Implementing Web Dashboard Application for Visualizing Earthquakes in South Korea with R and Shiny

소셜네트워크과학과 석사과정 한재윤

December 15, 2016

1 Introduction

최근 지진에 대한 국민들의 관심사가 어느 때보다 커지고 있다. 특히 2016년 9월 12일 한반도 관측 이래 최대의 지진 규모인 5.8을 기록한 경주 지진으로 인해 국민들의 불안감은 극대화되었다. 몇몇 네티즌들이 트위터에서 지진에 대한 버즈량이 급증할 경우 지진에 대해 푸시 알림을 제공하는 애플리케이션을 만드는 등, 지진 정보에 대한 국민들에 대한 수요 역시 커지고 있다. 그러나 국내에서 지진 데이터는 공공데이터포털[1] 및 기상청[2]을 통해서 어렵지 않게 구할 수 있으나, 이를 활용한 애플리케이션이나 2차 저작물은 찾아보기 힘들다. 외국의 경우 USGS의 데이터를 이용하여 전 세계의 지진에 관한 여러 정보를 태블로(Tableau) 등의 상용 시각화 소프트웨어를 사용하여 인터랙티브(interactive)한 웹 애플리케이션으로 배포하는 사례는 다수 존재한다.[3] 그러나 R과 같은 오픈소스 소프트웨어를 사용한 경우는 찾아보기 어렵다. 이에 본 프로젝트에서는 기상청의 최근 10년간 지진 데이터를 중심으로 오픈 소스 소프트웨어 및 웹 애플리케이션 플랫폼인 R과 Shiny를 활용하여 지진 분석 웹 애플리케이션을 제작하였다.

2 Methodology

2.1 R

R은 통계 계산과 데이터 시각화를 위한 프로그래밍 언어이자 소프트웨어 환경이다.[4] 대부분의 문법과 통계처리 부분은 AT&T 벨 연구소에서 개발했던 S 언어를 참고하였고, 이를 오픈소스화하여 뉴질랜드 오클랜드 대학의 로버트 젠틀맨(Robert Gentleman)과 로스 이하카(Ross Ihaka)에 의해 대중화되었다. 대표적인 함수형 언어인 Lisp에서 파생한 언어 특징을 가지고 있으며, 동시에 절차적인 문법 구조도 지원하며 모듈화된 패키지 개발을 위한 객체지향적 요소들을 포함하고 있다. R 언어는 부가적인 개발환경을 통해 기본적인 통계 기법부터 모델링, 최신 데이터 마이닝 기법까지 구현 및 개선이 가능하다. 구현한 결과는 Java, C, Python 등의 다른 프로그래밍 언어와도 연계하여 사용할 수 있다. 이러한 장점으로 R은 구글, 페이스북, 아마존 등의 데이터 분석이 필요한 기업에서 데이터 분석, 데이터 마이닝 및 시각화를 위해 널리 사용되고 있다.

2.2 Leaflet

Leaflet은 모바일 친화적인 인터랙티브 지도를 제공하는 오픈 소스 자바스크립트 라이브러리다.[5] 오픈스트리트맵(OpenStreetMap) 기반이며 인터랙티브에 초점이 잘 맞추어져 있기 때문에, 지도를 페이지에 출력한 후 다양한 분석이 가능하다는 장점이 있다. 또한 라이브러리의 용량이 33KB로 굉장히 작기 때문에 많은 개발자들이 선호하는 라이브러리 중 하나이기도 하다. Leaflet은 R 환경에서도 R용 패키지인 `leafletR`을 이용하여 사용할 수 있다. 기존에 자주 사용하는 CSV(Comma Separated Value) 형태의 데이터 뿐만 아니라 JSON 형태의 데이터까지 다룰 수 있다. 또한 R 환경 내에서 웹 애플리케이션을 제작 패키지인 `Shiny`등을 이용하여 지도가 포함된 웹 애플리케이션 제작에도 용이하다.

