Markdown

Louise Lai

July 19, 2018

Q1

Create the following vectors, populated with information about the four MSAN boot-camp classes, create a table summarizing the type and class for each vector

```
courseNum <- c("593", "501", "504", "502")
courseName <- c("Exploratory Data Analysis", "Computation for Analytics",
                "Review of Probability and Statistics", "Linear Algebra")
courseProf <- c("Paul Intrevado", "Terrence Parr", "Jeff Hamrick", "Xuemei Chen")
enrolled <- as.logical(c(1, 1, 1, 0))
anticipatedGrade <- c("A", "C", "B", NA)
anticipatedHours <- c(15, 10, 15, NA)

# make Matrix
names <- c("courseNum", "courseName", "courseProf", "enrolled", "anticipatedGrade",
           "anticipatedHours")
types <- c(typeof(courseNum), typeof(courseName), typeof(courseProf), typeof(enrolled),
           typeof(anticipatedGrade), typeof(anticipatedHours))
class <- c(class(courseNum), class(courseName), class(courseProf), class(enrolled),
           class(anticipatedGrade), class(anticipatedHours))

myMatrix <- matrix(c(names, types, class), nrow=6, ncol=3, byrow=FALSE)
colnames(myMatrix) <- c("items", "type", "class")
View(myMatrix)
```

1.2) Create a data frame called bootcampDataFrame by combining all of the above vectors and create another table summarizing the type and class for the data frame. Do the data frame variables retain their original types/classes?

```
bootcampDataFrame <- data.frame(
  courseNum,
  courseName,
  courseProf,
  enrolled,
  anticipatedGrade,
  anticipatedHours
)

View(bootcampDataFrame)

types_2 <-  c()
class_2 <-  c()

for(i in 1:length(names)){
  types_2[i] <- typeof(bootcampDataFrame[,i])
  class_2[i] <- class(bootcampDataFrame[,i])
}
```

```
myMatrix_2 <- matrix(c(names, types_2, class_2), nrow=6, ncol=3, byrow=FALSE)
colnames(myMatrix_2) <- c("items", "type", "class")

# no, character in list turns into type integer and type factor in list
```

1.3) Combine the vectors from 1.1 into a list called bootcampDataList, where each vector is an element of the list. Assign the names of each element to be the names of the original vectors. Do the elements of the list maintain their original types/classes?

```
bootcampDataList <- list(
  courseNum,
  courseName,
  courseProf,
  enrolled,
  anticipatedGrade,
  anticipatedHours
)

names(bootcampDataList) <- names
types_3 <- c()
class_3 <- c()

for(i in 1:length(names)){
  types_3[i] <- typeof(bootcampDataList[[i]])
  class_3[i] <- class(bootcampDataList[[i]])
}

# yes, they do retain their types from the initial matrix, unlike when we did the dataframe
```

1.4) Write code that returns the following values in code chunks using echo = TRUE so that your code as well as your output is displayed after each calculation:

```
# number of hours anticpate spending on coursework, per week
sum(bootcampDataFrame$anticipatedHours, na.rm=TRUE) #40
```

```
## [1] 40
```

```
# number of hours anticpate spending on coursework, over all bootcamp
sum(bootcampDataFrame$anticipatedHours, na.rm=TRUE) * 5 #200
```

```
## [1] 200
```

```
# data frame with onlt the third row and first two columns of `bootcampDataFrame`
bootcampDataFrame[3,1:2]
```

```
##   courseNum                          courseName
## 3       504 Review of Probability and Statistics
```

```
# the first value in the second element of bootcampDataList
bootcampDataList[[2]][1]
```

```
## [1] "Exploratory Data Analysis"
```

```
View(bootcampDataList)

bootcampDataFrame$anticipatedGrade <- factor(bootcampDataFrame$anticipatedGrade,
                                             levels=c("C", "B", "A"), order=TRUE)
```

```
bootcampDataFrame
```

```
##   courseNum                          courseName     courseProf enrolled
## 1       593           Exploratory Data Analysis Paul Intrevado     TRUE
## 2       501           Computation for Analytics  Terrence Parr     TRUE
## 3       504 Review of Probability and Statistics   Jeff Hamrick     TRUE
## 4       502                      Linear Algebra    Xuemei Chen    FALSE
##   anticipatedGrade anticipatedHours
## 1                A               15
## 2                C               10
## 3                B               15
## 4             <NA>               NA
```

1.5) If you haven't already, convert the anticipatedGrade variable in bootcampDataFrame into an ordinal factor

```
maxGrade <- max(bootcampDataFrame$anticipatedGrade, na.rm=TRUE)

highestCourseName <- toString(bootcampDataFrame[bootcampDataFrame$anticipatedGrade == maxGrade,2])
highestCourseNum <- bootcampDataFrame[bootcampDataFrame$anticipatedGrade==maxGrade,1]

printf <- function(...) invisible(print(sprintf(...)))
printf("MSAN %d : %s", highestCourseNum, highestCourseName)
```

```
## [1] "MSAN 4 : Exploratory Data Analysis, NA"
## [2] "MSAN NA : Exploratory Data Analysis, NA"
```

