

11

강

R 데이터 분석

R을 이용한 고급 그래픽 기법 II

이화여자대학교 이은경 교수



강의 목차

8강 R 통계 그래픽스 I 이은경

9강 R 통계 그래픽스 II 이은경

10강 R을 이용한 고급 그래픽 기법 I 이은경

11강 R을 이용한 고급 그래픽 기법 II 이은경

12강 일반화 선형모형 I 이윤동

13강 일반화 선형모형 II 이윤동

14강 분류 I 이윤동

15강 분류 II 이윤동



학습목차

- 1 shiny 설치 및 shiny app 구동하기
- 2 shiny로 다양한 widget 구현하기
- 3 knitr 패키지를 이용한 다이나믹 문서 작성

R 데이터 분석

1 shiny 설치 및 shiny app 구동하기

shiny란?

- R에서 웹 프로그래밍을 가능하게 해주는 패키지
- 최신 웹 브라우저를 이용하여 인터랙티브한 데이터 정리와 질의를 쉽게 할 수 있는 기능을 제공
- 다양한 위젯이 있어 사용자 인터페이스와 인터랙티브 기능을 쉽고 빠르게 구현할 수 있게 함.
- 자바 스크립트를 함께 사용가능
- Rstudio에서 이용하는 것이 편리함.

10 R을 이용한 고급 그래픽 기법 I

shiny 설치 및 시작하기

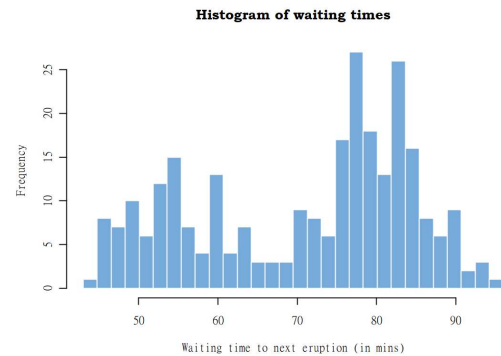
- R에 내장되어 있는 package가 아니므로 install 해야 함.

```
install.packages("shiny")
```

- Shiny package 내에 11개의 example을 제공하고 있음.

```
library(shiny)  
runExample("01_hello")
```

Hello Shiny!



shiny App의 기본 구성 요소

- 하나의 파일로 구성하는 경우
 - app.R : 하나의 파일에 user interface와 input/output 관련된 부분을 모두 정의
- 두 개 이상의 파일로 구성하는 경우
 - ui.R: App의 user interface에 관련된 부분을 정의
 - server.R: App의 input/output에 관련된 부분을 정의
 - helpers.R: UI, input/output을 제외한 나머지 부분에 관련된 함수들 정의

10 R을 이용한 고급 그래픽 기법 I

01_hello example 살펴보기: app.R 전체

```

library(shiny)

ui <- fluidPage(
  titlePanel("Hello Shiny!"),
  sidebarLayout(
    sidebarPanel(
      sliderInput(inputId = "bins",
        label = "Number of bins:",
        min = 1,
        max = 50,
        value = 30)
    ),
    mainPanel(
      plotOutput(outputId = "distPlot")
    )
  )
)

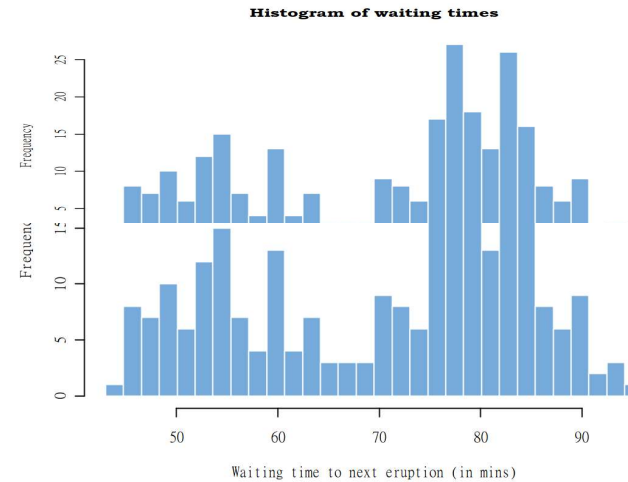
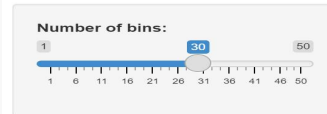
server <- function(input, output) {
  output$distPlot <- renderPlot({
    x <- faithful$waiting
    bins <- seq(min(x), max(x),
      length.out = input$bins + 1)

    hist(x,
      breaks = bins,
      col = "#75AADB",
      border = "white",
      xlab = "Waiting time to next eruption (in mins)",
      main = "Histogram of waiting times")
  })
}

shinyApp(ui = ui, server = server)

```

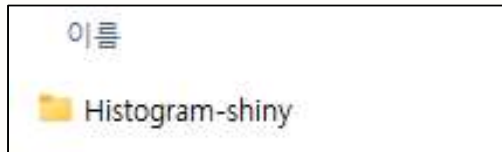
Hello Shiny!



