 산점도 그래프 그리기

```
xyplot(lat ~ long | depth3 * mag3, data = convert,
       main = "Fiji Earthquakes",
       ylab = "latitude", xlab = "longitude",
       pch = "@", col = c("red", "blue"))
```

2.7 조건 그래프

조건 a에 의해서 x에 대한 y의 그래프를 그린다.
조건으로 지정된 변수의 값을 일정한 구간으로 범주화하여 조건 그래프를 그린다.

coplot()함수
형식: coplot(y축 컬럼 ~ x축 컬럼 | 조건 컬럼, data)
https://www.rdocumentation.org/packages/graphics/versions/3.6.2/topics/coplot

실습 (depth 조건에 의해서 위도와 경도의 조건 그래프 그리기)

```
coplot(lat ~ long | depth, data = quakes)
```

실습 (조건의 구간 크기와 겹침 간격 적용 후 조건 그래프 그리기)

1단계: 조건의 구간 막대기가 0, 1 단위로 겹쳐 범주화

```
coplot(lat ~ long | depth, dat = quakes,
       overlap = 0.1)
```

2단계: 조건 구간을 5개로 지정하고, 1행 5열의 패널로 조건 그래프 작성

```
coplot(lat ~ long | depth, data = quakes,
       number = 5, row = 1)
```

실습 (패널과 조건 막대에 색을 적용하여 조건 그래프 그리기)

1단계: 패널 영역에 부드러운 곡선 추가

```
coplot(lat ~ long | depth, data = quakes,
       number = 5, row = 1,
       panel = panel.smooth)
```

2단계: 패널 영역과 조건 막대에 색상 적용

```
coplot(lat ~ long | depth, data = quakes,
```

```
        number = 5, row = 1,
        col = 'blue',
        bar.bg = c(num = 'green'))
```

2.8 3차원 산점도 그래프

cloud()함수
형식: cloud(z축변수 ~ y축변수 *x축 변수, data)
https://www.rdocumentation.org/packages/lattice/versions/0.5-7/topics/cloud

실습 (위도, 경도, 깊이를 이용하여 3차원 산점도 그리기)

```
cloud(depth ~ lat * long, data = quakes,
      zlim = rev(range(quakes$depth)),
      xlab = "경도", ylab = "위도", zlab = "깊이")
```

실습 (테두리와 회전 속성을 추가하여 3차원 산점도 그래프 그리기)

```
cloud(depth ~ lat * long, data = quakes,
      zlim = rev(range(quakes$depth)),
      panel.aspect = 0.9,
      screen = list(z = 45, x = -25),
      xlab = "경도", ylab = "위도", zlab = "깊이")
```

cloud()함수의 주요 속성
panel.aspect: 테두리 사이즈
screen=list(z=45, x=-25): z축과 x축 회전

3. 기하학적 기법 시각화

일반 그래픽 관련 패키지: 이미지 표현에 중점
ggplot2: 기하학적 객체(점, 선, 막대 등)에 미적 특성(색상, 모양, 크기)을 연결. 데이터 객체와 그래픽 객체를 서로 분리하고 재사용 가능.

[표 8.3] ggplot2패키지의 주요 함수
qplot(), ggplot(), ggsave()

3.1 qplot()함수

기하학적 객체(점, 선, 다각형 등)에 미적 특성(색상, 모양, 크기)을 매핑하여 그래프에 그려주는 ggplot2패키지에서 제공 함수

실습 (ggplot2 패키지설치와 실습 데이터 가져오기)

```
install.packages("ggplot2")
library(ggplot2)
data(mpg)
str(mpg)
head(mpg)
summary(mpg)

table(mpg$drv)
```

mpg데이터 사용

더 알아보기 (mpg 데이터 셋)
https://archive.ics.uci.edu/ml/datasets/auto+mpg

(1) 한 개 변수 대상으로 qplot() 함수 적용

qplot()함수
형식: qplot(x축 ~ y축, data, facet, geom., stat, position, xlim, ylim, log, main, xlab, ylab, asp)
https://www.rdocumentation.org/packages/ggplot2/versions/3.3.3/topics/qplot

실습 (qplot()함수의 fill과 binwidth 속성 적용)

1단계: 도수 분포를 세로 막대 그래프로 표현

qplot(hwy, data = mpg)

2단계: fill속성 적용

qplot(hwy, data = mpg, fill = drv)

3단계: binwidth 속성 적용

qplot(hwy, data = mpg, fill = drv, binwidth = 2)

실습 (facets속성을 사용하여 drv변수값으로 행/열 단위로 패널 생성)

1단계: 열 단위 패널 생성

qplot(hwy, data = mpg, fill = drv, facets = . ~ drv, binwidth = 2)

'facets = .~dry' 속성: 컬럼 단위로 패널이 생성되어 그래프를 그린다.