2.3 Shiny

Shiny는 R을 이용해 손쉽게 인터랙티브한 웹 애플리케이션을 만들 수 있는 패키지다.[6] Shiny는 R의 데이터 분석 기능 및 시각화 기능을 그대로 확장하여 HTML, CSS, JavaScript

등의 웹 프로그래밍 언어에 대한 지식 없이도 강력한 웹 애플리케이션을 만들 수 있는 기능을 제공한다.[7] 또한 만든 웹 애플리케이션을 무료로 호스팅할 수 있는 기능도 제공하기 때문에, 본 프로젝트의 목적과 매우 부합한다. 최근 데이터 시각화에 대한 수요가 늘어나면서 분석한 데이터를 바로 웹 애플리케이션으로 출판할 수 있는 Shiny를 사용하는 빈도가 늘고 있다. Jahanshiri and Shariff(2014)는 정밀농업 (Precision farming)을 위한 데이터를 분석하고 분석 결과를 Shiny를 이용해 웹 애플리케이션으로 배포하였다.[8] Liu *et al.*(2015)는 브라우저 기반의 생존 분석 시각화 툴을 웹 애플리케이션으로 배포하였다.[9]

3 Results

3.1 Overview

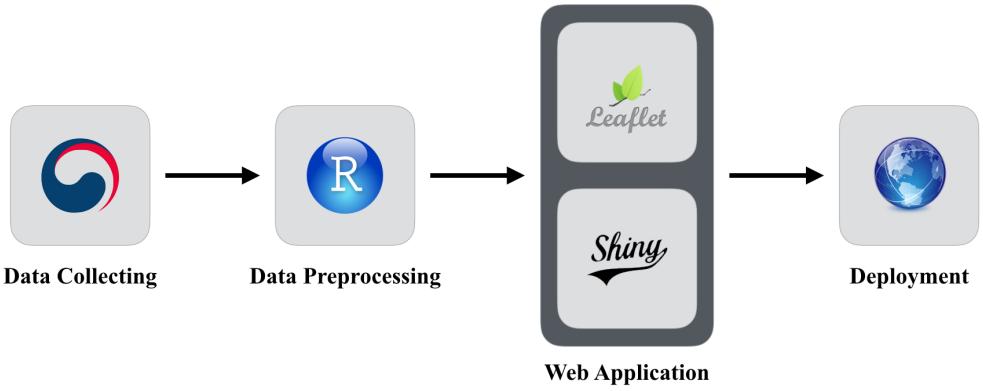

Figure 1: 웹 애플리케이션 제작 프로세스

웹 애플리케이션의 제작 프로세스는 Figure 1과 같다. 기상청의 지진 데이터를 스크래핑 또는 크롤링하여 수집하고, R을 이용하여 데이터 정제를 한다. 정제한 데이터를 Leaflet을 이용하여 지도를 출력하고, 해당 지도를 Shiny를 이용해 웹 애플리케이션으로 제작한다.

제작한 웹 애플리케이션은 shinyapps.io 또는 GitHub 등을 이용하여 배포한다. Shiny를 통해 제작한 웹 애플리케이션은 서버 파일과 UI(User Interface) 파일로 나뉜다. 각각의 파일의 코드는 부록에 첨부한다.