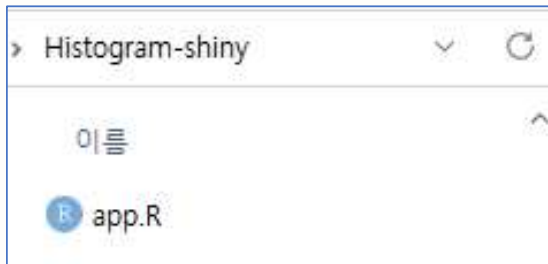
10 R을 이용한 고급 그래픽 기법 I

shiny app 실행 방법

- app의 이름과 같은 이름의 directory 만들기



- app.R을 directory 내에 저장

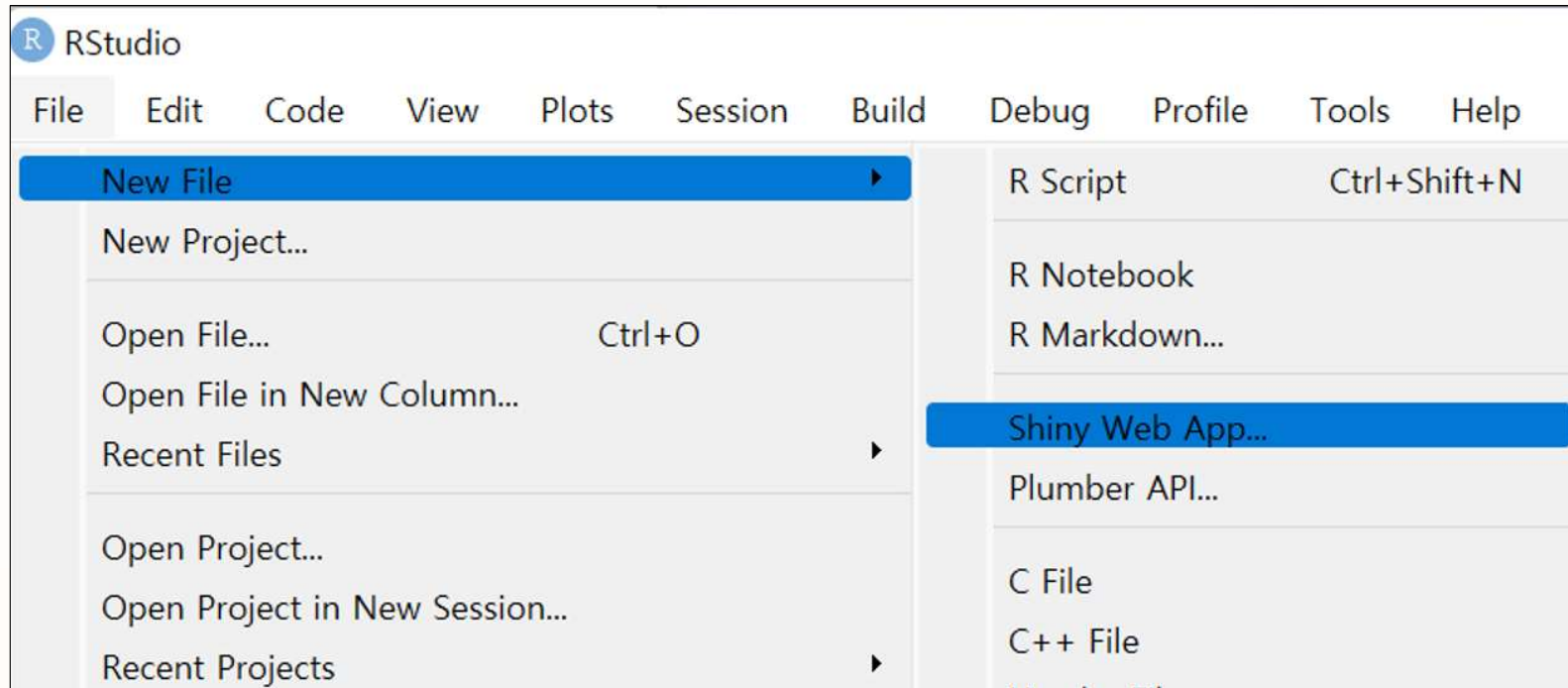


- runApp("Histogram-shiny") 명령어로 실행

```
runApp("Histogram-shiny")
```

R studio에서 shiny app 실행 방법

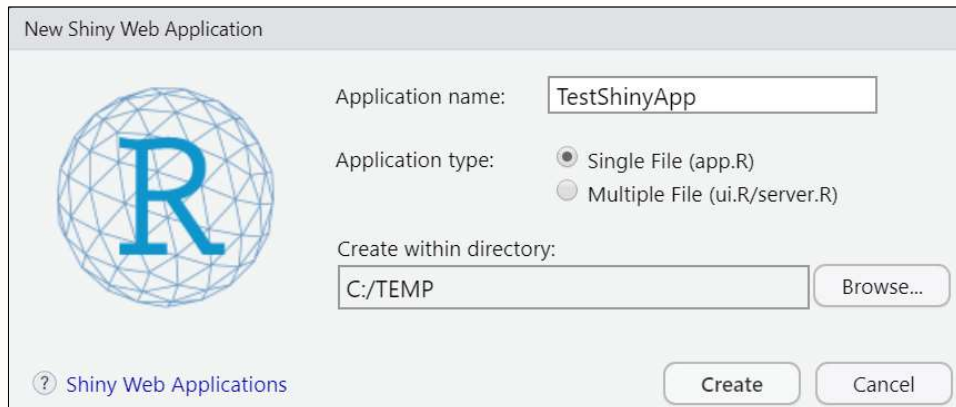
- File -> New File -> Shiny Web App 메뉴 선택하기