2단계: 행 단위 패널 생성

qplot(hwy, data = mpg, fill = drv, facets = drv ~ ., binwidth = 2)

'facets = dry~.' 속성: 행 단위로 패널이 생성되어 그래프를 그린다.

(2) 두개 변수 대상으로 qplot()함수 적용

실습 (qplot()함수에서 color속성을 사용하여 두 변수 구분하기)

1단계: 두변수로 displ과 hwy변수 사용

```
qplot(displ, hwy, data = mpg)
```

2단계: 두 변수로 displ과 hwy변수 사용하며 dry변수에 색상 적용

```
qplot(displ, hwy, data = mpg, color = drv)
```

실습 (displ과 hwy변수의 관계를 driv변수로 구분하기)

```
qplot(displ, hwy, data = mpg, color = drv, facets = . ~ drv)
```

(3) 미적요소 맵핑(mapping)

실습 (mtcars 데이터 셋에 색상, 크기, 모양 적용하기)

1단계: 실습용 데이터 셋 확인하기

```
head(mtcars)
```

더 알아보기 (mtcars 데이터 셋)
https://www.rdocumentation.org/packages/datasets/versions/3.6.2/topics/mtcars

2단계: 색상 적용

```
qplot(wt, mpg, data = mtcars, color = factor(carb))
```

3단계: 크기 적용

```
qplot(wt, mpg, dat = mtcars,
      size = qsec, color = factor(carb))
```

4단계: 모양 적용

```
qplot(wt, mpg, data = mtcars,
```

```
        size = qsec, color = factor(carb), shape = factor(cyl))
```

(4) 기하하적 객체 적용

geom속성 이용

실습 (diamonds 데이터 셋에 막대, 점, 선 레이아웃 적용하기)

1단계: 실습용 데이터 셋 확인하기

```
head(diamonds)
```

더 알아보기 (diamonds 데이터 셋)
https://www.rdocumentation.org/packages/ggplot2/versions/3.3.5/topics/diamonds

2단계: geom속성과 fill속성 사용

```
qplot(clarity, data = diamonds, fill = cut, geom = "bar")
```

3단계: 테두리 색 사용

```
qplot(clarity, data = diamonds, colour = cut, geom = "bar")
```

4단계: geom.="point"속성으로 산점도 그래프 그리기

```
qplot(wt, mpg, data = mtcars, size = qsec, geom = "point")
```

5단계: 산점도 그래프에 cyl변수의 요인으로 포인트 크기 적용 & carb변수의 요인으로 포인트 색 적용

```
qplot(wt, mpg, data = mtcars, size = factor(cyl),
        color = factor(carb), geom = "point")
```

6단계: 산점도 그래프에 qsec변수의 요인으로 포인트 크기 적용 & cyle변수의 요인으로 포인트 모양 적용

```
qplot(wt, mpg, data = mtcars, size = qsec,
        color = factor(carb),
        shape = factor(cyl), geom = "point")
```

7단계: geom.="smooth"속성으로 산점도 그래프에 평활 그리기

```
qplot(wt, mpg, data = mtcars,
        geom = c("point", "smooth"))
```

8단계: 산점도 그래프의평활에 cyl변수의 요인으로 색상 적용하기

```
qplot(wt, mpg, dat = mtcars, color = factor(cyl),
        geom = c("point", "smooth"))
```

9단계: geom.="line" 속성으로 그래프 그리기

```
qplot(mpg, wt, data = mtcars,
        color = factor(cyl), geom = "line")
```

10단계: geom.=c("point", "line")속성으로 그래프 그리기

```
qplot(mpg, wt, data = mtcars,
        color = factor(cyl), geom = c("point", "line"))
```

3.2 ggplot()함수

ggplot()함수는 데이터 속성에 미적요소를 매핑한 후 스케일링과정을 거쳐서 생성된 객체를 + 연산자를 사용하여 미적 요소 맵핑을 새로운 레이어에 상속받아 재사용할 수 있도록 지원하는 ggplot2 패키지의 함수

(1) 미적 요소 맵핑

aes()함수를 이용하여 미적요소만 별도 지정 가능
https://www.rdocumentation.org/packages/ggplot2/versions/3.3.3/topics/aes

