Problem Set 2

Advanced Forecasting, UNCC, 2019

Ercan Karadas

Due: February 6, 2019

Exercise 1

Store the values `-20, -15, -5, 8, 12, 9, 2, 23, 19` into the variable `x`.

a. Use the R command `sum` to verify that the sum of the values is 33.
b. Compute an average by using the `R` command `mean`?
c. Compute the average with using the `R` command `sum`?
d. Use `R` to sum the positive values in `x`.
e. Use the `which` command to get the average of the values ignoring the largest value.
f. Speculate about the values corresponding to the command `x[abs(x)>=8 & x<8]`. Verify your speculation running this R command.

Exercise

Let `x = c(1, 8, 2, 6, 3, 8, 5, 5, 5, 5)`

a. Describe two different R commands for summing the values in `x` ignoring the value 2 stored in `x[3]` and the value 3 stored in `x[5]`.

b. Use two different R commands to sum all of the values not equal to 5.

c. Use a single R command to change all values equal to 8 to 7.

Exercise

a. Create a 10×5 matrix M whose elements are random draws from a normal distribution with mean 5 and variance 2.

b. Create another matrix N of the same size which contains all zeros, except 5 NA and locations of these NAs are randomly determined (i.e. in the sense that each time you run your code their locations are expected to change).

c. Using M and N generate a random matrix from $N(5, 2)$ which contains 5 NA values that are arbitrarily located in the matrix.

d. Describe how the R function `is.na` can be used to eliminate the rows with missing values.

Exercise

R has a built-in data set called `chickwts`, which is stored in a data frame with two columns. The first column contains the weight of chicks, and the second column indicates the type of feed they received, one of which is labeled `horsebean`. Use R to compute the average weight among chicks that were fed horsebean.

Exercise

a. Create a vector named `my_vec` containing the integers 1 through 100 and then divide each element of `my_vec` by 3 and store the result as `my_vec2`. (Your answer should contain two lines of R commands).

b. Compute the average of the vector `my_vec` you created before without using built-in R function `mean`.

c. Create a vector named `my_vec3` containing the elements of `my_vec` that are between 20 and 35. (Your answer should contain a single line of R commands)

Exercise

Briefly explain what would be the output of following R command: `mean(rnorm(1000))`

%a. Create a 2 by 3 matrix named `my_matrix` containing 6 random draws from the standard normal distribution. (Your answer should contain one line of R commands)

Exercise

Make a script file which constructs two random normal vectors of length 10. Call these vectors `x1` and `x2`. Make a data frame called `T` with two columns (called `a` and `b`) containing respectively `x1` and `x1+x2`.

Exercise

When the function `head` called on the data frame `Orange` it produces the following output:

```
> head(Orange, n=2)
  Tree  age circumference
1    1  118            30
2    1  484            58
```

Write the command that adds up Y (defined as `circumference`) and X (defined as square root of `age`). (Your answer should contain at most three lines of R commands)

Exercise

`Orange` is a data frame with two numeric variables `circumference` and `age`. Create a dummy variable that is 0 when `age` is less than or equal to 900 and 1, otherwise.

Exercise

Put all even integers from 30 to 89 in a vector named P and then in a matrix with 6 rows and 5 columns named Q.

Exercise

Declare a function in R, named `my_function` which takes a vector, say `x`, as input and returns the sum of the elements in the vector and the mean of the values in the vector. Also, make sure that this function returns the message "Data is not numeric!" when you feed in a non-numeric vector.

Exercise

Using a `for` loop in R, write a script that produces the sum of the first $n = 40$ integers.

Exercise

R has a built-in data set called `ChickWeight`. Verify that the R command

```
mean(ChickWeight[,1])
```

returns 121.8 but that the command

```
mean(ChickWeight[,3])
```

returns `NA` and a warning message even though the values in column 3 appear to be numeric. The reason for the warning message is that column 3 is stored as a factor variable. Arithmetic operations can only by performed on numeric or logical variables. Verify that

```
mean(as.numeric(ChickWeight[ ,3]))
```

returns 26.26.