10 R을 이용한 고급 그래픽 기법 I

R studio에서 shiny app 실행 방법

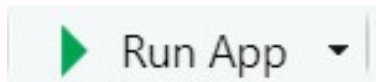
- Application name에 App 이름 지정하기
- application type 지정하기
 - app.R의 하나의 파일을 이용하는 경우 : Single File 선택
 - ui.R과 server.R의 두 파일을 이용하는 경우 : Multiple File 선택
- shiny app 파일을 저장할 directory 지정하기
- create 버튼 누르기



10 R을 이용한 고급 그래픽 기법 I

R studio에서 shiny app 실행 방법

- app.R 파일 열어서 수정하기
- editor 화면 오른쪽 위에 있는



버튼을 눌러

실행시키기

```
1 #
2 # This is a Shiny web application. You can run the application by cli
3 # the 'Run App' button above.
4 #
5 # Find out more about building applications with Shiny here:
6 #
7 # http://shiny.rstudio.com/
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15   # Application title
16   titlePanel("Old Faithful Geyser Data"),
17
18   # Sidebar with a slider input for number of bins
19   sidebarLayout(
20     sidebarPanel(
21       sliderInput("bins",
22                 "Number of bins:",
23                 min = 1,
24                 max = 50,
25                 value = 30)
26     ),
27
28     # Show a plot of the generated distribution
29     mainPanel(
30       plotOutput("distPlot")
31     )
32   )
33 )
34
35 # Define server logic required to draw a histogram
36 server <- function(input, output) {
37
38   output$distPlot <- renderPlot({
39     # generate bins based on input$bins from ui.R
40     x <- faithful[, 2]
41     bins <- seq(min(x), max(x), length.out = input$bins + 1)
42
43     # draw the histogram with the specified number of bins
44     hist(x, breaks = bins, col = 'darkgray', border = 'white')
45   })
46 }
47
48 # Run the application
49 shinyApp(ui = ui, server = server)
50
```

01_hello example 살펴보기 : ui 부분

```
ui <- fluidPage(  
  titlePanel("Hello Shiny!"),  
  sidebarLayout(  
    sidebarPanel(  
      sliderInput(inputId = "bins",  
                  label = "Number of bins:",  
                  min = 1,  
                  max = 50,  
                  value = 30)  
    ),  
    mainPanel(  
      plotOutput(outputId = "distPlot")  
    )  
  )  
)
```

- fluidPage는 browser의 크기에 따라 자동조절이 가능한 webpage로 만들기 위한 함수

01_hello example 살펴보기 : ui 부분

```
ui <- fluidPage(  
  titlePanel("Hello Shiny!"),  
  sidebarLayout(  
    sidebarPanel(  
      sliderInput(inputId = "bins",  
                  label = "Number of bins:",  
                  min = 1,  
                  max = 50,  
                  value = 30)  
    ),  
    mainPanel(  
      plotOutput(outputId = "distPlot")  
    )  
  )  
)
```

App의 title을 제공

- sidebar panel과 main panel을 구성
- sidebar는 왼쪽에, main panel은 오른쪽에 나타남.

01_hello example 살펴보기 : ui 부분

```
ui <- fluidPage(  
  titlePanel("Hello Shiny!"),  
  sidebarLayout(  
    sidebarPanel(  
      sliderInput(inputId = "bins",  
                  label = "Number of bins:",  
                  min = 1,  
                  max = 50,  
                  value = 30)  
    ),  
    mainPanel(  
      plotOutput(outputId = "distPlot")  
    )  
  )  
)
```

- sidebar panel 구성
- slideInput을 이용
- input
 - slideInput로부터의 bins
 - default값은 30

- main panel 구성
- plotOutput을 이용하여 그래프 출력
- output : distPlot

01_hello example 살펴보기: server 부분

```
server <- function(input, output) {  
  output$distPlot <- renderPlot({  
    x <- faithful$waiting  
    bins <- seq(min(x), max(x),  
                length.out = input$bins + 1)  
    hist(x,  
          breaks = bins,  
          col = "#75AADB",  
          border = "white",  
          xlab = "Waiting time to next eruption (in mins)",  
          main = "Histogram of waiting times")  
  })  
}
```

- App의 모든 입력과 출력을 관할
- input과 output은 list 형태
- UI로부터의 모든 입력은 input으로, 모든 출력은 output으로 함.