Question 2

2.1) Read in the file titanic.csv and store the data in the data frame titanicData.

```
# assumes titanic datafile is in the same diretory
#setwd("/home/louiselai88gmail/Desktop/programming/USF/r")
titanicData <- read.csv("/home/louiselai88gmail/Desktop/programming/USF/r/titanic.csv", na='\\N')
```

2.2) How many rows are in this data frame?

```
nrow(titanicData)
```

```
## [1] 891
```

2.3) How many columns are in this data frame?

```
ncol(titanicData)
```

```
## [1] 12
```

2.4) Which variable has the most NA entries?

```
maxNA  = 0

for (i in c(1:ncol(titanicData))) {
  if(sum(is.na(titanicData[i])) > maxNA){
    maxNA = sum(is.na(titanicData[i]))
    print(colnames(titanicData[i]))
    print(sum(is.na(titanicData[i])))
  }
}
```

```
## [1] "Age"
## [1] 177
```

```
print(sum(is.na(titanicData$Age))) # there are 117 NAs for age
```

```
## [1] 177
```

2.5) Which variables, if any, should be converted to a different type than the default type they were imported as? Include of list of those you wish to change, what type they were previously, and what type you changed them to.

```
for (i in c(1:ncol(titanicData))) {
  print(colnames(titanicData[i]))
  print(class(titanicData[i][1,]))
}
```

```
## [1] "PassengerId"
## [1] "integer"
## [1] "Survived"
## [1] "integer"
## [1] "Pclass"
## [1] "integer"
## [1] "Name"
## [1] "factor"
## [1] "Sex"
## [1] "factor"
## [1] "Age"
## [1] "numeric"
## [1] "SibSp"
## [1] "integer"
## [1] "Parch"
## [1] "integer"
## [1] "Ticket"
## [1] "factor"
## [1] "Fare"
## [1] "numeric"
## [1] "Cabin"
## [1] "factor"
## [1] "Embarked"
## [1] "factor"
```

```
titanicTypes <- sapply(titanicData, typeof)

# survived & sex should be logical (binary), instead of integers, becuase there are only two options
# Pclass should be a factor with levels, as there is a natural hiearchy for the cabin classes
```

2.6) If you haven't already, coerce the survived variable into type logical.

```
titanicData$Survived <- as.logical(titanicData$Survived)

avgAgeSurvivor <- mean(titanicData$Age[titanicData$Survived==TRUE], na.rm=TRUE)
# mean age of survivors = 28.34

avgAgeNonsurvivor <- mean(titanicData$Age[titanicData$Survived==FALSE], na.rm=TRUE)
# mean age of survivors = 30.63

sum(titanicData$Survived==TRUE) # 342
```

```
## [1] 342
```

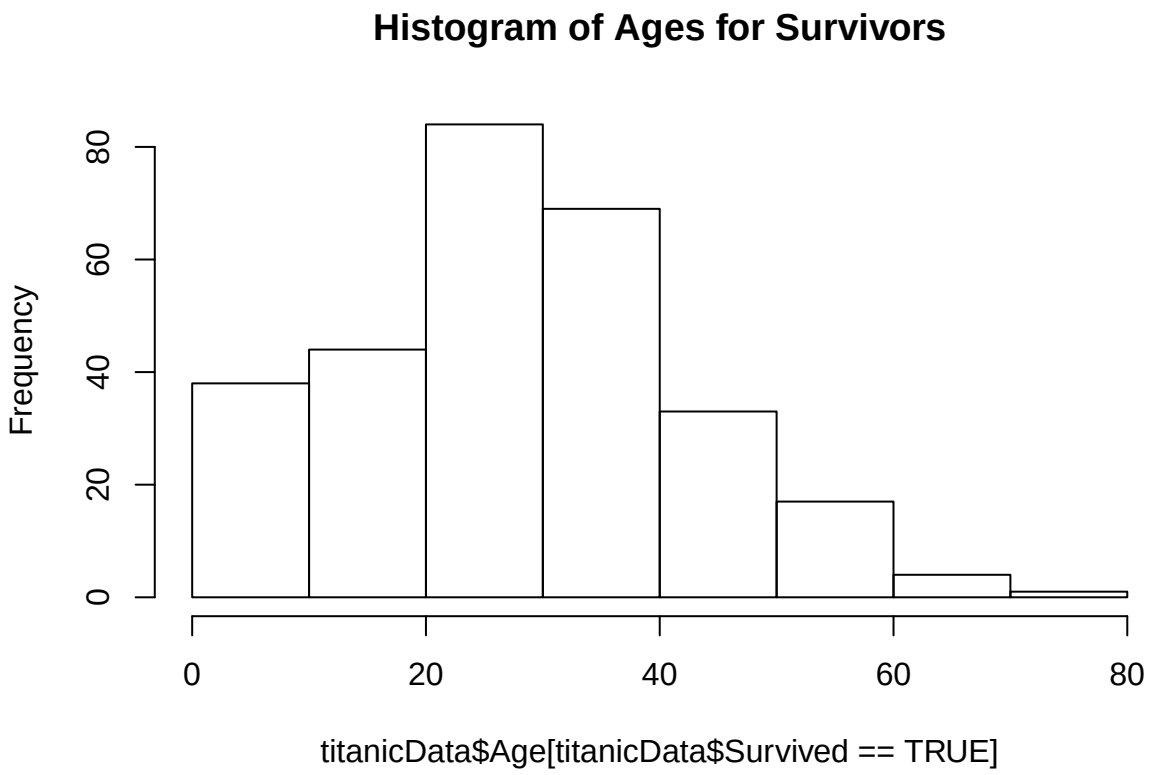
```
sum(titanicData$Survived==FALSE) # 549
```

```
## [1] 549
```

```
sum(is.na(titanicData$Survived)) # 0
```