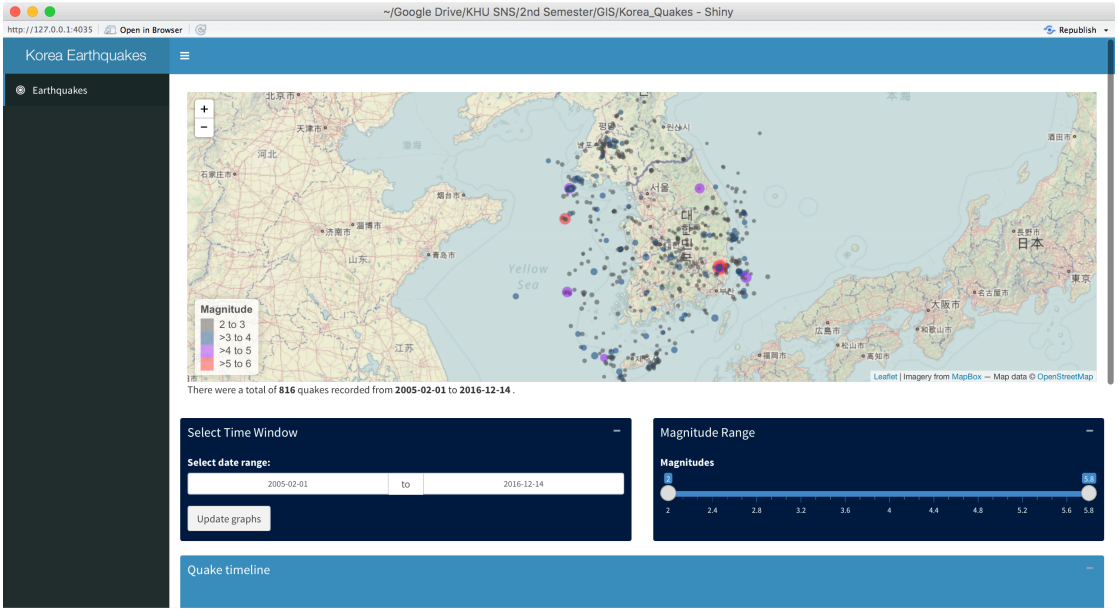

Figure 2: 웹 애플리케이션 인터페이스

Figure 2은 제작한 웹 애플리케이션의 전체적인 인터페이스를 보여준다. Leaflet을 통해 지진 데이터와 지도를 연동하여 맨 위에 출력하였고, 지도 바로 밑에 특정 기간에 일어난 지진에 대한 정보를 출력하도록 하였다. 지진 데이터의 경우, 기상청에서 규모 2.0 이상의 지진만 제공하기 때문에 범례의 기준을 2부터 1 단위로 6까지 나타내도록 하였다. 지진이 일어난 위치에 해당하는 지점의 포인트의 반지름은 $\text{Radius} = 1.5^{\text{Magnitude}}$로 정의한다.

