R_Programming_Project

Saurabh Gupta

June 12, 2017

Introduction

The data for this assignment come from the Hospital Compare web site (http://hospitalcompare.hhs.gov) run by the U.S. Department of Health and Human Services. The purpose of the web site is to provide data and information about the quality of care at over 4,000 Medicare-certified hospitals in the U.S. This dataset essentially covers all major U.S. hospitals. This dataset is used for a variety of purposes, including determining whether hospitals should be fined for not providing high quality care to patients (see http://goo.gl/jAXFX for some background on this particular topic).

Question 1

Write a function called best() that takes TWO (2) arguments: (a) the TWO(2)-character abbreviated name of a state; and (b) an outcome name. The function reads the outcome-of-care-measures.csv file and returns a character vector with the name of the hospital that has the best (i.e. LOWEST) 30-day mortality for the specified outcome in that state. The hospital name is the name provided in the Hospital.Name variable. The outcomes can be one of "heart attack", "heart failure", or "pneumonia". The function should use the following template.

```
best <- function(state, outcome) {
          ## Read outcome data
          ## Check that state and outcome are valid
          ## Return hospital name in that state with lowest 30-day death rate
  }
```

The function should check the validity of its arguments. If an invalid state value is passed to best(), the function should throw an error via the stop() function with the exact message "invalid state". If an invalid outcome value is passed to best(), the function should throw an error via the stop() function with the exact message "invalid outcome".

```r
suppressMessages(library(dplyr))

best <- function(State,Outcome){

  #load the data
  mydata <- read.csv("outcome-of-care-measures.csv",header=TRUE,stringsAsFactors = FALSE)

  #checking for valid state and outcome
  if(!(State %in% mydata$State)){
    stop("invalid state")
  }else if(!(Outcome %in% c("heart attack","heart failure","pneumonia"))){
    stop("invalid outcome")
  }


  #data cleaning
  subdata <- select(mydata,Hospital.Name,State,starts_with("Hospital.30.Day.Death"))
  names(subdata) <- gsub("[.]"," ",tolower(names(subdata)))
  names(subdata)[3:length(names(subdata))] <- c("heart attack","heart failure","pneumonia")
  subdata[grepl("Not Available",subdata$`heart attack`),] <- NA
  subdata[grepl("Not Available",subdata$`heart failure`),] <- NA
  subdata[grepl("Not Available",subdata$pneumonia),] <- NA
  subdata[3:5] <- apply(subdata[3:5],2,as.numeric)

  State="SC"
  Outcome="heart attack"

  #solution for the problem
  desiredHospital <- subdata %>%
                     filter(state == State) %>%
                     select(`hospital name`,state,contains(Outcome)) %>%
                     arrange_(.dots = as.name(Outcome)) %>%
                     slice(1)

  desiredHospital$`hospital name`

}

#call the function
best("SC","heart attack")
```

```
## [1] "MUSC MEDICAL CENTER"
```

Question 2

Write a function called rankhospital() that takes THREE (3) arguments: (a) the TWO(2)-character abbreviated name of a state (state); (b) an outcome (outcome); and © the ranking of a hospital in that state for that outcome (num). The function reads the outcome-of-care-measures.csv file and returns a character vector with the name of the hospital that has the ranking specified by the num argument. For example, the call:

```
rankhospital("MD", "heart failure", 5)
```

would return a character vector containing the name of the hospital with the FIFTH (5th) LOWEST THIRTY(30)-day death rate for heart failure. The num argument can take values “best”, “worst”, or an integer indicating the ranking (SMALLER numbers are better). If the number given by num is LARGER THAN the number of hospitals in that state, then the function should return NA. The function should use the following template.

```
rankhospital <- function(state, outcome, num = "best") {
                  ## Read outcome data
                  ## Check that state and outcome are valid
                  ## Return hospital name in that state with the given rank
                  ## THIRTY(30)-day death rate
  }
```

Hospitals that do NOT have data on a particular outcome should be excluded from the set of hospitals when deciding the rankings.

If there is MORE THAN ONE (1) hospital for a given ranking, then the hospital names should be sorted in alphabetical order and the FIRST (1st) hospital in that set should be returned (i.e. if hospitals “b”, “c”, and “f” are tied for a given rank, then hospital “b” should be returned).