01_hello example 살펴보기: server 부분

UI의 main panel에 그려진 그래프로
plotOutput을 이용하여 출력

```
server <- function(input, output) {  
  output$distPlot <- renderPlot({  
    x <- faithful$waiting  
    bins <- seq(min(x), max(x),  
                 length.out = input$bins + 1)  
    hist(x,  
         breaks = bins,  
         col = "#75AADB",  
         border = "white",  
         xlab = "Waiting time to next eruption (in mins)",  
         main = "Histogram of waiting times")  
  })  
}
```

UI의 sliderbar로부터의
input으로 정수를 나타냄

01_hello example 살펴보기: shinyApp부분

```
shinyApp(ui=ui, server=server)
```

- shiny app 구동을 하기 위한 함수
- ui 부분과 server 부분 지정

Examples in shiny library

```
runExample("01_hello") # a histogram
runExample("02_text") # tables and data frames
runExample("03_reactivity")
    # a reactive expression
runExample("04_mpg") # global variables
runExample("05_sliders") # slider bars
runExample("06_tabsets") # tabbed panels
runExample("07_widgets")
    # help text and submit buttons
runExample("08_html") # Shiny app build from HTML
runExample("09_upload") # file upload wizard
runExample("10_download") # file download wizard
runExample("11_timer") # an automated timer
```

R 데이터 분석

2 shiny로 다양한 widget 구현하기

다양한 control widget들

- `actionButton` : Action Button
- `checkboxGroupInput` : A group of check boxes
- `checkboxInput` : A single check box
- `dateInput` : A calendar to aid date selection
- `dateRangeInput` : A pair of calendars for selecting a date range
- `fileInput` : A file upload control wizard

다양한 control widget들

- `helpText` : Help text that can be added to an input form
- `numericInput` : A field to enter numbers
- `radioButtons` : A set of radio buttons
- `selectInput` : A box with choices to select from
- `sliderInput` : A slider bar
- `submitButton` : A submit button
- `textInput` : A field to enter text

11 R을 이용한 고급 그래픽 기법 II

single checkbox 만들기

■ ui

```
checkboxInput("checkbox",  
          label = "Choice A",  
          value = TRUE)  
textOutput("checkboxOut"))
```

■ server

```
output$checkboxOut <-  
renderPrint({input$checkbox})
```

Single checkbox

☒ Choice A

[1] TRUE

11 R을 이용한 고급 그래픽 기법 II

select box 만들기

■ ui

```
selectInput("select",  
  label = "Select box",  
  choices = list("Choice 1" = 1,  
                 "Choice 2" = 2,  
                 "Choice 3" = 3),  
  selected = 1))  
textOutput("selectOut"))
```



Select box

Choice 1 ▼

[1] "1"

■ server

```
output$selectOut <-  
renderPrint({ input$select })
```


11 R을 이용한 고급 그래픽 기법 II

checkbox group 만들기

■ ui

```
checkboxGroupInput("checkGroup",  
  label = h3("Checkbox group"),  
  choices = list("Choice 1" = 1,  
                 "Choice 2" = 2,  
                 "Choice 3" = 3),  
  selected = 1))  
textOutput("checkGroupOut"))
```

■ server

```
output$checkGroupOut <-  
renderPrint({ input$checkGroup })
```

Checkbox group

- ☒ Choice 1
- ☐ Choice 2
- ☒ Choice 3

[1] "1" "3"

11 R을 이용한 고급 그래픽 기법 II

radio button group 만들기

■ ui

```
radioButtons("radio",  
  label = h3("Radio buttons"),  
  choices = list("Choice 1" = 1,  
                 "Choice 2" = 2,  
                 "Choice 3" = 3),  
  selected = 1))  
textOutput("radioOut"))
```

Radio buttons

- ☐ Choice 1
- ☐ Choice 2
- ☒ Choice 3

[1] "3"

■ server

```
output$radioOut <-  
  renderPrint({ input$radio })
```

11 R을 이용한 고급 그래픽 기법 II

최대값 조정 slider 만들기

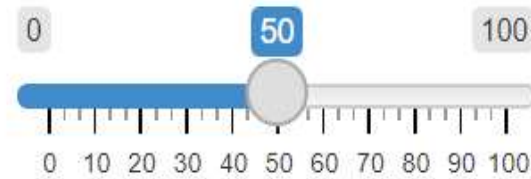
■ ui

```
sliderInput("slider1",  
  label = h3("slider 1"),  
  min = 0, max = 100,  
  value = 50)  
textOutput("slider1Out"))
```

■ server

```
output$slider1Out <-  
renderPrint({ input$slider1 })
```

Slider 1



[1] 50

11 R을 이용한 고급 그래픽 기법 II

최솟값과 최대값을 조정하는 slider 만들기

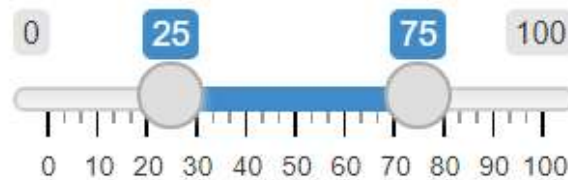
■ ui

```
sliderInput("slider2",  
  label = h3("slider 2"),  
  min = 0, max = 100,  
  value = c(25, 75))  
textOutput("slider2Out"))
```

■ server

```
output$slider2Out <-  
renderPrint({ input$slider2 })
```

Slider 2



[1] 25 75

11 R을 이용한 고급 그래픽 기법 II

text input 만들기

■ ui

```
textInput("text",  
  label = h3("Text input"),  
  value = "Enter text...")  
textOutput("textOut"))
```