3.2 Features

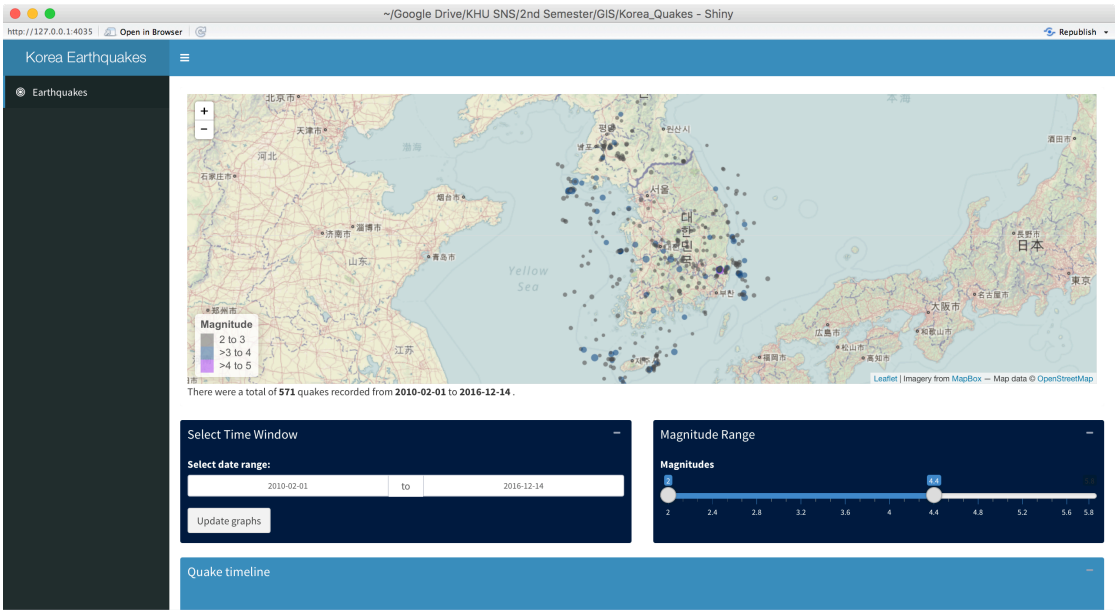

Figure 3: 타임 윈도우와 지진 규모 슬라이더

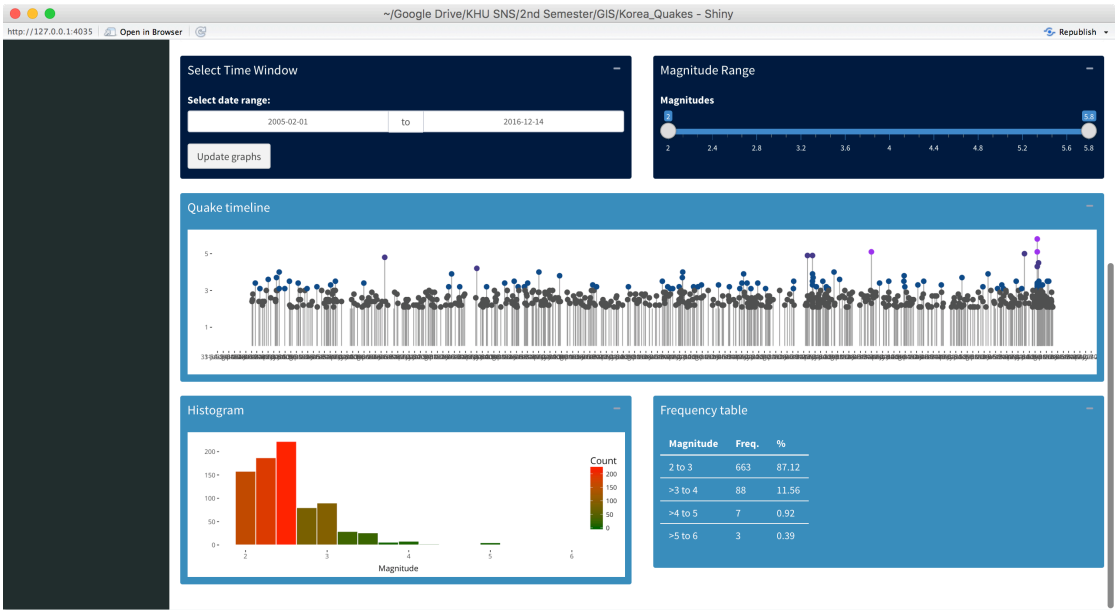

Figure 4: 타임 윈도우와 지진 규모 슬라이더로 필터링한 지진 데이터에 대한 정보를 한 눈에 확인할 수 있다.

지도 밑에 데이터 필터링을 위한 타임 윈도우와 슬라이더를 추가하였다. 원하는 기간과 원하는 규모 범위를 선택하면 실시간으로 지도에 표시되는 지진 데이터가 필터링된다. 이후에 'Update graphs' 버튼을 클릭하면 지진 데이터와 관련한 여러 가지 시각화 결과를 확인할 수 있다.

타임 윈도우의 경우, Figure 3의 왼쪽 하단 패널과 같이 2005년 2월 1일부터 원하는 시작 날짜를 정할 수 있으며 마지막 날짜의 경우 오늘 날짜부터 원하는 마지막 날짜까지 지정할 수 있다. Figure 3의 오른쪽 하단의 지진 규모 슬라이더는 기상청에서 제공되는 데이터의 지진 규모값의 최솟값, 최댓값을 범위로 한다. 상술했듯이 지진 규모의 최솟값은 데이터 특성상 2.0이며, 최댓값은 2016년 9월 12일 발생한 경주 지진의 규모인 5.8이다. 해당 슬라이더에서 0.1 단위로 데이터를 필터링할 수 있다.

