

# Programming Assignment 3 INSTRUCTIONS: Hospital Quality

Mihai M. Craiu

3/7/2020

## Introduction

Download the file Programming Assingment 3 data.zip file containing the data for Programming Assignment 3 from the Coursera web site. Unzip the file in a directory that will serve as your working directory. When you start up R make sure to change your working directory to the directory where you unzipped the data.

The data for this assignment come from the Hospital Compare web site 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.

The Hospital Compare web site contains a lot of data and we will only look at a small subset for this assignment. The zip file for this assignment contains three files:

- outcome-of-care-measures.csv: Contains information about 30-day mortality and readmission rates for heart attacks, heart failure, and pneumonia for over 4,000 hospitals.
- hospital-data.csv: Contains information about each hospital.
- Hospital\_Revised\_Flatfiles.pdf: Descriptions of the variables in each file (i.e the code book).

A description of the variables in each of the files is in the included PDF file named Hospital\_Revised\_Flatfiles.pdf. This document contains information about many other files that are not included with this programming assignment. You will want to focus on the variables for Number 19 (“Outcome of Care Measures.csv”) and Number 11 (“Hospital Data.csv”). You may find it useful to print out this document (at least the pages for Tables 19 and 11) to have next to you while you work on this assignment. In particular, the numbers of the variables for each table indicate column indices in each table (i.e. “Hospital Name” is column 2 in the outcome-of-care-measures.csv file)

## Detailed Instructions

Please download this document for detailed instructions about the assignment

ProgAssignment3.PDF

## Data

The zip file containing the data for this assignment can be downloaded here

Programming Assingment 3 data

## Data Summary

```
outcome <- read.csv("outcome-of-care-measures.csv", colClasses = "character")
#head(outcome)
```

```
str(outcome[,c(2,7,11,17,19,23)])
```

```
## 'data.frame': 4706 obs. of 6 variables:
## $ Hospital.Name : chr "SOUTH"
## $ State : chr "AL"
## $ Hospital.30.Day.Death..Mortality..Rates.from.Heart.Attack : chr "14.3"
## $ Hospital.30.Day.Death..Mortality..Rates.from.Heart.Failure : chr "11.4"
## $ Lower.Mortality.Estimate...Hospital.30.Day.Death..Mortality..Rates.from.Heart.Failure: chr "9.5"
## $ Hospital.30.Day.Death..Mortality..Rates.from.Pneumonia : chr "10.9"
```

```
summary(outcome[,c(2,7,11,17,19,23)])
```

```
## Hospital.Name      State
## Length:4706      Length:4706
## Class :character  Class :character
## Mode :character  Mode :character
## Hospital.30.Day.Death..Mortality..Rates.from.Heart.Attack
## Length:4706
## Class :character
## Mode :character
## Hospital.30.Day.Death..Mortality..Rates.from.Heart.Failure
## Length:4706
## Class :character
## Mode :character
## Lower.Mortality.Estimate...Hospital.30.Day.Death..Mortality..Rates.from.Heart.Failure
## Length:4706
## Class :character
## Mode :character
## Hospital.30.Day.Death..Mortality..Rates.from.Pneumonia
## Length:4706
## Class :character
## Mode :character
```

