2020년도 1학기 통계프로그래밍언어 연습 문제

## Prerequisites

1. rmarkdown, knitr, tidyverse, readxl 패키지를 설치 하시오.

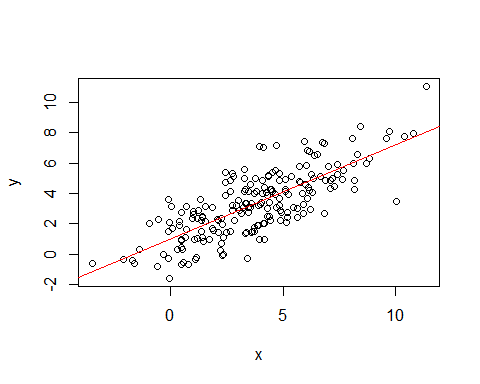
## 기말고사 대비 연습문제

1. 아래 기술된 절차를 수행하기 위한 R 스크립트를 순차적으로 작성 하시오.
   1. 현재 작업공간(workspace)에 저장되어 있는 모든 R 객체(objects)를 확인할 수 있는 스크립트를 작성 하시오.
   2. 현재 작업공간에 저장되어 있는 **모든** R 객체를 삭제하는 스크립트를 작성 하시오.
   3. 현재 작업폴더(working directory)를 확인하는 스크립트를 작성 하시오.
   4. 현재 작업폴더 내 존재하는 파일을 확인하는 스크립트를 작성 하시오.
   5. 작업폴더를 C: 또는 D: 드라이브로 변경하는 스크립트를 작성 하시오.
   6. R 작업공간에서 폴더를 생성하는 함수에 대해 googling을 통해 검색하고, 검색한 함수를 적용해 C 또는 D 드라이브에 final-exam-practice 라는 폴더를 생성하시오.
   7. 작업폴더를 final-exam-practice로 변경 하시오.
   8. R 작업공간에서 폴더를 변경을 위해 절대 폴더(정확한 폴더명, 예: C:/folder\_a/folder\_b)를 입력하는 방법과 현재 작업 폴더를 기준으로 폴더를 변경하게 위해 상대적 폴더를 사용하는 방법이 있다. 상대적 폴더를 지정 및 사용 방법을 예시를 통해 설명 하시오.

ls() #1-a  
rm(list = ls()) #1-b  
getwd()  
dir() #1-d: 또는 list.files()  
setwd("D:/") # 1-e  
dir.create("final-exam-practice") #1-f  
setwd("final-exam-practice") #1-g  
#h. 상대적 경로는 현재 작업하고 있는 폴더(디렉토리) 또는 파일이 컴파일되고 있는 폴더가 기준  
# 현재 작업 폴더에서 상위 폴더(디렉토리)로 가기 위해 ".."을 사용

1. 다음은 R 객체 vector와 matrix 와 관련한 질문이다. 해당 코드를 순차적으로 작성 하시오. 단, 재현성을 위해 set.seed(20200618)을 실행 후 아래 문제에 대해 답하시오.
   1. 평균이 4 이고 분산이 9인 정규분포에서 200개의 난수를 생성 후 x 라는 객체에 저장 하시오.
   2. , 라고 할 때, 기술한 분포와 개수만큼 난수를 생성한 후 e라는 객체에 저장 하시오.
   3. 위에서 생성한 x와 e를 활용해 결과를 y에 저장하기 위한 스크립트를 작성 하시오.
   4. x와 y에 대한 산점도를 생성하는 스크립트를 작성 하시오.
   5. x를 이용해 아래와 같은 행렬 X를 생성하는 스크립트를 작성하시오.
   6. 객체 X와 y를 이용해 의 계산 결과를 bhat에 저장하는 스크립트를 작성하시오
   7. f 에서 구한 bhat의 의미를 기술 하시오.
   8. bhat의 첫번째 값을 절편, 두 번째 값을 기울기로 사용하여 d. 에서 생성한 산점도 위에 빨강색 직선을 삽입하는 스크립트를 작성 하시오.
   9. bhat을 라고 할 때, 를 계산한 결과를 벡터 타입으로 yhat에 저장하는 스크립트를 작성 하시오.
   10. yhat 계산 결과의 의미에 대해 기술 하시오.
   11. 지금까지 스크립트 결과를 이용하여 잔차(residual, ), 결정계수()를 계산 하시오(**hint**: , , 계산).