The function should check the validity of its arguments. If an invalid state value is passed to rankhospital(), the function should throw an error via the stop() function with the exact message “invalid state”. If an invalid outcome value is passed to rankhospital(), the function should throw an error via the stop() function with the exact message “invalid outcome”. The num variable can take values “best”, “worst”, or an integer indicating the ranking (SMALLER numbers are better). If the number given by num is larger than the number of hospitals in that state, then the function should return NA.

```r
library(dplyr)

rankhospital <- function(State,Outcome,Rank){

  #load the data
  mydata <- read.csv("outcome-of-care-measures.csv",header=TRUE,stringsAsFactors = FALSE)

  #checking for valid state and outcome
  if(!(State %in% mydata$State)){
    stop("invalid state")
  }else if(!(Outcome %in% c("heart attack","heart failure","pneumonia"))){
    stop("invalid outcome")
  }



  #data cleaning
  subdata <- select(mydata,Hospital.Name,State,starts_with("Hospital.30.Day.Death"))
  names(subdata) <- gsub("[.]"," ",tolower(names(subdata)))
  names(subdata)[3:length(names(subdata))] <- c("heart attack","heart failure","pneumonia")
  subdata[grepl("Not Available",subdata$`heart attack`),] <- NA
  subdata[grepl("Not Available",subdata$`heart failure`),] <- NA
  subdata[grepl("Not Available",subdata$pneumonia),] <- NA
  subdata[3:5] <- apply(subdata[3:5],2,as.numeric)


  #solution for the problem
  desiredHospital <- subdata %>%
    filter(state == State) %>%
    select(`hospital name`,state,contains(Outcome)) %>%
    arrange_(.dots = c(as.name(Outcome),as.name("hospital name")))

  if(tolower(Rank)=="best"){
    pos <- 1
  }else if(tolower(Rank) =="worst"){
    pos <- nrow(desiredHospital)
  }else{
    pos <- Rank
  }

  desiredHospital <- slice(desiredHospital,pos)
  desiredHospital$`hospital name`

}

# funcation call
rankhospital("NC","heart attack","worst")
```

```
## [1] "WAYNE MEMORIAL HOSPITAL"
```

Question 3

Write a function called rankall() that takes TWO (2) arguments: (a) an outcome name (outcome); and (b) a hospital ranking (num). The function reads the outcome-of-care-measures.csv file and returns a TWO(2)-column data frame containing the hospital in EACH state that has the ranking specified in num. For example the function call

```
rankall("heart attack", "best")
```

would return a data frame containing the names of the hospitals that are the best in their respective states for THIRTY(30)-day heart attack death rates. The function should return a value for EVERY state (some may be NA). The FIRST (1st) column in the data frame is named hospital, which contains the hospital name, and the SECOND (2nd) column is named state, which contains the TWO(2)-character abbreviation for the state name. The function should use the following template.

```
rankall <-  function(outcome, num = "best") {
                ## Read outcome data
                ## For each state, find the hospital of the given rank
                ## Return a data frame with the hospital names and the (abbreviated)
                ## state name
  }
```

Hospitals that do NOT have data on a particular outcome should be excluded from the set of hospitals when deciding the rankings.

If there is MORE THAN ONE (1) hospital for a given ranking, then the hospital names should be sorted in alphabetical order and the FIRST (1st) hospital in that set should be returned (i.e. if hospitals "b", "c", and "f" are tied for a given rank, then hospital "b" should be returned).

NOTE: For the purpose of this part of the assignment (and for efficiency), your function should NOT call the rankhospital() function from the previous section.

The function should check the validity of its arguments. If an invalid outcome value is passed to rankall(), the function should throw an error via the stop() function with the exact message "invalid outcome". The num variable can take values "best", "worst", or an integer indicating the ranking (SMALLER numbers are better). If the number given by num is larger than the number of hospitals in that state, then the function should return NA.

```
library(dplyr)

rankall <- function(Outcome,Rank){

  #load the data
  mydata <- read.csv("outcome-of-care-measures.csv",header=TRUE,stringsAsFactors = FALSE)

  #checking for valid state and outcome
  if(!(Outcome %in% c("heart attack","heart failure","pneumonia"))){
    stop("invalid outcome")
  }


  #data cleaning
  subdata <- select(mydata,Hospital.Name,State,starts_with("Hospital.30.Day.Death"))
  names(subdata) <- gsub("[.]"," ",tolower(names(subdata)))
  names(subdata)[3:length(names(subdata))] <- c("heart attack","heart failure","pneumonia")
  subdata[grepl("Not Available",subdata$`heart attack`),] <- NA
  subdata[grepl("Not Available",subdata$`heart failure`),] <- NA
  subdata[grepl("Not Available",subdata$pneumonia),] <- NA
  subdata[3:5] <- apply(subdata[3:5],2,as.numeric)

  #solution for the problem
  desiredHospital <- subdata %>%
    select(`hospital name`,state,contains(Outcome)) %>%
    group_by(state) %>%
    arrange_(.dots = c(as.name(Outcome),as.name("hospital name")))


  if(tolower(Rank)=="best"){
    pos <- 1
  }else if(tolower(Rank) =="worst"){
    pos <- do(desiredHospital,n(.))
  }else{
    pos <- Rank
  }

  desiredHospital <- slice(desiredHospital,pos)
  desiredHospital[,c("hospital name","state")]

}

rankall(Rank=4,Outcome="heart attack")
```

```
## Source: local data frame [52 x 2]
## Groups: state [52]
##
##                    `hospital name` state
##                              <chr> <chr>
## 1     ALASKA NATIVE MEDICAL CENTER    AK
## 2               GEORGIANA HOSPITAL    AL
## 3    NEA BAPTIST MEMORIAL HOSPITAL    AR
## 4          FLAGSTAFF MEDICAL CENTER    AZ
## 5      CEDARS-SINAI MEDICAL CENTER    CA
## 6   EXEMPLA LUTHERAN MEDICAL CENTER    CO
## 7             SAINT MARYS HOSPITAL    CT
## 8        WASHINGTON HOSPITAL CENTER    DC
## 9       NANTICOKE MEMORIAL HOSPITAL    DE
## 10   PALM SPRINGS GENERAL HOSPITAL    FL
## # ... with 42 more rows
```