

Factors, Data Frames, Data Tables and Tibbles

1. Factors

Example

Question: Lets create a vector v

#

```
v <- c(1,3,5,8,2,1,3,5,3,5)
```

Then determine whether this vector is a factor

#

```
is.factor(v)
```

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

Challenge

Question: Calculate the categorical distribution as shown and figure out why the given output

#

```
factor(v)
```

```
## [1] 1 3 5 8 2 1 3 5 3 5
```

```
## Levels: 1 2 3 5 8
```

Example

Question: Assign factor v to x and print out x

#

```
x <- factor(v)
```

x

```
## [1] 1 3 5 8 2 1 3 5 3 5
```

```
## Levels: 1 2 3 5 8
```

Challenge

Question: Determine whether x is a factor below.

Hint: Just like the way you did when you were finding out whether vector v is a factor

```
is.factor(x)
```

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

```

#

## Example
# ---
# Question: First we create a vector as input, check whether its a factor,
# apply the factor function to create a factor from the vector
# ---
#
data <-
c("East", "West", "East", "North", "North", "East", "West", "West", "West", "East", "North")

# Then print out this vector
data

## [1] "East" "West" "East" "North" "North" "East" "West" "West" "West"
## [10] "East" "North"

# Now, check whether this is a factor
is.factor(data)

## [1] FALSE

# Then, apply the factor function to create a factor from the vector
factor_data <- factor(data)

# Then see our newly created factor

factor_data

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

# Check whether this is a factor
is.factor(factor_data)

## [1] TRUE

# Example
# ---
# Creating a factor, determine and check the levels
# ---
# OUR CODE GOES BELOW
#
sex <- factor(c("male", "female", "female", "male"))

# Determining the levels
levels(sex)

## [1] "female" "male"

```

```

# Then checking the number of levels using nlevels()
nlevels(sex)

## [1] 2

# Sometimes, the order of the factors does not matter, other times you might
# want to specify the order
# because it is meaningful (e.g., "low", "medium", "high") or it
# is required by particular type of analysis.
# Additionally, specifying the order of the levels allows us to compare
# levels:

food <- factor(c("low", "high", "medium", "high", "low", "medium", "high"))

# then print out levels of food
levels(food)

## [1] "high" "low" "medium"

```

2. Data Frames

Creating a Dataframe

2.1 Creating a Dataframe

Example

```

# ---
# Question: Lets create a data frame BMI
# ---
#
BMI <- data.frame(
  gender = c("Male", "Male", "Female"),
  height = c(152, 171.5, 165),
  weight = c(81, 93, 78),
  Age    = c(42, 38, 26)
)

```

Then print it out below

BMI

```

##   gender height weight Age
## 1  Male  152.0     81  42
## 2  Male  171.5     93  38
## 3 Female  165.0     78  26

```

Challenge

```

# ---
# Question: Create a data frame family with column names Name, Age, Gender
# and Occupation.
# Populate it with 5 your own family members.
# ---
#

```

```
family <- data.frame(
  Name = c("Vincent", "Levi", "Susan", "Lynn", "John"),
  Age = c(20, 45, 13, 6, 70),
  Gender = c("Male", "Male", "Female", "Female", "Male"),
  Occupation = c("Nurse", "CO", "IT", "AgriBiz", "Doctor")
)
family
```

	Name	Age	Gender	Occupation
## 1	Vincent	20	Male	Nurse
## 2	Levi	45	Male	CO
## 3	Susan	13	Female	IT
## 4	Lynn	6	Female	AgriBiz
## 5	John	70	Male	Doctor

Selecting Elements From a DataFrame

2.2 Selecting Elements From a DataFrame

Example

Question: Selecting elements from the BMI dataframe

#

selecting row 1

BMI[1,]

```
##   gender height weight Age
## 1   Male    152     81  42
```

selecting rows 1 to 2

BMI[1:2,]

```
##   gender height weight Age
## 1   Male  152.0     81  42
## 2   Male  171.5     93  38
```

selecting column 1

BMI[,1]

```
## [1] "Male" "Male" "Female"
```

selecting column 1 to 2

BMI[,1:2]

```
##   gender height
## 1   Male  152.0
## 2   Male  171.5
## 3 Female  165.0
```

selecting row 1 in column 2

BMI[1,2]

```
## [1] 152

## Challenge
# ---
# Question: Select the column 2 from the BMI dataframe
# ---
BMI[,2]

## [1] 152.0 171.5 165.0

#

## Challenge
# ---
# Question: Select the second and third members of your family
family[2:3,]

##      Name Age Gender Occupation
## 2   Levi  45   Male           CO
## 3  Susan  13 Female           IT
```

Sorting

2.3 Sorting

```
# Question: Sort the BMI dataframe by using the order() function
# ---
#

# Sort in ascending order by gender
# ---
#
sorted_by_gender <- BMI[order(BMI$gender),]

sorted_by_gender

##   gender height weight Age
## 3 Female  165.0     78  26
## 1  Male   152.0     81  42
## 2  Male   171.5     93  38

# Sort in descending order by weight
# ---
#
sorted_by_weight <- BMI[order(-BMI$weight),]

# Print out sorted_by_weight below
# ---
#
sorted_by_weight
```

```
##   gender height weight Age
## 2   Male  171.5     93  38
## 1   Male  152.0     81  42
## 3 Female  165.0     78  26

# And sort in descending order by gender below
gender_desc <- BMI[order(BMI$gender, decreasing=TRUE),]

gender_desc

##   gender height weight Age
## 1   Male  152.0     81  42
## 2   Male  171.5     93  38
## 3 Female  165.0     78  26
```