Exercise

The final exam scores for 15 students are

$$73, 74, 92, 98, 100, 72, 74, 85, 76, 94, 89, 73, 76, 99$$

Compute the mean, 20% trimmed mean, and median using R.

Exercise

Let $\{x_1, x_2, \ldots, x_n\}$ be a sample and suppose that we declare x_i an *outlier* if it is more than two standard deviation of the mean, i.e.

$$\frac{|x_i - \overline{X}|}{s} > 2\,.$$

For the values

$$20, 121, 132, 123, 145, 151, 119, 133, 134, 130, 200$$

write an R command to determine whether any outliers exist.

Exercise

Importing data from a plain text file can be illustrated with an example of a data set available on the website "Data and Story Library" (DASL). The Massachusetts lunatics data is available at http://lib.stat.cmu.edu/DASL/Datafiles/lunaticsdat.html.

a. Saved the data as `lunatics.txt` in a folder named "R Practice" on your desktop.

b. Use the `read.table` function to read the file into a data frame.

c. The `str` (structure) function provides a quick check that 14 observations of six variables.

d. Convert `lunatics.txt` into `lunatics.csv` and save in the same folder.

e. Import `lunatics.csv` using the package `readr` and compute the total population of 14 counties.

Exercise

Many of the interesting data sets that one may wish to analyze are available on a web page. R provides an easy way to access data from a file on the internet using the URL of the web page. The function `read.table` can be used to input data directly from the internet.

a. The data file `PiDigits.dat`, located at http://www.itl.nist.gov/div898/strd/univ/data/PiDigits.dat, contains the first 5000 digits of the mathematical constant π. Use `read.table` to save the data in the folder "R Practice".

b. Redo the same thing but now skip the first 50 digits and save as `PiDigitsSkipped.csv`.

c. Using `head` command print some of the data on the console.

d. Are the digits of π uniformly distributed? Using `table` command compute the relative frequencies of each digit.

e. Use `barplot` to visualize the tabulated data you obtained above.

Exercise

Let's say you have a data frame named `mydata`, with variables `x1` and `x2`, and you want to create a new variable `sumx` that adds these two variables and a new variable called `meanx` that averages the two variables. Furthermore, you want to add these new variables to the original data frame. Here is how you do it:

```
# Prepare data
mydata <- data.frame(x1 = c(2, 2, 6, 4),
                     x2 = c(3, 4, 2, 8))
```

a. Method 1: use `$` to append these two variables to the data frame directly

b. Method 2: use `attach()` function

c. Method 3: use `transform()` function

Exercise

Suppose you have the following data set

```
manager <- c(1, 2, 3, 4, 5)
date <- c("10/24/08", "10/28/08", "10/1/08", "10/12/08", "5/1/09")
country <- c("US", "US", "UK", "UK", "UK")
gender <- c("M", "F", "F", "M", "F")

age <- c(32, 45, 25, 39, 99)
        q1 <- c(5, 3, 3, 3, 2)
        q2 <- c(4, 5, 5, 3, 2)
        q3 <- c(5, 2, 5, 4, 1)
        q4 <- c(5, 5, 5, NA, 2)
        q5 <- c(5, 5, 2, NA, 1)

leadership <- data.frame(manager, date, country, gender, age,
                                 q1, q2, q3, q4, q5, stringsAsFactors=FALSE)
```

a. Recode the value 99 for `age` to indicate that the value is missing

b. Let's say you want to recode the ages of the managers in the leadership dataset from the continuous variable age to the categorical variable `agecat` (Young, Middle Aged, Elder).

c. Note that `agecat` is a character variable. Turn it into an ordered factor.

d. Let's say you want to change the variable `manager` to `managerID` and `age` to `ages`.

e. Suppose you have data `y <- c(1, 2, 3, NA)`. Then `is.na(y)` returns `c(FALSE, FALSE, FALSE, TRUE)`. What would be the output of the following command: `is.na(leadership[4:5, 6:10])`

f. Save the `date` variable as a date variable

Exercise

Explain why `sum(c(1, -2, NA, 3))` would return `NA`, but not `sum(c(1, -2, NA, 3), na.rm=TRUE)`?