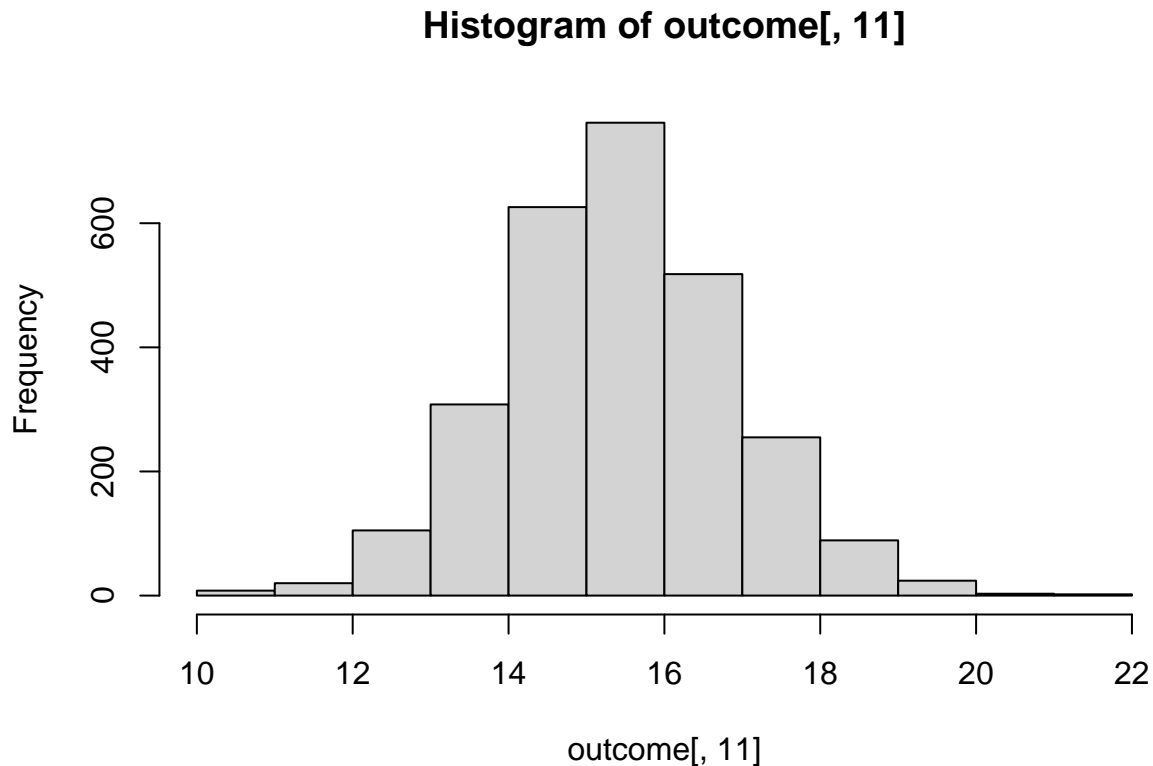
## Part 1: Plot the 30-day mortality rates for heart attack

To make a simple histogram of the 30-day death rates from heart attack (column 11 in the outcome dataset), run

```
outcome[,11] <- as.numeric(outcome[,11])
```

```
## Warning: NAs introducidos por coerción
```

```
## You may get a warning about NA being introduced; that is okay
hist(outcome[,11])
```



## Part 2: Finding the best hospital in a state

Write a function called `best` that take two arguments: the 2-character abbreviated name of a state and 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”. Hospitals that do not have data on a particular outcome should be excluded from the set of hospitals when deciding the rankings.

Handling ties. If there is a tie for the best hospital for a given outcome, then the hospital names should be sorted in alphabetical order and the first hospital in that set should be chosen (i.e. if hospitals “b”, “c”, and “f” are tied for best, then hospital “b” should be returned).

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”.

Here is some sample output from the function.

```
source("best.R")
```

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

```
[1] "CYPRESS FAIRBANKS MEDICAL CENTER"
```

```
best("TX", "heart failure")
```

```
[1] "FORT DUNCAN MEDICAL CENTER"
```

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

```
[1] "JOHNS HOPKINS HOSPITAL, THE"
```

```
best("MD", "pneumonia")
```

```
[1] "GREATER BALTIMORE MEDICAL CENTER"
```

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

```
Error in best("BB", "heart attack") : invalid state
```

```
best("NY", "hert attack")
```

```
Error in best("NY", "hert attack") : invalid outcome
```

Save your code for this function to a file named `best.R`.

## My Solution

```
best <- function(state, outcome){  
  ##Read outcome data  
  data<- read.csv("outcome-of-care-measures.csv", colClasses = "character")  
  fd <- as.data.frame(cbind(data[,2],      # hospital  
                             data[,7],    # state  
                             data[,11],   # heart attack  
                             data[,17],   # heart failure  
                             data[,23]),  # pneumonia  
                      stringsAsFactors = FALSE)  
  colnames(fd) <- c("hospital", "state", "heart attack", "heart failure",  
                   "pneumonia")
```

```

##Check that state and outcome are valid
if(!state %in% fd[, "state"]){
  stop('invalid state')
}else if (!outcome %in% c("heart attack", "heart failure", "pneumonia")){
  stop('invalid outcome')
}else {
  si <- which(fd[, "state"] == state)
  ts <- fd[si,] #extracting data for the called state
  oi <- as.numeric(ts[,eval(outcome)])
  min_val <- min(oi,na.rm = TRUE)
  result <- ts[, "hospital"][which(oi == min_val)]
  output <- result[order(result)]
}
return(output)
}

# example output:
best("SC", "heart attack")

```

```
## Warning in best("SC", "heart attack"): NAs introducidos por coerción
```

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

### Part 3: Ranking hospitals by outcome in a state

Write a function called `rankhospital` that takes three arguments: the 2-character abbreviated name of a state (state), 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 5th lowest 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. Hospitals that do not have data on a particular outcome should be excluded from the set of hospitals when deciding the rankings.

