Programming Assignment 3

Computing for Data Analysis

Introduction

Download the file `ProgAssignment3-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 (`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).

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).

1 Plot the 30-day mortality rates for heart attack

Read the outcome data into R via the `read.csv` function and look at the first few rows.

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

There are many columns in this dataset. You can see how many by typing `ncol(outcome)` (you can see the number of rows with the `nrow` function). In addition, you can see the names of each column by typing `names(outcome)` (the names are also in the PDF document.

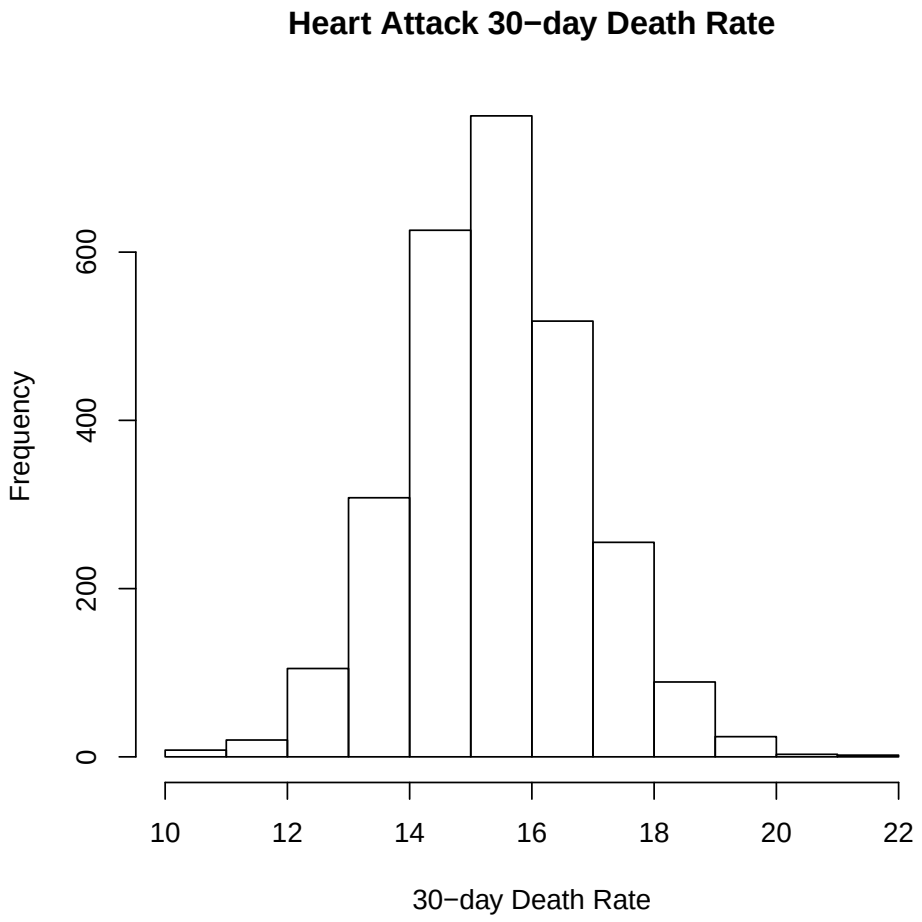
To make a simple histogram of the 30-day death rates from heart attack (column 11 in the outcome dataset), run

```
> outcome[, 11] <- as.numeric(outcome[, 11])
> ## You may get a warning about NAs being introduced; that is okay
> hist(outcome[, 11])
```

Because we originally read the data in as character (by specifying `colClasses = "character"` we need to coerce the column to be numeric. You may get a warning about NAs being introduced but that is okay. This code creates a histogram of the death rates but could benefit from some better labelling.

1. Add a label to the x-axis that says "30-day Death Rate"
2. Add a title for the histogram that says "Heart Attack 30-day Death Rate"

Your final figure should look like the figure below.



There is nothing to submit for this part of the assignment.

