

Worksheet-4a in R

Necel Kate Noblezada

2024-10-16

1.The table below shows the data about shoe size and height. Create a data frame.

```
sframe <- data.frame(
  Shoe_size = c(6.5, 9.0, 8.5, 8.5, 10.5, 7.0, 9.5, 9.0, 13.0, 7.5, 10.5, 8.5, 12.0, 10.5),
  Height = c(66.0, 68.0, 64.5, 65.0, 70.0, 64.0, 70.0, 71.0, 72.0, 64.0, 74.5, 67.0, 71.0),
  Gender = c("F", "F", "F", "F", "M", "F", "F", "F", "M", "F", "M", "F", "M", "M", "M", "M", "F", "F", "F")
)sframe
```

##	Shoe_size	Height	Gender
## 1	6.5	66.0	F
## 2	9.0	68.0	F
## 3	8.5	64.5	F
## 4	8.5	65.0	F
## 5	10.5	70.0	M
## 6	7.0	64.0	F
## 7	9.5	70.0	F
## 8	9.0	71.0	F
## 9	13.0	72.0	M
## 10	7.5	64.0	F
## 11	10.5	74.5	M
## 12	8.5	67.0	F
## 13	12.0	71.0	M
## 14	10.5	71.0	M
## 15	13.0	77.0	M
## 16	11.5	72.0	M
## 17	8.5	59.0	F
## 18	5.0	62.0	F
## 19	10.0	72.0	M
## 20	6.5	66.0	F
## 21	7.5	64.0	F
## 22	8.5	67.0	M
## 23	10.5	73.0	M
## 24	8.5	69.0	F
## 25	10.5	72.0	M
## 26	11.0	70.0	M
## 27	9.0	69.0	M

```
## 28      13.0    70.0      M
```

a.Describe the data.

The data contains two sets of observations for shoe size, height, and gender.

b.Create a subset by males and females with their corresponding shoe size and height.

What its result? Show the R scripts.

```
males <- sframe[sframe$Gender == "M", c("Shoe_size", "Height")]
females <- sframe[sframe$Gender == "F", c("Shoe_size", "Height")]
```

males

```
##      Shoe_size Height
## 5          10.5   70.0
## 9          13.0   72.0
## 11         10.5   74.5
## 13         12.0   71.0
## 14         10.5   71.0
## 15         13.0   77.0
## 16         11.5   72.0
## 19         10.0   72.0
## 22          8.5   67.0
## 23         10.5   73.0
## 25         10.5   72.0
## 26         11.0   70.0
## 27          9.0   69.0
## 28         13.0   70.0
```

females

```
##      Shoe_size Height
## 1           6.5   66.0
## 2           9.0   68.0
## 3           8.5   64.5
## 4           8.5   65.0
## 6           7.0   64.0
## 7           9.5   70.0
## 8           9.0   71.0
## 10          7.5   64.0
## 12          8.5   67.0
## 17          8.5   59.0
## 18          5.0   62.0
## 20          6.5   66.0
## 21          7.5   64.0
## 24          8.5   69.0
```

c. Find the mean of shoe size and height of the respondents. Write the R scripts and its

result.

```
mean_shoe_size <- mean(sframe$Shoe_size)
mean_height <- mean(sframe$Height)
```

```
mean_shoe_size
```

```
## [1] 9.410714
```

```
mean_height
```

```
## [1] 68.57143
```

d. Is there a relationship between shoe size and height? Why?

```
correlation <- cor(sframe$Shoe_size, sframe$Height)
correlation
```

```
## [1] 0.7766089
```

Yes, there is a positive relationship between shoe size and height, as individuals with larger shoe sizes tend to also have greater heights, which can be supported by the correlation analysis and visual representation of the data.