실습 (aes()함수 속성을 추가하여 미적 요소 맵핑하기)

1단계: diamonds데이터 셋에 미적 요소 맵핑

```
p <- ggplot(diamonds, aes(carat, price, color = cut))
p + geom_point()
```

2단계: mtcars데이터 셋에 미적 요소 맵핑

```
p <- ggplot(mtcars, aes(mpg, wt, color = factor(cyl)))
p + geom_point()
```

(2) 기하학적 객체 적용

geom.()함수는 ggplot()함수에서 정의된 미적요소 맵핑 객체를 상속받아서 서로 다른 레이어에서 별도로 재사용 할 수 있다.
https://www.rdocumentation.org/packages/ggplot2/versions/3.3.3/topics/ggplot

실습(geom_line()과 geom._point()함수를 적용하여 레이어 추가)

1단계: geom._line()레이어 추가

```
p <- ggplot(mtcars, aes(mpg, wt, color = factor(cyl)))
```

```
p + geom_line()
```

2단계: geom._point레이어 추가

```
p <- ggplot(mtcars, aes(mpg, wt, color = factor(cyl)))
p + geom_point()
```

(3) 미적 요소 맵핑과 기하학적 객체 적용

stat_bin()함수는 aes()함수와 geom.()함수의 기능을 동시에 제공

실습 (stat_bin()함수를 사용하여 막대 그래프 그리기)

1단계: 기본 미적 요소 맵핑 객체를 생성한 뒤에 stat_bin()함수 사용

```
p <- ggplot(diamonds, aes(price))
p + stat_bin(aes(fill = cut), geom = "bar")
```

2단계: price빈도를 밀도(전체의 합=1)로 스케일링하여 stat_bin()함수 사용

```
p + stat_bin(aes(fill = ..density..), geom = "bar")
```

더 알아보기 (ggplot()함수에 의해서 그래프가 그려지는 절차)

ggplot()함수는 다음의 순서에 따라 그래프를 그린다.
1) 미적요소 맵핑(aes): x 축, y 축, color속성
2) 통계적인 변환(stat): 통계계산 작업
3) 기하학적 객체 적용(geom): 차트 유형
4) 위치조정(position, adjustment), 채우기(fill), 스택(stack), 닷지(dodge) 유형

실습 (stat_bin()함수 적용 영역과 산점도 그래프 그리기)

1단계: stat_bin()함수 적용 영역 나타내기

```
p <- ggplot(diamonds, aes(price))
p + stat_bin(aes(fill = cut), geom = "area")
```

2단계: stat_bin()함수로 산점도 그래프 그리기

```
p + stat_bin(aes(color = cut,
                 size = ..density..), geom = "point")
```

(4) 산점도와 회귀선 적용

회귀선을 시각화하기 위해
geom_count()함수: 숫자의 크기에 따라서 산점도를 시각화
geom_smooth()함수: 속성으로 method="lm"을 지정하면 회귀선과 보조선으로 시각화 사용.

실습 (산점도에 회귀선 적용하기)

```
library(UsingR)
data("galton")
p <- ggplot(data = galton, aes(x = parent, y = child))
p + geom_count() + geom_smooth(method ="lm")
```

(5) 테마(Theme) 적용

테마: 그래프의 외형 지정

실습 (테마를 적용하여 그래프 외형 속성 설정)

1단계: 제목을 설정한 산점도 그래프

```
p <- ggplot(diamonds, aes(carat, price, color = cut))
p <- p + geom_point() + ggtitle("다이아몬드 무게와 가격의 상관관계")
print(p)
```

2단계: theme()함수를 이용하여 그래프의 외형 속성 적용

```
p + theme(
   title = element_text(color = "blue", size = 25),
   axis.title = element_text(size = 14, face ="bold"),
   axis.title.x = element_text(color = "green"),
   axis.title.y = element_text(color = "green"),
   axis.text = element_text(size = 14),
   axis.text.y = element_text(color = "red"),
   axis.text.x = element_text(color = "purple"),
   legend.title = element_text(size = 20,
                               face = "bold",
                               color = "red"),
   legend.position = "bottom",
   legend.direction = "horizontal"
)
```

theme()함수

https://www.rdocumentation.org/packages/ggplot2/versions/3.3.3/topics/theme

3.3 ggsave()함수

그래프를 pdf 또는 이미지 형식(jpg, png 등)의 파일로 저장
이미지의 해상도, 폭, 너비 지정 가능

실습 (그래프를 이미지 파일로 저장하기)