■ server

```
output$textOut <-  
renderPrint({ input$text})
```

Text input

문자열 입력

[1] "문자열 입력"

11 R을 이용한 고급 그래픽 기법 II

numeric input 만들기

■ ui

```
numericInput("num",  
  label = h3("Numeric input"),  
  value = 1)  
textOutput("numOut"))
```

Numeric input

[1] 1

■ server

```
output$numOut <-  
renderPrint({ input$num})
```

11 R을 이용한 고급 그래픽 기법 II

app.R

```
library(shiny)
ui <- fluidPage(
  titlePanel("Basic widgets"),
  fluidRow(
    column(3, h3("Single checkbox"),
      checkboxInput("checkbox", label = "Choice A", value = TRUE)),
    column(3, selectInput("select",
      label = h3("Select box"),
      choices = list("Choice 1" = 1,
                    "Choice 2" = 2,
                    "Choice 3" = 3),
      selected = 1)),
    column(3, checkboxGroupInput("checkGroup",
      label = h3("Checkbox group"),
      choices = list("Choice 1" = 1,
                    "Choice 2" = 2, "Choice 3" = 3),
      selected = 1)),
    column(3, radioButtons("radio", label = h3("Radio buttons"),
      choices = list("Choice 1" = 1, "Choice 2" = 2,
                    "Choice 3" = 3), selected = 1))
  ),
),
```

11 R을 이용한 고급 그래픽 기법 II

app.R

```
fluidRow(
  column(3, textOutput("checkboxOut")),
  column(3, textOutput("selectOut")),
  column(3, textOutput("checkGroupOut")),
  column(3, textOutput("radioOut"))
),
fluidRow(
  column(3, sliderInput("slider1", label = h3("Slider 1"),
    min = 0, max = 100, value = 50)),
  column(3, sliderInput("slider2", label = h3("Slider 2"),
    min = 0, max = 100, value = c(25, 75))),
  column(3, textInput("text", label = h3("Text input"),
    value = "Enter text...")),
  column(3, numericInput("num",
    label = h3("Numeric input"),
    value = 1))
),
fluidRow(
  column(3, textOutput("slider1Out")),
  column(3, textOutput("slider2Out")),
  column(3, textOutput("textOut")),
  column(3, textOutput("numOut"))
)
)
```


11 R을 이용한 고급 그래픽 기법 II

app.R

```
server <- function(input, output) {  
  output$checkboxOut <- renderPrint({ input$checkbox })  
  output$checkGroupOut <- renderPrint({ input$checkGroup })  
  output$radioOut <- renderPrint({ input$radio })  
  output$selectOut <- renderPrint({ input$select })  
  
  output$slider1Out <- renderPrint({ input$slider1 })  
  output$slider2Out <- renderPrint({ input$slider2 })  
  output$textOut <- renderPrint({ input$text })  
  output$numOut <- renderPrint({ input$num })  
  
}  
# Run the application  
shinyApp(ui = ui, server = server)
```

11 R을 이용한 고급 그래픽 기법 II

app.R 실행 결과

Basic widgets

Single checkbox

☒ Choice A

[1] TRUE

Select box

Choice 1 ▼

[1] "1"

Checkbox group

☒ Choice 1☐ Choice 2☐ Choice 3

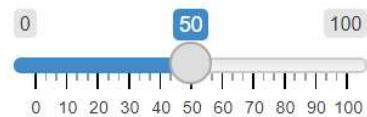
[1] "1"

Radio buttons

☒ Choice 1☐ Choice 2☐ Choice 3

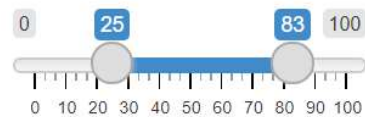
[1] "1"

Sliders



[1] 50

Sliders



[1] 25 83

Text input

Enter text...

[1] "Enter text..."

Numeric input

1

[1] 1

R 데이터 분석

3 knitr 패키지를 이용한 다이나믹 문서 작성

markdown이란?

- 일반 문서를 웹문서 형태로 바꾸어 주는 도구
- 웹문서의 HTML 태그에 대한 사전 지식 없이도 웹문서를 만들 수 있도록 개발된 도구
- 일반 문서를 markdown에서 제공하는 형식대로 작성한 후 pandoc 프로그램을 이용하여 여러 형태의 문서로 바꿀 수 있음
- R code에 대한 형식을 추가하여 문서 변환 시 R code와 시행 결과를 함께 문서에 포함시킬 수 있도록 하는 R markdown이 개발되어 있음.