* set.seed(20200618)  
  x <- rnorm(200, mean = 4, sd = sqrt(9)) # 2-a  
  e <- rnorm(200, mean = 0, sd = sqrt(2)) # 2-b  
  y <- 1 + 0.6\*x + e # 2-c  
  plot(x, y) # 2-d  
  X <- cbind(1, x) # 2-e  
  bhat <- solve(t(X) %\*% X) %\*% t(X) %\*% y # 2-f  
  # 2-g: 일변량 회귀계수의 추정 절편 및 기울기 벡터  
  abline(a = bhat[1], b = bhat[2], col = "red") # 2-h



yhat <- c(X %\*% bhat) #2-i  
# 2-j: 추정한 회귀계수로 적합(fitting)한 예측값  
# 2-k  
residual <- y - yhat  
SST <- sum((y - mean(y))^2); SST

[1] 925.3353

SSE <- sum((y - yhat)^2); SSE

[1] 386.0555

SSR <- SST - SSE; SSR

[1] 539.2798

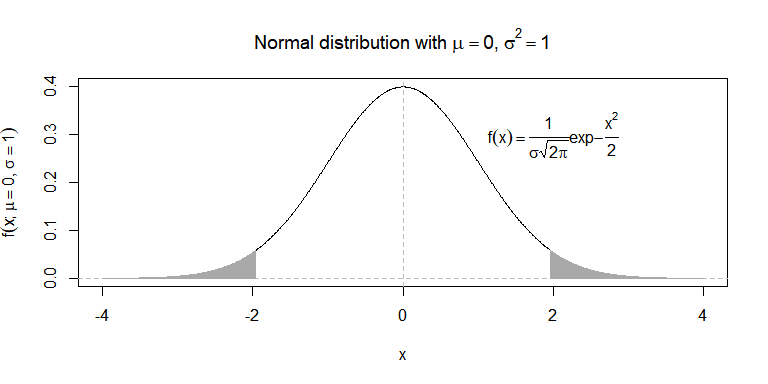
R2 <- SSR/SST; R2

[1] 0.5827939

#

1. 평균이 이고, 분산이 인 정규분포의 확률밀도함수(probability density function, pdf)는 아래와 같다.
2. 평균이 이고 분산이 인 확률밀도함수 값을 구하는 함수 pdf\_norm() 함수를 작성 하시오.
3. 아래 그림과 같이 -4에서 4 사이의 의 그림을 출력한 후,
4. ploygon 함수에 대한 사용법을 도움말을 통해 익힌 다음, , 영역을 아래 그림과 같이 표시 하시오
   * 음영색은 darkgray
   * x = 0, y = 0 에 대한 선은 회색 점선
   * x = 2, y = 3 에 표준정규분포 수식 입력

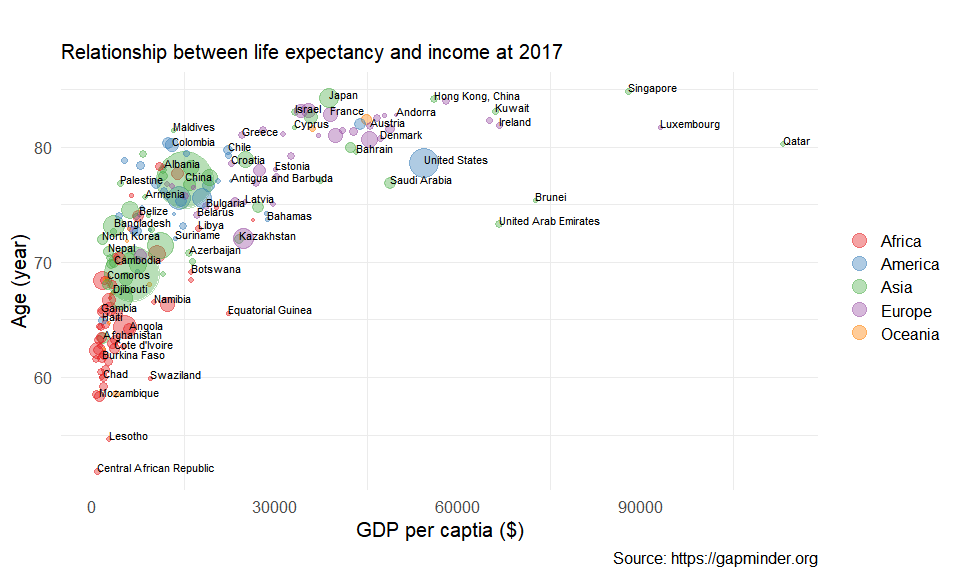
# pdf\_norm <- function(x, mu = 0, sigma = 1) {  
# 1/(sqrt(2\*pi)\*sigma) \* exp(-(x - mu)^2/(2\*sigma^2))  
# }  
x <- seq(-4, 4, length = 1000)  
y <- dnorm(x)  
# y <- pdf\_norm(x)  
plot(x, y, type = "n",   
 xlab = expression(x),   
 ylab = expression(f(x\*";"~mu==0, sigma==1)),   
 xlim = c(-4, 4),   
 ylim = c(0, 0.4),   
 main = expression(paste("Normal distribution with ",   
 mu == 0, ", ", sigma^2 == 1)))  
  