데이터를 필터링한 후 'Update graph' 버튼을 누르면 대쉬보드 맨 끝에 데이터 시각화 결과물을 출력한다. 우선 시각화 패널 중 상단 패널에 각 날짜에 지진이 발생한 경우를 클리블랜드 점 그래프로 표현하였다. 시각화 왼쪽 하단 패널과 오른쪽 하단 패널은 각각 히스토그램과 빈도표를 구현했다.

4 Discussion

해당 웹 애플리케이션은 shinyapps.io를 통해 배포하였고, `https://otzslayer.shinyapps.io/kor_quakes/`에서 확인할 수 있다. 현재 디버깅 과정에 있으며, 몇 가지 버그를 수정 중에 있다. 현재 확인된 버그는 다음과 같다.

- 타임 윈도우에서 2005년 2월 1일 이전의 시점을 고르면 웹 애플리케이션이 작동을 중단함.
- 시각화 패널의 클리블랜드 점 그래프에서 몇몇 데이터들이 소실됨.

또한 유지보수 뿐만 아니라 지속적인 업데이트를 할 예정이다. 추후 업데이트할 내용은 다음과 같다.

- 2005년 이전의 지진 데이터 추가 확보

- 현재 시점 이후의 데이터를 기상청에서 자동적으로 받아올 수 있는 기능 탑재
- 시각화 패널에 추가적인 그래프 도입

References

[1] 공공데이터포털, http://www.data.go.kr

[2] 기상청, http://www.kma.go.kr

[3] USGS Earthquake Dashboard, http://public.tableau.com/views/USGSEarthquakeDashboard/USGSEarthquakeDashboard?:embed=y&:showVizHome=no&:showTabs=y&:display_count=y&:display_static_image=y&:bootstrapWhenNotified=true

[4] The R Project for Statistical Computing, http://www.r-project.org/

[5] Leaflet, http://leafletjs.com/

[6] Chang, W., Cheng, J., Allaire, JJ., Xie, Y., McPherson, J. (2015). *shiny: Web application Framework for R.* http://CRAN.R-project.org/package=shiny

[7] Beeley, C. (2016). *Web application development with R using Shiny.* Packt Publishing Ltd.

[8] Jahanshiri, E., and Shariff, A. R. M. (2014). *Developing web-based data analysis tools for precision farming using R and Shiny.* In IOP Conference Series: Earth and Environmental Science (Vol. 20, No. 1, p. 012014). IOP Publishing.

[9] Liu, J., Corrigan, B., Flockhart, D., and Zhao, L. (2015, October). *SASR (Survival Analysis using "Shiny" R): A Browser Based Survival Analysis Visualization*

Tool using R "Shiny". In JOURNAL OF PHARMACOKINETICS AND PHARMACODYNAMICS (Vol. 42, pp. S70-S71). 233 SPRING ST, NEW YORK, NY 10013 USA: SPRINGER/PLENUM PUBLISHERS. ISO 690