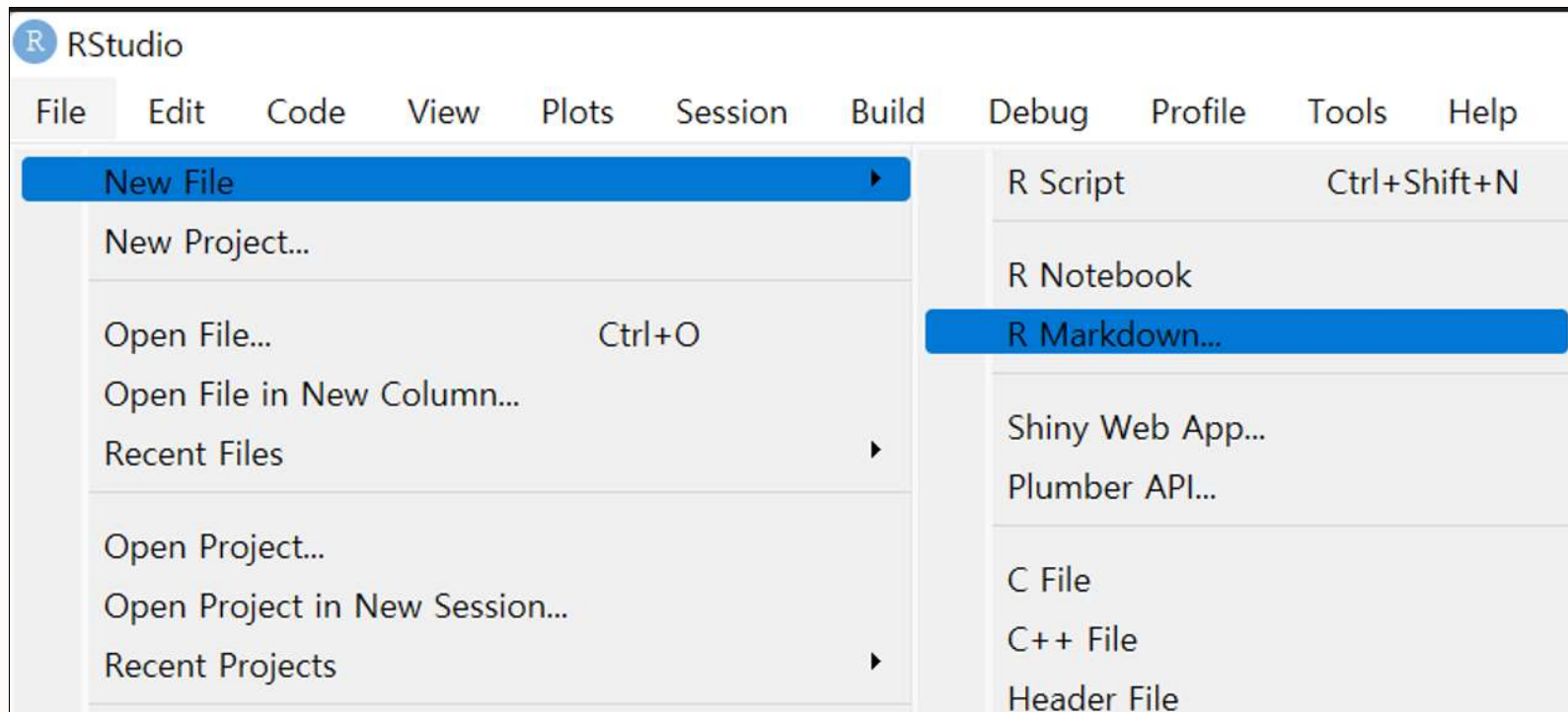
knitr 패키지

- R markdown 형태의 문서를 웹문서나 MS word 문서 혹은 tex 파일을 이용한 pdf 형태로 바꾸어 주는 패키지
- R studio에 내장되어 있음.
- R studio와 함께 이용하면 편리하게 최신의 정보를 이용한 보고서와 웹문서를 쉽게 만들 수 있음.
- R markdown과는 달리 pandoc 프로그램 없이 이용 가능