lines(x, y, lwd = 1)  
abline(v = 0, h=0, lty = 2, col = "gray")  
idx1 <- x >= 1.96; idx2 <- x <= -1.96   
polygon(c(1.96, x[idx1], 4),   
 c(0, y[idx1], 0),   
 border = "darkgray",   
 col="darkgray")  
polygon(c(-4, x[idx2], -1.96),   
 c(0, y[idx2], 0),   
 border = "darkgray",  
 col="darkgray")  
  
text(2, 0.3, expression(f(x) == frac(1, sigma\*sqrt(2\*pi)) \* "exp"\*-frac(x^2, 2)))



# 올바른 텍스트 표현은 아래와 같음  
# text(2, 0.3, expression(f(x) == frac(1, sigma\*sqrt(2\*pi)) \* "exp"\*bgroup("(", -frac(x^2, 2), ")")))

1. 다음은 dplyr 문법과 ggplot2 에 대한 내용이다. 아래 물음에 대해 스크립트를 작성 하시오.
   1. 사이버 캠퍼스 자료실 “통계프로그래밍언어 9강 실습 데이터”에서 다운로드 한 gapminder\_filter.csv 를 현재 작업 디렉토리 하위 폴더 gapminder 안에 저장 후, R 작업공간에서 해당 파일이 정상적으로 컴퓨터 내에 저장되었는지를 확인하는 스크립트를 작성 하시오.
   2. gapminder\_filter.csv 파일 내용을 gapdat 객체에 저장하는 스크립트를 작성 하시오.
   3. 아래 명령을 수행하는 스크립트를 파이프 연산자를 이용해 연결한 결과를 gap\_2017 객체에 저장 하시오.
      * 정규표현식을 이용해 gapdat 객체 내에 Asia, Europe, America, Oceania 값을 갖는 continent 변수를 생성 하시오.
      * continent 변수를 factor 형으로 변환
      * year가 2017년도인 데이터 추출
   4. 아래 도표는 c 에서 생성한 gap\_2017 데이터에서 GDP/Captia와 Age 간 산점도에 대한 그래프이다. 해당 그래프에서 각 점의 크기(size)는 해당 국가의 인구에 비례하며 각 점들의 색(color)은 대륙을 나타낸다. 그래프를 생성하기 위해 사용한 ggplot의 옵션에 대한 설명은 아래와 같다.
      * 산점도의 색상 투명도는 0.4로 설정
      * 대륙(continent)을 대표하는 색상은 scale\_color\_brewer()를 이용해 다양화 했으며, 사용한 palette는 "Set1" 임
      * 크기(size)의 최솟값과 최댓값은 각각 1, 20
      * 범례(legend)의 symbol (점 모양)의 크기는 5
      * 사용한 theme은 minimal
      * 표시된 텍스트는 국가명(country)을 나타내며, 수직 및 수평 정렬 값은 각각 0, 0 임. 겹치는(overlapping) 텍스트는 표시하지 않음

# dir("gapminder") # 1-a  
require(tidyverse)  
gapdat <- read\_csv("gapminder/gapminder\_filter.csv") # 1-b  
  
gapdat %>%   
 mutate(continent = gsub("(.+\\s)", "", region) %>%   
 factor) %>%   
 filter(year == 2017) -> gap\_2017 # 1-c  
  
ggplot(data = gap\_2017) + # 1-d  
 aes(x = gdp\_cap, y = life\_expectancy) +   
 geom\_point(  
 aes(size = population, color = continent),   
 alpha = 0.4  
 ) +   
 geom\_text(  
 aes(label = country),   
 vjust = 0,  
 hjust = 0, check\_overlap = TRUE,   
 size = 3  
 ) +   
 scale\_color\_brewer(palette = "Set1") +   
 scale\_size\_continuous(range = c(1, 20)) +   
 labs(title = "",   
 subtitle = "Relationship between life expectancy and income at 2017",   
 caption = "Source: https://gapminder.org",   
 x = "GDP per captia ($)",   
 y = "Age (year)") +   
 guides(size = FALSE,   
 color = guide\_legend(override.aes = list(size = 5))) +   
 theme\_minimal(base\_size = 15) +   
 theme(  
 panel.grid.major.x = element\_blank(),   
 legend.position = "right",   
 legend.title = element\_blank(),   
 )