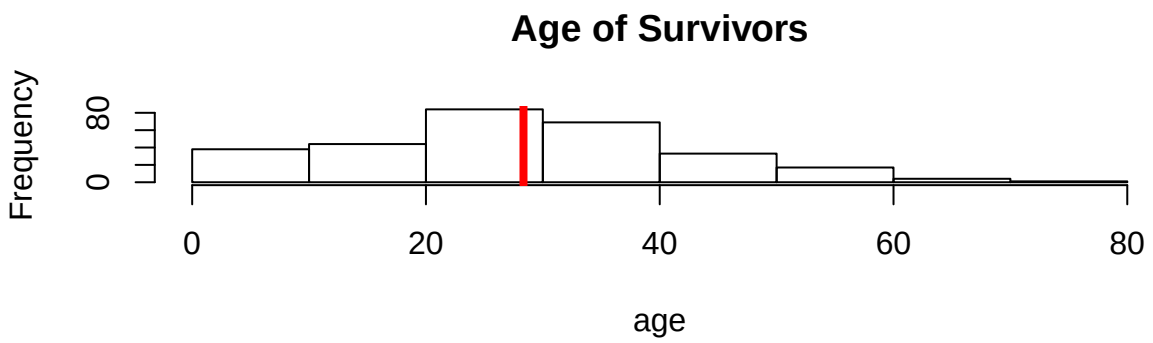
```
## [1] 0
```

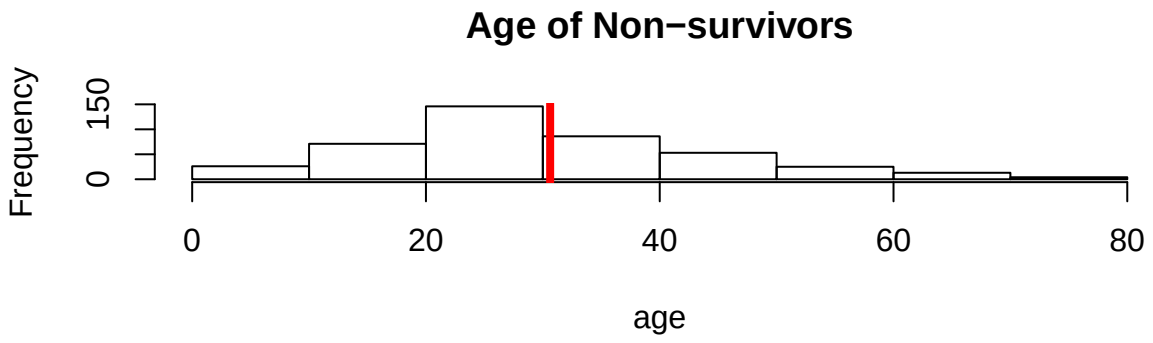
```
# plotting histogram
hist(titanicData$Age[titanicData$Survived == TRUE],
     main="Histogram of Ages for Survivors")
```




```
attach(titanicData)
par(mfrow=c(2,1)) # 2 rows 1 col
hist(titanicData$Age[titanicData$Survived == TRUE], main="Age of Survivors", xlab="age")
abline(v=avgAgeSurvivor, lwd=4, col="red")

hist(titanicData$Age[titanicData$Survived == FALSE], main ="Age of Non-survivors", xlab="age")
abline(v=avgAgeNonsurvivor, lwd=4, col="red")
```


Age of Survivors
Frequency
0
80
0
20
40
60
80
age


Age of Non−survivors
Frequency
0
150
0
20
40
60
80
age