Chapter 1

Shahaan Khan

2025-09-01

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 1 in "Data Science for Business with R" by Jeffrey S. Saltz and Jeffrey M. Stanton.

Case Study: Calculating NPS

This was done to get familiar with the what NPS (Net Promoter Score) is. How it can work in R and how a vector can represent the likelihood to recommend.

```
ltr <- c(9,8,3,9,7,8,9,6,7,8,9)
range(ltr)
```

```
## [1] 3 9
```

```
promoters <- ltr[ltr>8]
numPromoters <- length(promoters)
detractorsTrueFalse <- ltr < 7
numDetractors <- sum(detractorsTrueFalse)
total <- length (ltr)
nps <- (numPromoters/total - numDetractors/total)*100
nps
```

```
## [1] 18.18182
```

```
range(ltr)
```

```
## [1] 3 9
```

Result

The case study shows an NPS score of 18.18% with a response range of 3-9.

Chapter End Questions

1. Use the c() function to add another family member's age onto the end of the myFamilyAges vector.

```
myFamilyAges <- c(43, 43, 12, 8, 5)
myFamilyAges <- append(myFamilyAges, 25)
myFamilyAges
```

```
## [1] 43 43 12  8  5 25
```

2. Use square brackets subsetting to show just the first element of the myFamilyAges vector.

```r
myFamilyAges[c(1)]
```

```
## [1] 43
```

3. Use square brackets subsetting together with the c() command to show just the odd numbered items from the myFamilyAges vector (i.e., just the first, third, and fifth items from this vector).

```r
myFamilyAges[c(1,3,5)]
```

```
## [1] 43 12  5
```

```r
# Found another way
myFamilyAges[seq(1, length(myFamilyAges), 2)]
```

```
## [1] 43 12  5
```

4. Create a conditional expression that outputs a set of TRUEs and FALSEs. The expression should show TRUE when an element of the myFamilyAges vector is equal to 12.

```r
myFamilyAges == 12
```

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

5. Using the code from the previous item, put an exclamation point in front of the conditional expression.

```r
myFamilyAges != 12
```

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

6. Use the conditional expression from the previous item within the square brackets to select those elements of the myFamilyAges vector that are not equal to 12.

```r
myFamilyAges[myFamilyAges!=12]
```

```
## [1] 43 43  8  5 25
```

7. Power User: Using the built-in Nile dataset, create a conditional expression that shows TRUE for every observation where the level of the Nile was over 900. Then use the sum() command to count up how many times the Nile dataset had observations higher than 900.

```r
sum(Nile > 900)
```

```
## [1] 49
```