Chapter 2

Shahaan Khan

Introduction

The following work is completed for my personal advance for the IST 687 - Introduction to Data Science course taught by Christopher Dunham in the pursuit of my Master of Science in Applied Human Centered AI at Syracuse University. The questions completed are outlined at the end of Chapter 2 in "Data Science for Business with R" by Jeffrey S. Saltz and Jeffrey M. Stanton.

CASE STUDY: CALCULATING NPS USING A DATAFRAME

Let's practice working with dataframes by setting up a small number of survey responses. Specifically, six surveys with likelihood to recommend (LTR) values of 9,9,7,6,8,7 and the type of travel also defined as follows: "Business travel", "Business travel", "Business travel", "Mileage tickets", "Personal Travel", "Personal Travel". Given this, is there a difference in net promoter score (NPS), comparing all the survey responses to just the business travel tickets? In order to do this analysis, we first need to create a dataframe that represents the six surveys. Then, we can calculate and compare the overall NPS, with the NPS value for business travel tickets. Here is the code:

```r
myFamilyNames <- c("Dad", "Mom", "Sis", "Bro", "Dog")
myFamilyNames
```

```
## [1] "Dad" "Mom" "Sis" "Bro" "Dog"
```

```r
myFamilyAges <- c(43, 42, 12, 8, 5)
myFamilyGenders <- c("Male", "Female", "Female", "Male", "Female")
myFamilyWeights <- c(188, 136, 83, 61, 44)

myFamily <- data.frame(myFamilyNames, myFamilyAges, myFamilyGenders, myFamilyWeights)

str(myFamily)
```

```
## 'data.frame':    5 obs. of  4 variables:
##  $ myFamilyNames  : chr  "Dad" "Mom" "Sis" "Bro" ...
##  $ myFamilyAges   : num  43 42 12 8 5
##  $ myFamilyGenders: chr  "Male" "Female" "Female" "Male" ...
##  $ myFamilyWeights: num  188 136 83 61 44
```

```r
summary(myFamily)
```

```
##  myFamilyNames       myFamilyAges myFamilyGenders    myFamilyWeights
##  Length:5           Min.   : 5    Length:5           Min.   : 44.0  
##  Class :character   1st Qu.: 8    Class :character   1st Qu.: 61.0  
##  Mode  :character   Median :12    Mode  :character   Median : 83.0  
##                     Mean   :22                       Mean   :102.4  
##                     3rd Qu.:42                       3rd Qu.:136.0  
##                     Max.   :43                       Max.   :188.0  
```

```
myFamilyAges <- c(myFamilyAges, 11)
# Case Study: Calculating NPS using a Dataframe
ltr = c(9, 9, 7, 6, 8, 7)
TypeOfTravel = c("Business travel", "Business travel", "Business travel", "Mileage",
    "Personal Travel", "Personal Travel")
survey = data.frame(ltr, TypeOfTravel)

# Output of new dataframe
str(survey)
```

```
## 'data.frame':    6 obs. of  2 variables:
##  $ ltr         : num  9 9 7 6 8 7
##  $ TypeOfTravel: chr  "Business travel" "Business travel" "Business travel" "Mileage" ...
```

```
# Calculate number of promoters and detractors
numP = sum(survey$ltr > 8)
numD = sum(survey$ltr < 7)

# Calculate NPS
total = nrow(survey)
nps = (numP/total - numD/total) * 100
nps
```

```
## [1] 16.66667
```

```
# Analysis for the business travel tickets
busTravelDF = survey[survey$TypeOfTravel == "Business
travel", ]
# Calculate number of promoters and demoters
numP = sum(busTravelDF$ltr > 8)
numD = sum(busTravelDF$ltr < 7)

# Calculate NPS
total = nrow(busTravelDF)
bus.nps = (numP/total - numD/total) * 100
bus.nps
```

```
## [1] NaN
```

```
str(survey$TypeOfTravel)
```

```
##  chr [1:6] "Business travel" "Business travel" "Business travel" "Mileage" ...
```

```
levels(survey$TypeOfTravel)
```

```
## NULL
```

```
nps
```

```
## [1] 16.66667
```

```
bus.nps
```

```
## [1] NaN
```

Chapter Challenges

1. Use the c() command to create a new variable containing the favorite food of each family member. For example, your list could contain the entry "Pizza." Make sure that your new variable includes exactly five values. Call the new variable myFoods. Use str() on your new variable to show what kind of variable it is.

```
myFoods = c("Hot Dog", "Pizza", "Ice Cream", "Snowcone", "Butter")
```

2. Add your new variable to the myFamily dataframe. If you were running the code while reading this chapter, you will have myFamilyNames, myFamilyAges, myFamilyGenders, and myFamilyWeights already available. Otherwise, you will need to type in the data for those variables as shown in this chapter.

```
myFamily$myFoods = myFoods
myFamily
```

```
##   myFamilyNames myFamilyAges myFamilyGenders myFamilyWeights   myFoods
## 1           Dad           43            Male             188   Hot Dog
## 2           Mom           42          Female             136     Pizza
## 3           Sis           12          Female              83 Ice Cream
## 4           Bro            8            Male              61  Snowcone
## 5           Dog            5          Female              44    Butter
```

3. Rerun the summary() function on myFamily to get descriptive information on all of the variables including your new variable. Take note of the data type for your new variable and report it in a comment.

```
summary(myFamily)
```

```
##  myFamilyNames       myFamilyAges myFamilyGenders    myFamilyWeights
##  Length:5           Min.   : 5    Length:5           Min.   : 44.0
##  Class :character   1st Qu.: 8    Class :character   1st Qu.: 61.0
##  Mode  :character   Median :12    Mode  :character   Median : 83.0
##                     Mean   :22                       Mean   :102.4
##                     3rd Qu.:42                       3rd Qu.:136.0
##                     Max.   :43                       Max.   :188.0
##    myFoods
##  Length:5
##  Class :character
##  Mode  :character
##
##
##
```

```
# The new data type is Character
```

4. Create an expression that shows a list of TRUE and FALSE values based on the age of each family member. The variable should be TRUE if myFamily$myFamilyAges is less than 40. In other words, your index will be TRUE for kids and FALSE for adults. Assign the results of your expression to a new variable called myIndex.

```
myIndex = myFamily$myFamilyAges < 40
myIndex
```

```
## [1] FALSE FALSE  TRUE  TRUE  TRUE
```

5. Use myIndex from the previous problem to show the favorite foods for each kid in the family. If you used the variable name suggested in problem 1, the expression would be myFamily$myFoods[myIndex].

```r
myFamily$myFoods[myIndex]
```

```
## [1] "Ice Cream" "Snowcone"  "Butter"
```

6. The ! character is used to invert a set of Boolean values by changing each TRUE to FALSE and vice versa. Adapt the expression from the previous problem to show the favorite foods for each kid in the family

```r
myFamily$myFoods[!myIndex]
```

```
## [1] "Hot Dog" "Pizza"
```