2. Construct character vector months to a factor with factor() and assign the result to factor_months_vector. Print out factor_months_vector and assert that R prints out the factor levels below the actual values. Consider data consisting of the names of months:

“March”, “April”, “January”, “November”, “January”, “September”, “October”, “September”, “November”, “August”, “January”, “November”, “July”, “December”, “August”, “August”, “September”, “November”, “February”, “April”)

```
months_vector <- c(
  "March", "April", "January", "November", "January", "September", "October",
  "September", "November", "August", "January", "November", "November", "February",
  "May", "August", "July", "December", "August", "August", "September", "November",
  "February", "April")
months_vector
```

```
## [1] "March"      "April"      "January"    "November"   "January"    "September"
## [7] "October"    "September"  "November"   "August"     "January"    "November"
```

```
## [13] "November" "February" "May"          "August"      "July"        "December"
## [19] "August"    "August"    "September" "November"    "February"    "April"

factor_months_vector <- factor(months_vector)

factor_months_vector

## [1] March    April    January  November January  September October
## [8] September November August   January  November November February
## [15] May      August   July     December August   August   September
## [22] November February April
## 11 Levels: April August December February January July March May ... September
```

3. Then check the `summary()` of the `months_vector` and `factor_months_vector`. | Interpret the results of both vectors. Are they both equally useful in this case?

```
summary(months_vector)

##      Length      Class      Mode
##          24 character character

summary(factor_months_vector)

##      April    August  December  February  January    July    March    May
##          2      4          1          2          3          1          1          1
## November  October September
##          5          1          3
```

4. Create a vector and factor for the table below.

Direction Frequency

East 1

West 4

North 3

Note: Apply the factor function with required order of the level.

```
new_order_data <- factor(factor_data, levels = c("East", "West", "North"))
print(new_order_data)
```

```
directions_vector <- c("East", "West", "North")
frequencies_vector <- c(1, 4, 3)

factor_data <- factor(directions_vector)

new_order_data <- factor(factor_data, levels = c("East", "West", "North"))
```

```
new_order_data
```

```
## [1] East West North  
## Levels: East West North
```

5. Enter the data below in Excel with file name = import_march.csv

a. Import the excel file into the Environment Pane using read.table() function. Write the code.

```
data <- read.table("import_march.csv", header = TRUE, sep = ",")
```

b. View the dataset. Write the R scripts and its result.

```
data
```

```
## Students Strategy.1 Strategy.2 Strategy.3  
## 1 Male 8 10 8  
## 2 4 8 6  
## 3 0 6 4  
## 4 Female 14 4 15  
## 5 10 2 12  
## 6 6 0 9
```

Using Conditional Statements (IF-ELSE)

6. Full Search # Exhaustive search is a methodology for finding an answer by exploring all possible cases. When trying to find a desired number in a set of given numbers, the method of finding the corresponding number by checking all elements in the set one by one can be called an exhaustive search. Implement an exhaustive search function that meets the input/output conditions below. # a. Create an R Program that allows the User to randomly select numbers from 1 to 50. Then display the chosen number. If the number is beyond the range of the selected choice, it will have to display a string "The number selected is beyond the range of 1 to 50". If number 20 is inputted by the User, it will have to display "TRUE", otherwise display the input number.

```
exhaustive_search <- function() {  
  chosen_number <- as.integer(readline(prompt = "Please enter a number between 1 and 50: "))  
  
  if (is.na(chosen_number)) {  
    cat("Invalid input. Please enter a valid integer number between 1 and 50.\n")  
    return()  
  }  
  
  cat("You have chosen the number:", chosen_number, "\n")  
  
  if (chosen_number < 1 || chosen_number > 50) {  
    cat("The number selected is beyond the range of 1 to 50.\n")  
  } else if (chosen_number == 20) {  
    cat("TRUE\n")  
  } else {  
    cat("You selected the number:", chosen_number, "\n")  
  }  
}
```

```

    }
}

exhaustive_search()

## Please enter a number between 1 and 50:
## Invalid input. Please enter a valid integer number between 1 and 50.
## NULL

```

7. Change

At ISATU University's traditional cafeteria, snacks can only be purchased with bills. A long-standing rule at the concession stand is that snacks must be purchased with as few coins as possible. There are three types of bills: 50 pesos, 100 pesos, 200 pesos, 500 pesos, 1000 pesos.