A Codes

A.1 server.R

```
library(shinydashboard)
library(leaflet)
library(dplyr)
require(reshape2)
library(scales)
require(ggplot2)
library(readr)

quake.file <- "data/kor_earth.csv"
quake <- read_csv(quake.file)

quake <- quake %>%
    mutate(dateTime = as.POSIXct(dateTime)) %>%
    mutate(latitude = as.numeric(gsub(x = quake$latitude, pattern = "␣N",
        replacement = ""))) %>%
    mutate(longitude = as.numeric(gsub(x = quake$longitude, pattern = "␣E",
        replacement = "")))

quake.sub <- quake
quake.sub$size <- cut(quake.sub$mag,
    c(2, 3, 4, 5, 5.8),
    labels = c("2␣to␣3", ">3␣to␣4", ">4␣to␣5", ">5␣to␣6"))
pallet <- colorFactor(c("gray32", "dodgerblue4",  "purple", "firebrick1"),
    domain = c("2␣to␣3", ">3␣to␣4", ">4␣to␣5", ">5␣to␣6"))

function(input, output, session) {
    #filter quake fn
    getQuakes <- function() {
        startDate <- as.POSIXlt(paste(as.character(input$daterange[1]),
        "00:00:01"))
```

```
29          endDate <- as.POSIXlt(paste(as.character(input$daterange[2]),
30          "23:59:01"))
31          quake.sub <- quake.sub[quake.sub$mag >= input$range[1] & quake.sub$
                mag <= input$range[2],]
32          quake.s <- quake.sub[quake.sub$dateTime > startDate &
33          quake.sub$dateTime < endDate,]
34          return(quake.s)
35      }
36      #leaflet quake map
37      qm <- function() {
38          quake.get <- getQuakes()
39          # create html for popup
40          pu <- paste("<b>Mag:</b>", as.character(quake.get$mag), "<br>",
41              "<b>Time:</b>", as.character.POSIXt(quake.get$dateTime), "<br>",
42              "<b>Place:</b>", quake.get$place #noticed some pecularities with
                    the place, need to re-check
43          )
44          #map
45          tempmap <- leaflet(data=quake.get) %>% addProviderTiles('MapBox.
                asheshwor.m4g4pnci') %>%
46              setView((min(quake$longitude) + max(quake$longitude))/2, (min(
                    quake$latitude) + max(quake$latitude))/2,  zoom = 6) %>%
47              addCircleMarkers(~longitude, ~latitude,
48                  popup = pu,
49                  radius = ~((1.5)^mag),
50                  color = ~pallet(size),
51                  stroke = FALSE, fillOpacity = 0.6) %>%
52              addLegend(
53                  "bottomleft", pal = pallet,
54                  values = sort(quake.get$size),
55                  title = "Magnitude"
56              )
57      }
```

```
58      ## timeline
59      drawHist <- eventReactive(input$updateButton, {
60          quake.sub <- getQuakes()
61          ggplot(quake.sub, aes(dateTime, mag, colour=size)) +
62          geom_bar(stat="identity", colour="gray60",
63              fill="gray60", alpha=0.5) +
64          geom_point(size=3) +
65          scale_colour_manual(name = "size",
66              values = c("gray32", "dodgerblue4", "slateblue4",
67                  "purple","␣firebrick1")) +
68          scale_x_datetime(breaks = date_breaks("1␣month"),
69              labels = date_format("%d-%b-%Y")) +
70          scale_y_continuous(breaks=c(seq(1,9,2))) +
71          xlab("") + ylab("Magnitude") +
72          theme(plot.background = element_rect(fill = "white", colour = NA),
73              panel.background = element_rect(fill = "white", colour = NA),
74              title = element_text(colour="black", size = 13),
75              axis.title.x = element_text(hjust=1, colour="black", size = 8),
76              axis.title.y = element_text(vjust=90, colour="dodgerblue4",
77                  size = 8),
78              panel.grid.major = element_blank(),
79              panel.grid.minor = element_blank(),
80              panel.border = element_blank(),
81              legend.position = "none")
82      })
83      quakeSummaryTable <- eventReactive(input$updateButton, {
84          quake.sub <- getQuakes()
85          freq <- table(quake.sub$size)
86          ftab <- data.frame(cbind(names(freq),
87              freq,
88              round(prop.table(freq)*100, 2)
89          ))
90          names(ftab) <- c("Magnitude", "Freq.", "%")
```

```
91          row.names(ftab) <- NULL
92          ftab
93      })
94
95      #  frequency table
96      output$outFrequency <- renderTable(quakeSummaryTable())
97
98      quakeHist <- eventReactive(input$updateButton, {
99          quake.sub <- getQuakes()
100         #  draw quake histogram
101         ggplot(data=quake.sub, aes(x=mag)) +
102         geom_histogram(aes(fill = ..count..), binwidth=0.25,
103             colour = "white") +
104         xlab("Magnitude") + ylab("") +
105         scale_fill_gradient("Count", low = "darkgreen", high = "red") +
106         theme(plot.background = element_rect(fill = "white", colour = NA),
107             panel.background = element_rect(fill = "white", colour = NA),
108             title = element_text(colour="black", size = 13),
109             panel.grid.major = element_blank(),
110             panel.grid.minor = element_blank(),
111             panel.border = element_blank(),
112             legend.position = "right")
113         }
114     )
115
116     #update map
117     output$quakemap <- renderLeaflet(qm())
118     #count total quakes
119     output$countQuake <- renderText(paste("There␣were␣a␣total␣of<b>",
120         nrow(getQuakes()),
121         "</b>␣quakes␣recorded␣from␣<b>",
122         as.character(input$daterange[1]),
123         "</b>to<b>", as.character(input$daterange[2]),
```

```
124         "</b>.<br>"))
125     output$adf <- renderText({
126         paste(as.character(input$daterange))
127     })
128     #update timeline
129     output$magHist <- renderPlot(
130         drawHist()
131     )
132     #update histogram
133     output$quakeHist <- renderPlot(quakeHist())
134     #update table
135     # output$quaketable <- renderDataTable(quakeDataTable())
136     output$quaketable <- renderDataTable(getQuakes())
137 }
```