1. 다음은 ggplot2 패키지(tidyverse 패키지에 포함)에 내장되어 있는 diamonds 데이터 자료에 대한 분석 스크립트 이다.

diamond <- within(diamonds, {  
 status = ifelse(price <= 5000, "cheap",   
 ifelse(price <= 10000, "normal", "expensive"))  
})  
diamond$status <- with(diamond, factor(status,   
 levels = c("cheap",   
 "normal",   
 "expensive")))  
tab\_status\_cut <- data.frame(with(diamond, table(status, cut)))  
desc\_status <- aggregate(cbind(carat, x, y, z) ~ status + cut,   
 data = diamond,   
 FUN = function(x) c(mean = mean(x),   
 sd = sd(x),   
 median = median(x),   
 iqr = IQR(x)))  
  
desc\_status <- merge(tab\_status\_cut, desc\_status,   
 by.x = c("status", "cut"),   
 by.y = c("status", "cut"))

1. desc\_status를 도출하기 위한 위 스크립트을 dplyr 패키지에서 제공하는 함수들을 이용해 재구성 하시오(단, 재구성한 결과는 desc\_status\_new에 저장 하시오).

lab\_status <- c("cheap", "normal", "expensive")  
diamond <- diamonds %>%   
 mutate(status = ifelse(price <= 5000, lab\_status[1],   
 ifelse(price <= 10000, lab\_status[2], lab\_status[3])),   
 status = factor(status, levels = lab\_status))  
diamond %>%   
 add\_count(status, cut) %>%   
 rename(Freq = n) %>%   
 group\_by(status, cut, Freq) %>%   
 summarise\_at(vars(carat, x, y, z),   
 list(mean = ~ mean(.),   
 sd = ~ sd(.),   
 median = ~ median(.),   
 iqr = ~ IQR(.))) -> desc\_status\_new

1. 아래 표는 desc\_status\_new를 재구성한 표의 일부이다. dat\_status\_new의 결과를 아래의 표와 같은 형태로 변환 하시오(전체 결과는 30개의 행과 8개의 열로 구성된 티블 객체임).

desc\_status\_new %>%   
 pivot\_longer(  
 -status:-Freq,   
 names\_to = "var\_stat",   
 values\_to = "value"  
 ) %>%   
 separate(var\_stat, c("variable", "stat")) %>%   
 spread(stat, value) %>% # pivot\_wider 대신 spread 동사 사용  
 mutate(  
 `Mean (SD)` = sprintf("%.2f (%.2f)", mean, sd),   
 `Median (IQR)` = sprintf("%.2f (%.2f)", median, iqr)  
 ) %>%   
 select(-iqr:-sd) %>%   
 mutate(variable = factor(variable)) %>%   
 pivot\_longer(  
 `Mean (SD)`:`Median (IQR)`,   
 names\_to = "stat",   
 values\_to = "res"  
 ) %>%   
 spread(variable, res) -> tab\_ex  
  
# excel 테이블 추가 scrpit  
require(knitr)   
require(flextable)  
  
regulartable(head(tab\_ex)) %>%   
 fontsize(size = 9, part = "all") %>%   
 autofit

| status | cut | Freq | stat | carat | x | y | z |
| --- | --- | --- | --- | --- | --- | --- | --- |
| cheap | Fair | 1182 | Mean (SD) | 0.82 (0.25) | 5.84 (0.62) | 5.78 (0.62) | 3.72 (0.44) |
| cheap | Fair | 1182 | Median (IQR) | 0.90 (0.30) | 6.00 (0.75) | 5.94 (0.73) | 3.76 (0.63) |
| cheap | Good | 3665 | Mean (SD) | 0.66 (0.27) | 5.42 (0.80) | 5.43 (0.80) | 3.38 (0.50) |
| cheap | Good | 3665 | Median (IQR) | 0.70 (0.50) | 5.58 (1.41) | 5.58 (1.41) | 3.40 (0.88) |
| cheap | Very Good | 8739 | Mean (SD) | 0.59 (0.26) | 5.24 (0.79) | 5.27 (0.79) | 3.25 (0.58) |
| cheap | Very Good | 8739 | Median (IQR) | 0.53 (0.43) | 5.20 (1.37) | 5.23 (1.38) | 3.22 (0.87) |