RProgramming Assignment 3 - Hospital Quality

Part 1 - Nothing to Submit

Part 2 - Best

Write a function that returns the hospital with the lowest death rate for a given cause of death.

Format: best <- function(state, outcome){}

Parameters:

1. **The state must be valid.** If not, the function must return the error message via the stop function “invalid state.”
2. **The outcome must be either “heart attack”, “heart failure”, or “pneumonia”.**  If not, the function must return the error message via the stop function “invalid outcome.”
3. **No Information** - the program should exclude hospitals that have no information on the outcome of the provided condition
4. **Ties:** if a state has two or more hospitals that have the highest outcome, then the hospitals should be sorted alphabetically and the first hospital should be selected. (e.g. if hospitals “b”, “c”, and “f” are tied for best, then hospital “b” should be returned)

Psuedocode for best

best <- function(state, outcome){

##Step 1: read the csv

mydata <- read.csv("outcome-of-care-measures.csv", colClasses = "character", na.strings = c("Not Available"))

##make dataframe look at only the relevant info

allinfo <- mydata[ ,c(2,7,11,17,23)]

##Step 2: Create a character vector of allowed conditions

conditions <- c("heart attack", "heart failure", "pneumonia")

##Step 3 - Create a character vector of all states and provinces in data

loc <- as.vector(unique(mydata$State))

##Step 4 - create 2 big IF loops that covers the entire function. It should be something like IF State and Conditions are valid, then continue with the meat of the function. If not, return STOP

if(outcome %in% conditions){

if(state %in% loc){

###meat of program goes here

#Step 5 - subset dataframe by state

mystate <- subset(allinfo, allinfo$State == state)

## Step 6 - subset dataframe by condition

##Method: create a character vector assigning conditions to the column numbers

##colnum <- c("heart attack" = "mystate[,11]", "heart failure" = "mystate[,17]", "pneumonia" = "mystate[,23]")

} else {

stop("invalid state")

}

} else {

stop("invalid outcome")

}}

}

Final:

best <- function(state, outcome){

##read in our data frame, states, and conditions

mydata <- read.csv("outcome-of-care-measures.csv", colClasses = "character", na.strings = c("Not Available"))

allinfo <- mydata[ ,c(2,7,11,17,23)]

conditions <- c("heart attack", "heart failure", "pneumonia")

loc <- as.vector(unique(mydata$State))

if(outcome %in% conditions){

##print("valid outcome")

if(state %in% loc){

##print("valid state")

###meat of program goes here

mystate <- subset(allinfo, allinfo$State == state)

##create a character vector that assigns possible outcomes to the correct column entries in mystate

##colnum <- c("heart attack" = "mystate[,11]", "heart failure" = "mystate[,17]", "pneumonia" = "mystate[,23]")

##Now, create another if else loop to see if we can do something with the outcomes

if(outcome == "heart attack"){

##create a dataframe of just the heart attack

info <- mystate[,c(1,2,3)]

good <- complete.cases(info)

goodinf <- info[good,]

##work on this to get the order correct

a <- goodinf[order(as.numeric(goodinf[,3]),goodinf[,1]),]

print(a[1, 1])

}else if(outcome == "heart failure"){

info <- mystate[,c(1,2,4)]

good <- complete.cases(info)

goodinf <- info[good,]

##work on this to get the order correct

a <- goodinf[order(as.numeric(goodinf[,3]),goodinf[,1]),]

print(a[1, 1])

}else if(outcome == "pneumonia"){

info <- mystate[,c(1,2,5)]

good <- complete.cases(info)

goodinf <- info[good,]

##work on this to get the order correct

a <- goodinf[order(as.numeric(goodinf[,3]),goodinf[,1]),]

print(a[1, 1])

}

} else {

stop("invalid state")

}

} else {

stop("invalid outcome")

}

##print(str(mystate))

}

Part 3 – Rank hospital

Write a function called “rankhospital” that takes the state, outcome, and rank requested and gives back the hospital with the corresponding hospital with the given rate for that state. For example, rankhospital(“MD”, “heart attack”, 5) will give back the 5th best hospital for 30 day heart attack survival in Maryland. The rank can also be “best” or “worst”. Make the smaller numbers ranked be the better. If the number given is larger than the number of measurable hospitals in the state, the function should give back NA. Ties in rank should be broken alphabetically.

An entry of an invalid state should be returned with the stop function “invalid state”

An entry of an invalid outcome should be returned with the stop function “invalid outcome”

Structure:

rankhospital <- function(state, outcome, num = “best”){

## read the outcome data

##check to see if the state and outcome are valid

##Return hospital name in that state with the given rank

}

Notes: This function can recycle must of the best function. We’ll just add another parameter num = “best

**Parameters: num = “best” must be either “best”, “worst” or a number**

Code snippet to accomplish this

Let “goodinf” be the data frame cleaned of NA’s and containing the hospital, ordered by alphabetical, and outcome of the previous.