A.2 ui.R

```
1  library(shinydashboard)
2  library(leaflet)
3
4  header <- dashboardHeader(
5      title = "Korea␣Earthquakes"
6  )
7  sidebar <- dashboardSidebar(
8      sidebarMenu(
9          menuItem("Earthquakes", tabName="dashboard", icon = icon("bullseye"))
10     )
11 )
12 body <- dashboardBody(
13     tags$head(
14         tags$link(rel="stylesheet", type = "text/css", href = "custom.css"),
15         tags$style(type = "text/css", ".irs-grid-text␣{color:␣white;}")
16     ),
17     tabItems(
```

```
18          tabItem(tabName ="dashboard",
19              fluidRow(
20                  column(width = 12,
21                      box(width = NULL, solidHeader = TRUE,
22                          leafletOutput("quakemap", height = 400),
23                          htmlOutput("countQuake", inline = FALSE)
24                      )
25                  ),
26                  column(width=6,
27                      box(title="Select␣Time␣Window",
28                          background = "navy",
29                          # solidHeader = TRUE,
30                          width = NULL,
31                          height = 170,
32                          collapsible=TRUE,
33                          dateRangeInput("daterange", "Select␣date␣range:",
34                              start = "2005-02-01",
35                              end   = as.character(as.Date(Sys.time())),
36                              min = "2005-02-01",
37                              max = as.character(as.Date(Sys.time()))),
38                          actionButton("updateButton", "Update␣graphs")
39                      )),
40                  column(width=6,
41                      box(title="Magnitude␣Range",
42                          background = "navy",
43                          width = NULL,
44                          height = 170,
45                          collapsible=TRUE,
46                              sliderInput("range", "Magnitudes", 2, 5.8,
47                                  value = c(2.0, 5.8),
48                                  step = 0.1, width = "100%"
49                              )
50                      )),
```

```
51                  column(width=12,
52                      box(title='Quake timeline', solidHeader=TRUE,
53                          background = "light-blue",
54                          width = NULL,
55                          collapsible = TRUE,
56                          plotOutput("magHist", height=200)
57                      )
58                  ),
59                  column(width=6,
60                      box(title="Histogram",
61                          background = "light-blue",
62                          solidHeader = TRUE,
63                          width=NULL,
64                          collapsible=TRUE,
65                          plotOutput("quakeHist", height = 200)
66                      )),
67                  column(width=6, #3
68                      box(title="Frequency table",
69                          background = "light-blue",
70                          solidHeader = TRUE,
71                          width=NULL,
72                          collapsible=TRUE,
73                          tableOutput("outFrequency")
74                      ))
75                  ))
76              ))
77
78  dashboardPage(skin='blue',
79      header,
80      sidebar,
81      body
82  )
```