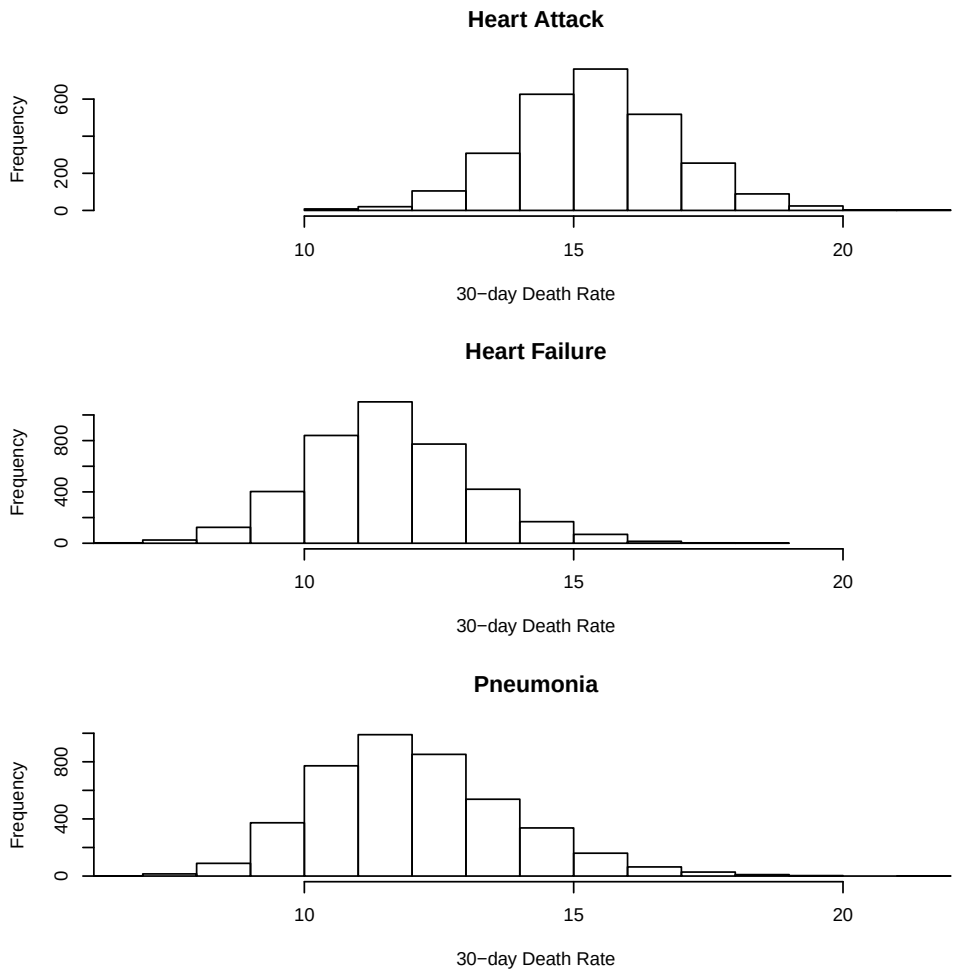
2 Plot the 30-day mortality rates for heart attack, heart failure, and pneumonia

If you haven't already, read in the `outcome-of-care-measures.csv` dataset using the code specified above.

1. Identify which columns of the data frame contain the 30-day death rate from heart attack, heart failure, and pneumonia.
2. Coerce these columns to be numeric using the `as.numeric` function as above. You may receive warnings about NAs but that is okay.

3. Make histograms of the death rates for each outcome and put the histograms on the same plot window. This can be done by running `par(mfrow = c(3, 1))` before calling `hist`. This sets the plot window to have 3 rows and 1 column.
4. For each plot (there should be three plots, one for each outcome) make sure the x-axis label is "30-day Death Rate".
5. For each plot, set the title of the plot to be the outcome (i.e. heart attack, heart failure, or pneumonia).
6. Each time you call hist, a new plot is constructed using the data to be plotted. However, this makes it difficult to compare histograms across outcomes. Set all of the histograms to have the same numerical range on the x-axis by using the `xlim` argument. You can calculate the range of a vector of numbers by using the `range` function.

Your final figure should like the figure below.



Try the following variations on this plot:

1. Instead of plotting the histograms on top of each other, plot them all in a row, side by side.
2. Using the `median` and the `abline` function, draw a vertical line on each histogram at the location of the median for that outcome.
3. In the title of each histogram, put in parentheses the mean death rate by adding ($\bar{X}$ =??) where ?? is the actual mean for that outcome. Consult the help page for `plotmath` to see how to get the $\bar{X}$ to appear on the plot.

4. Add a smooth density estimate on top of the histogram. To do this you need to use the `density` function and you need to set `prob=TRUE` when calling `hist`.

There is nothing to submit for this part of the assignment.

3 Plot 30-day death rates by state

The `outcome-of-care-measures.csv` file contains information about what state each hospital is located in (in the `State` variable). The goal of this part is to plot the hospital 30-day death rates by state.

If you have not done so yet, read in the outcome data to R and coerce the 30-day death rate for heart attack to be numeric.

```
> outcome <- read.csv("outcome-of-care-measures.csv", colClasses = "character")
> outcome[, 11] <- as.numeric(outcome[, 11])
```

First, check to see how many hospitals are included in the dataset by state. We want to remove some states where there are very few hospitals. You can use the `table` function to count the number of observations in each state.

```
> table(outcome$State)
```

```
 AK  AL  AR  AZ  CA  CO  CT  DC  DE  FL  GA  GU  HI  IA  ID  IL  IN  KS  KY  LA
 17  98  77  77 341  72  32   8   6 180 132   1  19 109  30 179 124 118  96 114
 MA  MD  ME  MI  MN  MO  MS  MT  NC  ND  NE  NH  NJ  NM  NV  NY  OH  OK  OR  PA
 68  45  37 134 133 108  83  54 112  36  90  26  65  40  28 185 170 126  59 175
 PR  RI  SC  SD  TN  TX  UT  VA  VI  VT  WA  WI  WV  WY
 51  12  63  48 116 370  42  87   2  15  88 125  54  29
```

Subset the original dataset and exclude states that contain less than 20 hospitals. Name this new subsetted dataset `outcome2`.