Exercise

You can remove any observation with missing data by using the `na.omit()` function. It deletes any rows with missing data. Remove the row with NA values from `leadership` data frame.

Exercise

(Counting Missing Values) Create a vector of lenght 100 with every third element missing (namely `NA`) by only using vector operations. Then count the number of missing values by utilizing only the following two function `is.na` and `sum`.

Exercise

(Leadership Example continued) Suppose you have the following data

```
leadershipOriginal <- leadership
```

a. Create a new dataset `newdata` containing rows sorted from youngest manager to oldest manager.

b. Sorts the rows into female followed by male, and youngest to oldest within each gender.

c. Sorts the rows by gender, and then from oldest to youngest manager within each gender.

Exercise

Let's say you have a data frame named `mydata`, with variables `x1` and `x2`, and you want to create a new variable `sumx` that adds these two variables and a new variable called `meanx` that averages the two variables. If you use the code

```
sumx  <-  x1 + x2
meanx <- (x1 + x2)/2
```

What would you get?

You would get an error. The right way of doing it:

```
sumx  <-  mydata$x1 + mydata$x2
meanx <- (mydata$x1 + mydata$x2)/2
```

Exercise (sorting)

Suppose you have the following data

```
manager <- c(1, 2, 3, 4, 5)
date <- c("10/24/08", "10/28/08", "10/1/08", "10/12/08", "5/1/09")
country <- c("US", "US", "UK", "UK", "UK")
gender <- c("M", "F", "F", "M", "F")

age <- c(32, 45, 25, 39, 99)
        q1 <- c(5, 3, 3, 3, 2)
        q2 <- c(4, 5, 5, 3, 2)
        q3 <- c(5, 2, 5, 4, 1)
        q4 <- c(5, 5, 5, NA, 2)
        q5 <- c(5, 5, 2, NA, 1)

leadershipOriginal <- data.frame(manager, date, country, gender, age,
                                 q1, q2, q3, q4, q5, stringsAsFactors=FALSE)
```

a. Create a new dataset `newdata` containing rows sorted from youngest manager to oldest man- ager.

```
leadership <- leadershipOriginal
   newdata <- leadership[order(leadership$age),]
   newdata
```

b. sorts the rows into female followed by male, and youngest to oldest within each gender.

```
   newdata <- leadership[order(leadership$gender, leadership$age),]
   newdata
```

c. Sorts the rows by gender, and then from oldest to youngest manager within each gender.

```
   newdata <- leadership[order(leadership$gender, -leadership$age),]
   newdata
```

Exercise (merging datasets)

Suppose we have these two data sets. Combine them in a meaningful way.

```
names_1 <- c("A", "B", "C")
names_2 <- c("A", "C", "B")
gender <- c("M", "F", "F")

age <- c(32, 45, 25)

dataFrame_1 <- data.frame(names_1, gender)
dataFrame_2 <- data.frame(names_2, age)
dataFrame_1
dataFrame_2

dataFrame_1new <- dataFrame_1[order(dataFrame_1$names_1),]
names(dataFrame_1new)[1] <- "names"

dataFrame_2new <- dataFrame_2[order(dataFrame_2$names_2),]
names(dataFrame_2new)[1] <- "names"

dataFrameOriginal <- merge(dataFrame_1new, dataFrame_2new, by = "names")
```

a. Suppose you want to add another variable to the data frame by using `cbind`:

```
names <- c("A", "B", "C")
income <- c(100, 85, 125)

dataFrame_3 <- data.frame(names, income)

dataFrame <- cbind(dataFrameOriginal, dataFrame_3["income"])
dataFrame
```