11 R을 이용한 고급 그래픽 기법 II

knitr을 이용하여 rmd 파일을 html 문서로 바꾸기

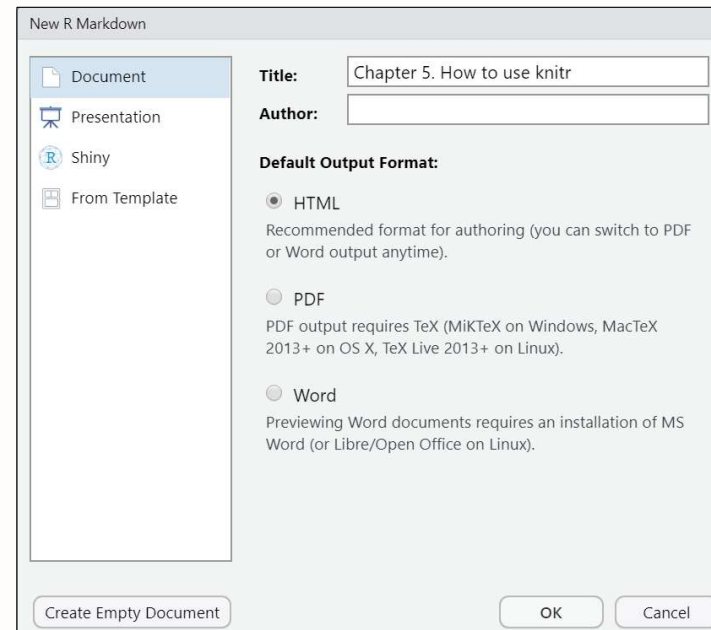
- File -> New File -> R markdown 메뉴 선택



11 R을 이용한 고급 그래픽 기법 II

knitr을 이용하여 rmd 파일을 html 문서로 바꾸기

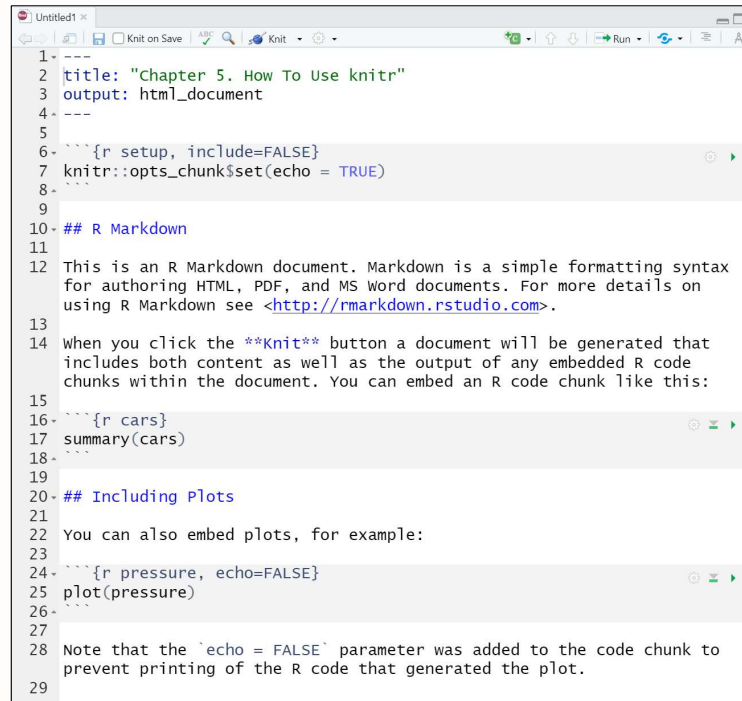
- title 지정하기
- 문서의 형태 지정하기
 - html : 웹문서
 - pdf : tex가 있는 경우에만 사용 가능
 - word : MS word문서
- OK 버튼 누르기



11 R을 이용한 고급 그래픽 기법 II

knitr을 이용하여 rmd 파일을 html 문서로 바꾸기

- 파일내용 수정하기
- 파일 저장하기
- knit 버튼을 눌러 문서 만들기



```
1 ---
2 title: "Chapter 5. How To Use knitr"
3 output: html_document
4 ---
5
6 ```{r setup, include=FALSE}
7 knitr::opts_chunk$set(echo = TRUE)
8 ```
9
10 ## R Markdown
11
12 This is an R Markdown document. Markdown is a simple formatting syntax
13 for authoring HTML, PDF, and MS word documents. For more details on
14 using R Markdown see <http://rmarkdown.rstudio.com>.
15
16 When you click the Knit button a document will be generated that
17 includes both content as well as the output of any embedded R code
18 chunks within the document. You can embed an R code chunk like this:
19
20 ```{r cars}
21 summary(cars)
22 ```
23
24 ## Including Plots
25
26 You can also embed plots, for example:
27
28 ```{r pressure, echo=FALSE}
29 plot(pressure)
30 ```
31
32 Note that the `echo = FALSE` parameter was added to the code chunk to
33 prevent printing of the R code that generated the plot.
```