A basic boxplot of the death rates by state can be made running the following code.

```
> death <- outcome2[, 11]
> state <- outcome2$State
> boxplot(death ~ state)
```

Add the following aspects to the plot

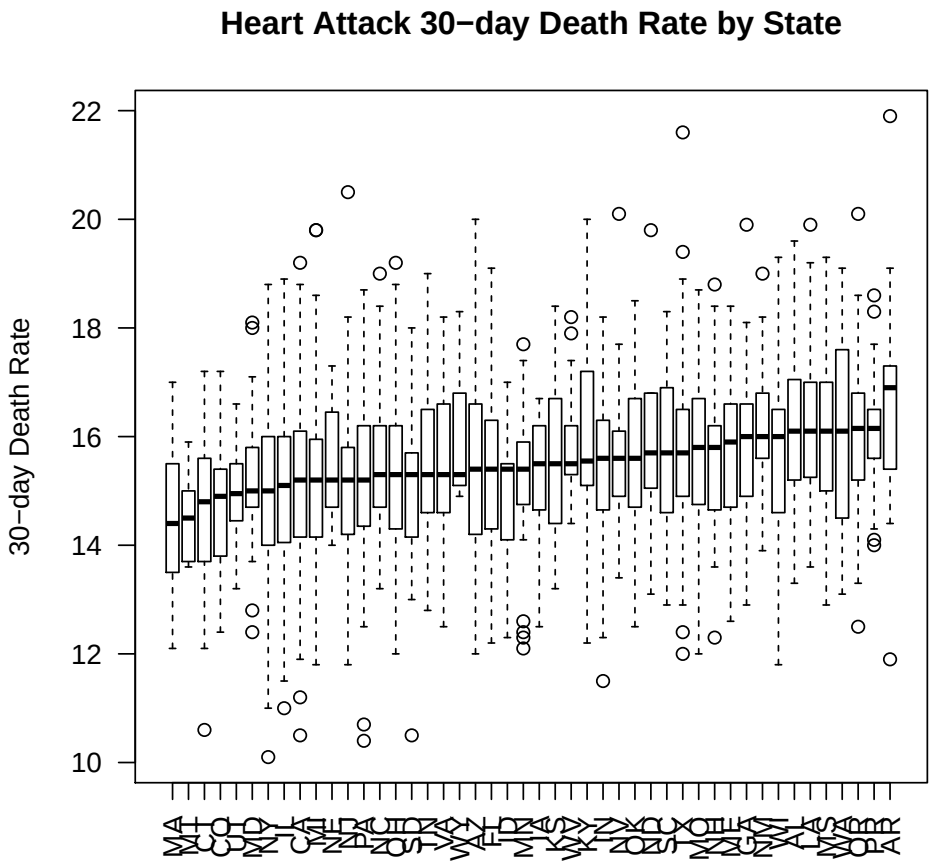
1. Set the y-axis label to say "30-day Death Rate"
2. Set the title of the plot to be "Heart Attack 30-day Death Rate by State"
3. Set the x- and y-axis tick labels to be perpendicular to the axis so that the abbreviated names of all the states will appear on the plot. Use the `par` function to set this.
4. **Challenge**: Sort the states by their median 30-day death rate and plot the boxplots in order of their median rate. Note that the `boxplot` function also accepts a list as its first argument in addition to a formula.

The final figure is shown on the next page.

You may also want to try

1. Shrink the x-axis tick labels so that the abbreviated state names do not overlap each other
2. **Challenge**: Alter the x-axis tick labels so that they include the number of hospitals in that state in parentheses. For example, the label for the state of Connecticut would be `CT (32)`. You will need the `axis` function and when you call the `boxplot` function you will want to set the option `xaxt` to be "n".

There is nothing to submit for this part of the assignment.



4 Plot 30-day death rates and numbers of patients

The **lattice** package can be used to plot relationships while conditioning on various factor variables. The goal of this part is the plot the relationship between the number of patients a hospital sees for a certain outcome and the 30-day death rate for that outcome. The hypothesis is that the more patients a hospital sees, the better the outcome for the patients. We are going to examine this relationship by the hospital ownership type.