b. Suppose you want to add another observation to the data frame by using `rbind`:

```
dataFrame_4 <- data.frame(names = c("D"), gender = c("F"), age = c(44))
dataFrame_4

dataFrame <- rbind(dataFrameOriginal, dataFrame_4)
dataFrame
```

c. Suppose you want to add another observation to the data frame by using `rbind` but this time two data frames have different colums:

```
dataFrame_5 <- data.frame(names = c("D"), gender = c("F"), age = c(44), occupation = c("teacher"))
dataFrame_5

# method 1: append the dataFrameOriginal
dataFrame <- cbind(dataFrameOriginal, occupation = c(NA))

dataFrame <- rbind(dataFrame, dataFrame_5)
dataFrame
```

```
dataFrame_5 <- data.frame(names = c("D"), gender = c("F"), age = c(44), occupation = c("teacher"))
dataFrame_5

# method 2: delete the NA column
dataFrame <- rbind(dataFrameOriginal, dataFrame_5[1:3])
dataFrame
```

Exercise (subsetting)

Selects variables q1, q2, q3, q4, and q5 from the leadership data frame and saves them to the data frame newdata.

```
leadership <- leadershipOriginal
# method 1
newdata <- leadership[, c(6:10)]
newdata

# method 2
myvars <- c("q1", "q2", "q3", "q4", "q5")
newdata <- leadership[myvars]
newdata

# method 3: using paste() function
myvars <- paste("q", 1:5, sep = "")
```

```
newdata <- leadership[myvars]
newdata
```

Exercise (dropping variables)

Drop q3 and q4 from the leadership data frame and saves them to the data frame newdata.

```
leadership <- leadershipOriginal
# method 1
newdata <- leadership[, c(-8,-9)]
newdata

# method 2
myvars <- names(leadership) %in% c("q3", "q4")
newdata <- leadership[!myvars]

# method 3
leadership$q3 <- leadership$q4 <- NULL
leadership
```

Exercise (selecting observations)

Again use `leadership` data.

a. select rows 1 through 3

```
leadership <- leadershipOriginal
newdata <- leadership[1:3,]
newdata
```

b. select observations where men over 30

```
leadership <- leadershipOriginal
newdata <- leadership[leadership$gender == "M" & leadership$age > 30, ]
newdata
```

c. select observations where men over 30 and be careful with comma. Figure out what is going on below

```
leadership <- leadershipOriginal
newdata <- leadership[leadership$gender == "M" & leadership$age > 30]
newdata
```

d. limit your analyses to observations collected between January 1, 2009 and December 31, 2009.

```
leadership <- leadershipOriginal

# Converts the date values read in originally as character values to date values using the format mm/dd
leadership$date <- as.Date(leadership$date, "%m/%d/%y")

# Create starting and ending dates
startdate <- as.Date("2009-01-01")
enddate <- as.Date("2009-10-31")

# Selects cases meeting your desired criteria
newdata <- leadership[which(leadership$date >= startdate &
           leadership$date <= enddate), ]
```

```
newdata
```

Exercise (subset() function)

The `subset()` function is probably the easiest way to select variables and observations.

a. Selects all rows that have a value of age greater than or equal to 35 or less than 24. Keeps variables q1 through q4.

```
leadership <- leadershipOriginal

newdata <- subset(leadership, age >= 35 | age < 24,
                  select=c(q1, q2, q3, q4))
newdata
```

b. Selects all men over the age of 25, and keeps variables gender through q4 (gender, q4, and all columns between them)

```
leadership <- leadershipOriginal

newdata <- subset(leadership, age > 25 & gender == "M",
                  select=c(gender:q4))
newdata
```

Exercise (sample() function)

The `sample()` function enables you to take a random sample (with or without replacement) of size n from a dataset.

Take a random sample of size 3 from the leadership dataset

```
leadership <- leadershipOriginal
mysample <- leadership[sample(1:nrow(leadership), 3, replace=FALSE),]
mysample
```