Handling ties. It may occur that multiple hospitals have the same 30-day mortality rate for a given cause of death. In those cases ties should be broken by using the hospital name. For example, in Texas (“TX”), the hospitals with lowest 30-day mortality rate for heart failure are shown here.

```
head(texas)
```

```
Hospital.Name Rate Rank
```

```
3935 FORT DUNCAN MEDICAL CENTER 8.1 1
```

```
4085 TOMBALL REGIONAL MEDICAL CENTER 8.5 2
```

```
4103 CYPRESS FAIRBANKS MEDICAL CENTER 8.7 3
```

3954 DETAR HOSPITAL NAVARRO 8.7 4

4010 METHODIST HOSPITAL,THE 8.8 5

3962 MISSION REGIONAL MEDICAL CENTER 8.8 6

Note that Cypress Fairbanks Medical Center and Detar Hospital Navarro both have the same 30-day rate (8.7). However, because Cypress comes before Detar alphabetically, Cypress is ranked number 3 in this scheme and Detar is ranked number 4. One can use the order function to sort multiple vectors in this manner (i.e. where one vector is used to break ties in another vector).

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  
  ## 30-day death rate  
  
}
```

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".

Here is some sample output from the function.

```
source("rankhospital.R")
```

```
rankhospital("TX", "heart failure", 4)
```

```
[1] "DETAR HOSPITAL NAVARRO"
```

```
rankhospital("MD", "heart attack", "worst")
```

```
[1] "HARFORD MEMORIAL HOSPITAL"
```

```
rankhospital("MN", "heart attack", 5000)
```

```
[1] NA
```

Save your code for this function to a file named rankhospital.R.

## My Solution

```
rankhospital <- function(state, outcome, rank = "best"){  
  ## Read outcome data  
  data <- read.csv("outcome-of-care-measures.csv", colClasses = "character")  
  fd    <- as.data.frame(cbind(data[, 2], # hospital  
                                data[, 7], # state
```

```

        data[, 11], # heart attack
        data[, 17], # heart failure
        data[, 23]), # pneumonia
        stringsAsFactors = FALSE)
colnames(fd) <- c("hospital", "state", "heart attack", "heart failure",
                 "pneumonia")

## Check that state and outcome are valid
if (!state %in% fd[, "state"]) {
  stop('invalid state')
} else if (!outcome %in% c("heart attack", "heart failure", "pneumonia")){
  stop('invalid outcome')
} else if (is.numeric(rank)) {
  si <- which(fd[, "state"] == state)
  ts <- fd[si, ] # extracting dataframe for the called state
  ts[, eval(outcome)] <- as.numeric(ts[, eval(outcome)])
  ts <- ts[order(ts[, eval(outcome)], ts[, "hospital"]), ]
  output <- ts[, "hospital"][rank]
} else if (!is.numeric(rank)){
  if (rank == "best") {
    output <- best(state, outcome)
  } else if (rank == "worst") {
    si <- which(fd[, "state"] == state)
    ts <- fd[si, ]
    ts[, eval(outcome)] <- as.numeric(ts[, eval(outcome)])
    ts <- ts[order(ts[, eval(outcome)], ts[, "hospital"],
                   decreasing = TRUE), ]
    output <- ts[, "hospital"][1]
  } else {
    stop('invalid rank')
  }
}
return(output)
}

# example output:
rankhospital("NC", "heart attack", "worst")

```

```

## Warning in rankhospital("NC", "heart attack", "worst"): NAs introducidos por
## coerción

```

```

## [1] "WAYNE MEMORIAL HOSPITAL"

```

## Part 4: Ranking hospitals in all states

Write a function called `rankall` that takes two arguments: an outcome name (`outcome`) and a hospital ranking (`num`). The function reads the `outcome-of-care-measures.csv` file and returns a 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 30-day heart attack death rates. The function should return a value for every state (some may be NA). The first column in the data frame is named `hospital`, which contains the

hospital name, and the second column is named state, which contains the 2-character abbreviation for the state name. Hospitals that do not have data on a particular outcome should be excluded from the set of hospitals when deciding the rankings.

Handling ties. The rankall function should handle ties in the 30-day mortality rates in the same way that the rankhospital function handles ties.

The function should use the following template.

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

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.

Here is some sample output from the function