Rank <- c(1:nrow(goodinf))

worst <- length(rank)

goodrank <- cbind(goodinf, rank)

if (num > worst){

print(“NA”)

} else {print(goodrank[num, 1])}

**FINAL**

rankhospital <- function(state, outcome, num = "best"){

##read in our data frame, states, and conditions

mydata <- read.csv("outcome-of-care-measures.csv", colClasses = "character", na.strings = c("Not Available"))

allinfo <- mydata[ ,c(2,7,11,17,23)]

conditions <- c("heart attack", "heart failure", "pneumonia")

loc <- as.vector(unique(mydata$State))

if(num == "best"){num <- 1}

if(outcome %in% conditions){

if(state %in% loc){

###meat of program goes here

mystate <- subset(allinfo, allinfo$State == state)

##Now, create another if else loop to see if we can do something with the outcomes

if(outcome == "heart attack"){

##create a dataframe of just the heart attack

info <- mystate[,c(1,2,3)]

good <- complete.cases(info)

goodinf <- info[good,]

##work on this to get the order correct

a <- goodinf[order(as.numeric(goodinf[,3]),goodinf[,1]),]

rank <- c(1:nrow(a))

w <- length(rank)

if(num == "worst"){num <- w}

goodrank <- cbind(a, rank)

if(num > w){

print("NA")

}else{

print(goodrank[num,1])

}

}else if(outcome == "heart failure"){

info <- mystate[,c(1,2,4)]

good <- complete.cases(info)

goodinf <- info[good,]

##work on this to get the order correct

a <- goodinf[order(as.numeric(goodinf[,3]),goodinf[,1]),]

rank <- c(1:nrow(a))

w <- length(rank)

if(num == "worst"){num <- w}

goodrank <- cbind(a, rank)

if(num > w){

print("NA")

}else{

print(goodrank[num,1])

}

}else if(outcome == "pneumonia"){

info <- mystate[,c(1,2,5)]

good <- complete.cases(info)

goodinf <- info[good,]

##work on this to get the order correct

a <- goodinf[order(as.numeric(goodinf[,3]),goodinf[,1]),]

rank <- c(1:nrow(a))

w <- length(rank)

if(num == "worst"){num <- w}

goodrank <- cbind(a, rank)

if(num > w){

print("NA")

}else{

print(goodrank[num,1])

}

}

} else {

stop("invalid state")

}

} else {

stop("invalid outcome")

}

##print(str(mystate))

}

**Part 4 - Ranking Hospitals in all States**

**Summary:**

Write a function called “rankall” that takes the argument “outcome” and the hospital ranking number “num” and returns a two-column dataframe with the state and name of the hospital of the corresponding rank. For example, the expression rankall(“heart attack”, 1) should return back a data frame with 50 hospitals and their state name that have the best outcome for heart attacks.

**Parameters:**

If a stae doesn’t include the ranking specified, it should return back NA in the dataframe for that state’s entry (for example,”rankall(“heart attack”, 300)” should return mostly NAs for hospital and “state” for the state).

If “num” is written as “best”, the function should return the #1 ranked hospital in the state

If “num” is written as “worst”, the function should return the worst ranking hospital per state, regardless of number of ranked hospitals in each state.

Ties: the lists should be sorted first by rank, then alphabetically by hospital. If two hospitals have the same rank, the “better” one is to be considered the one that is first alphabetically.

Validity testing: The state must be valid. If not, the function must return the error message via the stop function “invalid state.” The outcome must be either “heart attack”, “heart failure”, or “pneumonia”. If not, the function must return the error message via the stop function “invalid outcome.”

**Form:**

rankall <- function(outcome, num = "best") {

## Read outcome data

## Check that state and outcome are valid

## For each state, find the hospital of the given rank

## Return a data frame with the hospital names and the

## (abbreviated) state name

}

This assignment can have a similar format as the other one. However, I’ll have to create an empty data frame for the entries to go into, and a for loop that populates the data frame. I can take out the state IF statement, which I hope saves on execution time

**Psuedocode:**

rankall <- function(outcome, num = “best”){

##read in our data frame, states, and conditions

mydata <- read.csv("outcome-of-care-measures.csv", colClasses = "character", na.strings = c("Not Available"))

allinfo <- mydata[ ,c(2,7,11,17,23)]

conditions <- c("heart attack", "heart failure", "pneumonia")

if(num == "best"){num <- 1}

if(outcome %in% conditions){

~~if(state %in% loc){~~

###meat of program goes here

if(outcome == "heart attack"){

##create a dataframe of just the heart attack

}else if(outcome == "heart failure"){

##dostuff

}else if(outcome == "pneumonia"){

info <- mystate[,c(1,2,5)]

good <- complete.cases(info)

goodinf <- info[good,]

##work on this to get the order correct

a <- goodinf[order(as.numeric(goodinf[,3]),goodinf[,1]),]

print(a[1, 1])

}

} else {

stop("invalid state")

}

} else {

stop("invalid outcome")

}

##print(str(mystate))

}

}