First we need to read in the outcome data and the hospital data.

```
> outcome <- read.csv("outcome-of-care-measures.csv", colClasses = "character")
> hospital <- read.csv("hospital-data.csv", colClasses = "character")
```

Then we are going to want to merge the two datasets together to match the `Hospital.Ownership` variable to the death rate data.

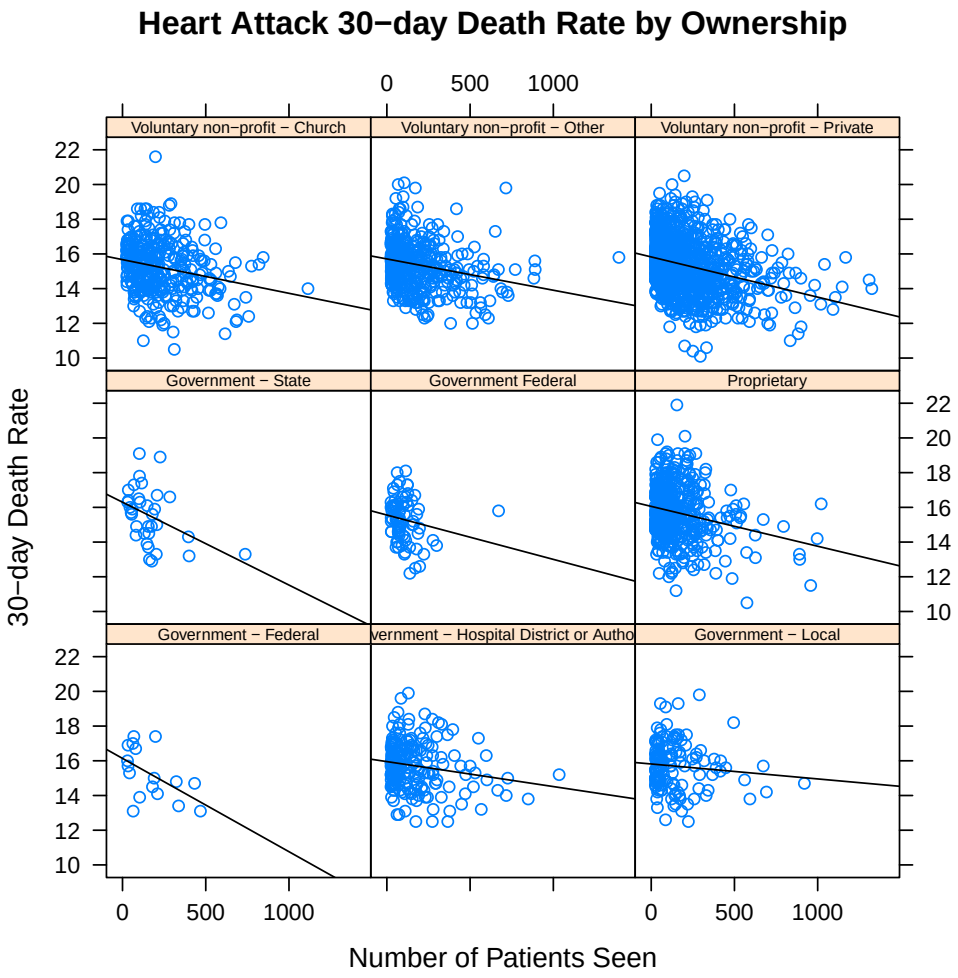
```
> outcome.hospital <- merge(outcome, hospital, by = "Provider.Number")
```

From here, we can create the relevant variables that we want to plot.

```
> death <- as.numeric(outcome.hospital[, 11])  ## Heart attack outcome
> npatient <- as.numeric(outcome.hospital[, 15])
> owner <- factor(outcome.hospital$Hospital.Ownership)
```

1. Use the `xyplot` function in the **lattice** package to make a plot of the relationship between 30-day death rate for heart attack versus the number of patients seen. The number of patients should be on the x-axis. Make sure you run `library(lattice)` before calling `xyplot`.
2. Set the x-axis label to be "Number of Patients Seen"
3. Set the y-axis label to be "30-day Death Rate"
4. Set the title of the plot to be "Heart Attack 30-day Death Rate by Ownership"
5. In each panel of the plot, add a linear regression line highlighting the relationship between number of patients seen and the death rate. Use the `panel.lmline` function for this.

The final plot is below.



There is nothing to submit for this part of the assignment.

5 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`.

Use the submit script provided to submit your solution to this part. There are 3 tests that need to be passed for this part of the assignment.

6 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 `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("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`.

Use the submit script provided to submit your solution to this part. There are 4 tests that need to be passed for this part of the assignment.

7 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 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
}
```

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                                <NA>    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                                <NA>    DC
DE                                <NA>    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`.

Use the submit script provided to submit your solution to this part. There are 3 tests that need to be passed for this part of the assignment.