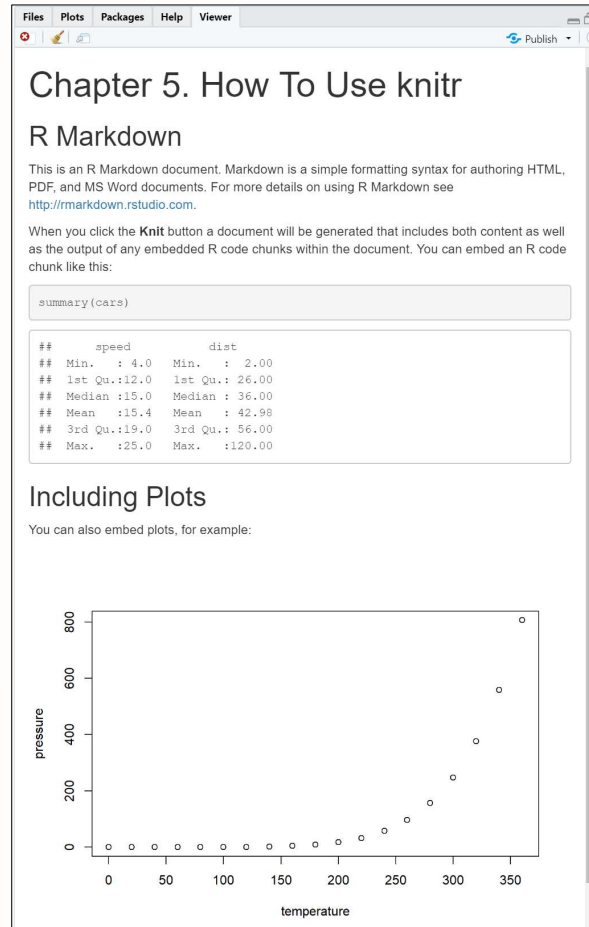

11 R을 이용한 고급 그래픽 기법 II

knitr을 이용하여 rmd 파일을 html 문서로 바꾸기

```

1  ---
2  title: "Chapter 5. How To Use knitr"
3  output: html_document
4  ---
5
6  ```{r setup, include=FALSE}
7  knitr::opts_chunk$set(echo = TRUE)
8  ```
9
10 ## R Markdown
11
12 This is an R Markdown document. Markdown is a simple formatting syntax
13 for authoring HTML, PDF, and MS Word documents. For more details on
14 using R Markdown see <http://rmarkdown.rstudio.com>.
15
16 when you click the Knit button a document will be generated that
17 includes both content as well as the output of any embedded R code
18 chunks within the document. You can embed an R code chunk like this:
19
20 ```{r cars}
21 summary(cars)
22 ```
23
24 ## Including Plots
25
26 You can also embed plots, for example:
27
28 ```{r pressure, echo=FALSE}
29 plot(pressure)
30 ```
31
32 Note that the `echo = FALSE` parameter was added to the code chunk to
33 prevent printing of the R code that generated the plot.

```



제목 글씨 크기 조정하기

```
# 글씨크기 조정  
# Header 1  
## Header 2  
### Header 3  
#### Header 4  
##### Header 5  
##### Header 6
```

글씨크기 조정

Header 1

Header 2

Header 3

Header 4

Header 5

Header 6

11 R을 이용한 고급 그래픽 기법 II

글씨 모양 조정하기

글씨 모양 조정

기울임꼴

굵게

글씨 모양 조정

기울임꼴

굵게

11 R을 이용한 고급 그래픽 기법 II

글머리 기호 목록 만들기

글머리 기호 목록 만들기

* list 1

* list 2

+ sub item 1

+ sub item 2

글머리 기호 목록 만들기

- list 1

- list 2

- sub item 1

- sub item 2

번호 매기기

번호 매기기

1. list 1

2. list 2

i) sub item 1

ii) sub item 2

번호 매기기

1. list 1

2. list 2

i. sub item 1

ii. sub item 2

11 R을 이용한 고급 그래픽 기법 II

표만들기

표 만들기

```
A1      | A2
-----|-----
cell 11 | cell 12
cell 21 | cell 22
```

표 만들기

A1	A2
cell 11	cell 12
cell 21	cell 22

수식넣기

수식넣기

$$F(x) = \int_{-\infty}^x f(y) dy$$

수식넣기

$$F(x) = \int_{-\infty}^x f(y) dy$$

11 R을 이용한 고급 그래픽 기법 II

R code 없이 결과만 넣기

R code 없이 결과만 넣기

```
` `` {r, echo=FALSE}
```

```
library(ggplot2)
```

```
ggplot(iris,
```

```
  aes(x=Sepal.Length,
```

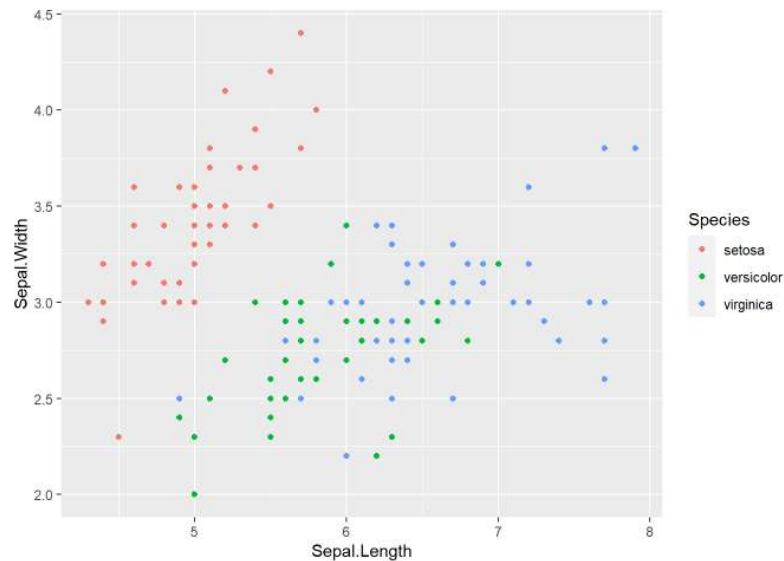
```
      y=Sepal.Width,
```

```
      color=Species))+
```

```
  geom_point()
```

```
` `` `
```

R code 없이 결과만 넣기



11 R을 이용한 고급 그래픽 기법 II

문장 가운데에 R code의 결과 넣기

```
### 문장 가운데에 R code의 결과 넣기  
* iris 자료의 개수는  
`r nrow(iris)` 이다.
```

문장 가운데에 R code의 결과 넣기

- iris 자료의 개수는 150 이다.

정리하기

- shiny를 이용하여 web app 구성하기
 - ui와 server 정의
- shiny의 다양한 control widget들
 - single checkbox, select box, checkbox group, radio button, slider, text input, numeric input
- knitr 패키지를 이용하여 다이나믹문서 만들기
- R markdown 문법들
 - 글씨 크기, 모양 조정, 수식넣기, 표 만들기
 - R code와 결과 넣기

12^강

다음시간 안내

일반화 선형모형 I

수고하셨습니다!