a. Write a function that prints the minimum number of bills that must be paid, given the price of the snack. Input: Price of snack (a random number divisible by 50) Output: Minimum number of bills needed to purchase a snack.

```

min_bills <- function(price) {
  bills <- c(1000, 500, 200, 100, 50)
  count <- 0

  for (bill in bills) {
    while (price >= bill) {
      price <- price - bill
      count <- count + 1
    }
  }

  cat("Minimum number of bills needed:", count, "\n")
}

min_bills(850)

## Minimum number of bills needed: 4

```

8. The following is each student's math score for one semester. Based on this, answer the following questions.

Name	Grade1	Grade2	Grade3	Grade4
------	--------	--------	--------	--------

Annie	85	65	85	100
-------	----	----	----	-----

Thea	65	75	90	90
------	----	----	----	----

Steve	75	55	80	85
-------	----	----	----	----

Hanna	95	75	100	90
-------	----	----	-----	----

a. Create a dataframe from the above table. Write the R codes and its output.

```
scores <- data.frame(  
  Name = c("Annie", "Thea", "Steve", "Hanna"),  
  Grade1 = c(85, 65, 75, 95),  
  Grade2 = c(65, 75, 55, 75),  
  Grade3 = c(85, 90, 80, 100),  
  Grade4 = c(100, 90, 85, 90)  
)
```

scores

```
##      Name Grade1 Grade2 Grade3 Grade4  
## 1 Annie      85      65      85      100  
## 2 Thea       65      75      90      90  
## 3 Steve      75      55      80      85  
## 4 Hanna      95      75     100      90
```

b. Without using the rowMean function, output the average score of students whose average math score over 90 points during the semester. write R code and its output. # Example Output: Annie's average grade this semester is 88.75.

```
scores <- data.frame(  
  Name = c("Annie", "Thea", "Steve", "Hanna"),  
  Grade1 = c(85, 65, 75, 95),  
  Grade2 = c(65, 75, 55, 75),  
  Grade3 = c(85, 90, 80, 100),  
  Grade4 = c(100, 90, 85, 90)  
)  
  
for (i in 1:nrow(scores)) {  
  average_score <- (scores$Grade1[i] + scores$Grade2[i] + scores$Grade3[i] + scores$Grade4[i]) / 4  
  if (average_score > 90) {  
    cat(scores$Name[i], "'s average grade this semester is", round(average_score, 2), "\n")  
  }  
}
```

c. Without using the mean function, output as follows for the tests in which the average score was less than 80 out of 4 tests.

Example output: The nth test was difficult.

```
scores <- data.frame(
  Name = c("Annie", "Thea", "Steve", "Hanna"),
  Grade1 = c(85, 65, 75, 95),
  Grade2 = c(65, 75, 55, 75),
  Grade3 = c(85, 90, 80, 100),
  Grade4 = c(100, 90, 85, 90)
)

for (test in 1:4) {
  total_score <- 0
  for (i in 1:nrow(scores)) {
    total_score <- total_score + scores[i, test + 1] # test + 1 because the first column is 'Name'
  }
  average_score <- total_score / nrow(scores)
  if (average_score < 80) {
    cat("The", test, "test was difficult.\n")
  }
}
```

The 2 test was difficult.

d. Without using the max function, output as follows for students whose highest score for a semester exceeds 90 points.

Example Output: Annie's highest grade this semester is 95.

```
scores <- data.frame(
  Name = c("Annie", "Thea", "Steve", "Hanna"),
  Grade1 = c(85, 65, 75, 95),
  Grade2 = c(65, 75, 55, 75),
  Grade3 = c(85, 90, 80, 100),
  Grade4 = c(100, 90, 85, 90)
)

for (i in 1:nrow(scores)) {
  highest_score <- scores[i, 2] # Start with the first grade
  for (j in 3:5) {
    if (scores[i, j] > highest_score) {
      highest_score <- scores[i, j]
    }
  }
  if (highest_score > 90) {
    cat(scores$Name[i], "'s highest grade this semester is", highest_score, "\n")
  }
}
```

Annie 's highest grade this semester is 100


```
## Hanna 's highest grade this semester is 100
```