1단계: 저장할 그리기

```
p <- ggplot(diamonds, aes(carat, price, color = cut))
p + geom_point()
```

2단계: 가장 최근에 그려진 그래프 저장

```
ggsave(file = "C:/Rwork/output/diamond_price.pdf")
ggsave(file = "C:/Rwork/output.diamond_price.jpg", dp = 72)
```

실습 (변수에 저장된 그래프를 이미지 파일로 저장하기)

```
p <- ggplot(diamonds, aes(clarity))
p <- p + geom_bar(aes(fill = cut), position = "fill")
ggsave(file = "C:/Rwork/output/bar.png",
       plot = p, width = 10, height = 5)
```

4. 지도 공간 기법 시각화

ggmap패키지
https://www.rdocumentation.org/packages/ggmap/versions/3.0.0/topics/ggmap

[표 8.4] ggmap패키지의 주요 함수
get_stamenmap(), geocode(), getgooglemap(), get_map(), getnavermap(), ggimage(), ggmap(), ggmapplot(), qmap(), qmplot()

구글(Google)관련 패키지를 이용하여 지도 이미지를 서비스 받기 위해서 관련 함수를 실행하면 오류 메시지

2019년부터 구글 지도 서비스를 받기 위해서는 구글 지도 API 인증키를 발급받아야 하기 때문에 인증키 없이 사용할 수 있는 get_stamenmap()함수만 이용하여 지도 이미지를 시작화

4.1 Stamen Maps API 이용

get_ stamenmap()함수

형식: get_stamenmap(bbox = c(left, bottom, right, top), zoom, maptype, crop, messaging = FALSE, urlonly = FALSE, color = c("color", "bw"), force = FALSE)

주요속성:
bbox: 지도가 그려질 경계 상자
zoom: 확대비율(수치가 작을수록 중심지역 기준으로 확대)
maptype: 지도 유형
crop: 원시 맵 타일을 지정된 경계 상자로 자른다. FALSE인 경우 결과맵이 지정된 경계상자를 덮는다.
messaging: 메시지 켜기/끄기
urlonly: url반환
color: 컬러 또는 흑백
force: 지도가 파일에 있으면 새 지도를 찾아야 할지 여부

https://www.rdocumentation.org/packages/ggmap/versions/3.0.0/topics/get_stamenmap

실습 (지도 관련 패키지 설치)

```
library(ggplot2)
install.packages("ggmap")
library(ggmap)
```

4.2 위도와 경도 중심으로 지도 시각화

실습 (서울을 중심으로 지도 시각화)

1단계: 서울 지역의 중심 좌표 설정

```
seoul <- c(left = 126.77, bottom = 37.40,
           right = 127.17, top = 37.70)
```

2단계: zoom, maptype으로 정적 지도 이미지 가져오기

```
map <- get_stamenmap(seoul, zoom = 12, maptype = 'terrain')
ggmap(map)
```

4.3 지도 이미지에 레이어 적용

실습 (2019년도 1월 대한민국 인구수를 기준으로 지역별 인구수 표시)

1단계: 데이터 셋 가져오기

```
pop <- read.csv(file.choose(), header = T)

library(stringr)

region <- pop$'지역명'
lon <- pop$LON
lat <- pop$LAT
```

```
tot_pop <- as.numeric(str_replace_all(pop$'총인구수', ',', ''))

df <- data.frame(region, lon, lat, tot_pop)
df
df <- df[1:17, ]
df
```

2단계: 정적 지도 이미지 가져오기

```
daegu <- c(left = 123.4423013, bottom = 32.8528306,
           right = 131.601445, top = 38.8714354)
map <- get_stamenmap(daegu, zoom = 7, maptype = 'watercolor')
```

3단계: 지도 시각화하기

```
layer1 <- ggmap(map)
layer1
```

4단계: 포인트 추가

```
layer2 <- layer1 + geom_point(data = df,
                              aes(x = lon, y = lat,
                                  color = factor(tot_pop),
                                  size = factor(tot_pop)))
Layer2
```

5단계: 텍스트 추가

```
layer3 <- layer2 + geom_text(data = df,
                             aes(x = lon + 0.01, y = lat + 0.08,
                                 label = region), size = 3)
Layer3
```

6단계: 크기를 지정하여 파일로 저장

```
ggsave("pop201901.png", scale = 1, width = 10.24, height = 7.68)
```

Ch8 연습문제 풀기