3. Data Tables

Creating a Data Table

```
## Example
# ---
# Question: Create a data table DT
# ---
library(data.table)
#
DT = data.table(
  ID = c("b", "b", "b", "a", "a", "c"),
  a = 1:6,
  b = 7:12,
  c = 13:18
)
DT

##      ID a  b  c
## 1:   b 1  7 13
## 2:   b 2  8 14
## 3:   b 3  9 15
## 4:   a 4 10 16
## 5:   a 5 11 17
## 6:   c 6 12 18
```

Selecting Elements From a Data Table

3.3 Selecting Elements From a Data Table

```
# Question: Select elements from the given datatable DT
# ---

# Selecting Row 1
DT[1,]

##      ID a  b  c
## 1:   b 1  7 13
```

```

#

# Selecting Rows 1 to 2
DT[1:2,]

##      ID a b  c
## 1:   b 1 7 13
## 2:   b 2 8 14

# Find out what happens when we print out the following statement
DT[,1]

##      ID
## 1:   b
## 2:   b
## 3:   b
## 4:   a
## 5:   a
## 6:   c

# print out the first column

# Find out what happens when we print out the following statement
DT[,1:2]

##      ID a
## 1:   b 1
## 2:   b 2
## 3:   b 3
## 4:   a 4
## 5:   a 5
## 6:   c 6

#print out the first and 2nd column

# And Lastly find out what happens when we print out the following statement
DT[1,2]

##      a
## 1: 1

#print out value in the first column

# Select the fourth and third rows from the data table
# ---
DT[3:4,]

##      ID a  b  c
## 1:   b 3  9 15
## 2:   a 4 10 16

```

Sorting a Data Table

3.4 Sorting a Data Table

```
## Example
# ---
# Question: Sorting the datatable in ascending order by c
# ---
# OUR CODE GOES BELOW
#

# Performing the sort
#
sorted_by_c <- DT[order(DT$c),]

# Printing out sorted_by_c
# ---
sorted_by_c

##      ID a  b  c
## 1:   b 1  7 13
## 2:   b 2  8 14
## 3:   b 3  9 15
## 4:   a 4 10 16
## 5:   a 5 11 17
## 6:   c 6 12 18

# Sort in descending order by b,
# ---
#
sorted_by_b <- DT[order(-DT$b),]

# Finally printing out sorted_by_b
# ---
sorted_by_b

##      ID a  b  c
## 1:   c 6 12 18
## 2:   a 5 11 17
## 3:   a 4 10 16
## 4:   b 3  9 15
## 5:   b 2  8 14
## 6:   b 1  7 13

#
```

4. Tibbles

4.1 Creating a Tibble

```
## Example
# ---
```



```

# Question: Create a tibble tb

# First, we load the tibble package
library(tibble)
# Then create our tibble tb
tb <- tibble(
  x = 1:5,
  y = 1,
  z = x ^ 2 + y
)

# And finally print the created tibble
# ---
tb

## # A tibble: 5 x 3
##       x     y     z
##   <int> <dbl> <dbl>
## 1     1     1     2
## 2     2     1     5
## 3     3     1    10
## 4     4     1    17
## 5     5     1    26

```

Selecting a Tibble Code Example

4.1 Selecting a Tibble

```

## Example
# ---
# Question: Find out what happens when we print the following
# ---
# First row
tb[1,]

## # A tibble: 1 x 3
##       x     y     z
##   <int> <dbl> <dbl>
## 1     1     1     2

# print 1st row and 2nd row
tb[1:2, ]

## # A tibble: 2 x 3
##       x     y     z
##   <int> <dbl> <dbl>
## 1     1     1     2
## 2     2     1     5

# print the 1st column
tb[,1]

```

```
## # A tibble: 5 x 1
##       x
##   <int>
## 1     1
## 2     2
## 3     3
## 4     4
## 5     5

# print out the 1st and 2nd columns
tb[,1:2 ]

## # A tibble: 5 x 2
##       x     y
##   <int> <dbl>
## 1     1     1
## 2     2     1
## 3     3     1
## 4     4     1
## 5     5     1

# Select the second and third rows
#
tb[2:3,]

## # A tibble: 2 x 3
##       x     y     z
##   <int> <dbl> <dbl>
## 1     2     1     5
## 2     3     1    10
```

Sorting a Tibble

4.1 Sorting a Tibble

```
## Example
# ---
# Question: Find out what happens when we sort by doing the following
# ---
# Sort in ascending order by z,
sorted_by_1 <- tb[order(tb$z),]
sorted_by_1

## # A tibble: 5 x 3
##       x     y     z
##   <int> <dbl> <dbl>
## 1     1     1     2
## 2     2     1     5
## 3     3     1    10
## 4     4     1    17
## 5     5     1    26
```

```
# Sort in descending order by x,  
sorted_by_2 <- tb[order(-tb$x),]  
sorted_by_2
```

```
## # A tibble: 5 x 3  
##       x     y     z  
##   <int> <dbl> <dbl>  
## 1     5     1    26  
## 2     4     1    17  
## 3     3     1    10  
## 4     2     1     5  
## 5     1     1     2
```

```
# Sort tb in ascending order by x below  
# ---  
sorted_by_x_asc <- tb[order(tb$x),]  
sorted_by_x_asc
```

```
## # A tibble: 5 x 3  
##       x     y     z  
##   <int> <dbl> <dbl>  
## 1     1     1     2  
## 2     2     1     5  
## 3     3     1    10  
## 4     4     1    17  
## 5     5     1    26
```