```
source("rankall.R")
```

```
head(rankall("heart attack", 20), 10)
```

```
  hospital state
```

```
AK AK
```

```
AL D W MCMILLAN MEMORIAL HOSPITAL AL
```

```
AR ARKANSAS METHODIST MEDICAL CENTER AR
```

```
AZ JOHN C LINCOLN DEER VALLEY HOSPITAL AZ
```

```
CA SHERMAN OAKS HOSPITAL CA
```

```
CO SKY RIDGE MEDICAL CENTER CO
```

```
CT MIDSTATE MEDICAL CENTER CT
```

```
DC DC
```

```
DE DE
```

```
FL SOUTH FLORIDA BAPTIST HOSPITAL FL
```



```
tail(rankall("pneumonia", "worst"), 3)
```

hospital state

WI MAYO CLINIC HEALTH SYSTEM - NORTHLAND, INC WI

WV PLATEAU MEDICAL CENTER WV

WY NORTH BIG HORN HOSPITAL DISTRICT WY

```
tail(rankall("heart failure"), 10)
```

hospital state

TN WELLMONT HAWKINS COUNTY MEMORIAL HOSPITAL TN

TX FORT DUNCAN MEDICAL CENTER TX

UT VA SALT LAKE CITY HEALTHCARE - GEORGE E. WAHLEN VA MEDICAL CENTER UT

VA SENTARA POTOMAC HOSPITAL VA

VI GOV JUAN F LUIS HOSPITAL & MEDICAL CTR VI

VT SPRINGFIELD HOSPITAL VT

WA HARBORVIEW MEDICAL CENTER WA

WI AURORA ST LUKES MEDICAL CENTER WI

WV FAIRMONT GENERAL HOSPITAL WV

WY CHEYENNE VA MEDICAL CENTER WY

Save your code for this function to a file named rankall.R.

## My Solution

```
rankall <- function(outcome, num = "best"){  
  ## Read outcome data  
  data <- read.csv("outcome-of-care-measures.csv", colClasses = "character")  
  fd <- as.data.frame(cbind(data[, 2], # hospital  
                             data[, 7], # state  
                             data[, 11], # heart attack  
                             data[, 17], # heart failure  
                             data[, 23]), # pneumonia  
                      stringsAsFactors = FALSE)  
  colnames(fd) <- c("hospital", "state", "heart attack", "heart failure", "pneumonia")  
  fd[, eval(outcome)] <- as.numeric(fd[, eval(outcome)])  
  
  ## Check that state and outcome are valid  
  
  if (!outcome %in% c("heart attack", "heart failure", "pneumonia")){  
    stop('invalid outcome')  
  } else if (is.numeric(num)) {
```

```

by_state <- with(fd, split(fd, state))
ordered <- list()
for (i in seq_along(by_state)){
  by_state[[i]] <- by_state[[i]][order(by_state[[i]][, eval(outcome)],
                                     by_state[[i]][, "hospital"]), ]
  ordered[[i]] <- c(by_state[[i]][num, "hospital"], by_state[[i]][, "state"][1])
}
result <- do.call(rbind, ordered)
output <- as.data.frame(result, row.names = result[, 2], stringsAsFactors = FALSE)
names(output) <- c("hospital", "state")
} else if (!is.numeric(num)) {
  if (num == "best") {
    by_state <- with(fd, split(fd, state))
    ordered <- list()
    for (i in seq_along(by_state)){
      by_state[[i]] <- by_state[[i]][order(by_state[[i]][, eval(outcome)],
                                           by_state[[i]][, "hospital"]), ]
      ordered[[i]] <- c(by_state[[i]][1, c("hospital", "state")])
    }
    result <- do.call(rbind, ordered)
    output <- as.data.frame(result, stringsAsFactors = FALSE)
    rownames(output) <- output[, 2]
  } else if (num == "worst") {
    by_state <- with(fd, split(fd, state))
    ordered <- list()
    for (i in seq_along(by_state)){
      by_state[[i]] <- by_state[[i]][order(by_state[[i]][, eval(outcome)],
                                           by_state[[i]][, "hospital"],
                                           decreasing = TRUE), ]
      ordered[[i]] <- c(by_state[[i]][1, c("hospital", "state")])
    }
    result <- do.call(rbind, ordered)
    output <- as.data.frame(result, stringsAsFactors = FALSE)
    rownames(output) <- output[, 2]
  } else {
    stop('invalid num')
  }
}
return(output)
}

# example output:
r <- rankall("heart attack", 4)

```

```
## Warning in rankall("heart attack", 4): NAs introducidos por coerción
```

```
as.character(subset(r, state == "HI")$hospital)
```

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