R review: data structures

Marcus Birkenkrahe (pledged)

January 1, 2023

1 README

This file covers main data structures in R:

- Vectors, or ordered n-tuples
- Factor vectors, or vectors for ordered or nominal categories
- Lists, or collections of any other R data structure
- Data frames, or a table of features (cols) and observations (rows)

Most of this material can be found in Lantz, Machine Learning with R (3e, 2019), Packt. Solutions can be found in GitHub.

2 DONE Identify yourself

1. In Emacs, replace the placeholder `[yourname]` at the top of this file by your own name and write `(pledged)` next to it
2. Go with the cursor on the headline and hange the `TODO` label to `DONE` by entering `S-<right>` ("Shift + right-arrow").

3 DONE Vectors

1. Construct a set of vectors containing data on three medical patients:
   - Create a `character` vector named `subject_name` to store the three patient names: John Doe, Jane Doe, and Steve Graves
   - Create a `numeric` vector named `temperature` to store each patient's body temperature in degrees Fahrenheit: 98.1, 98.6, 101.4

- Create a `logical` vector named `flu_status` to store each patient's diagnosis: `TRUE` if he or she has influenza, `FALSE` otherwise: John and Jane do not have the flu, but Steve does have the flu.

```
subject_name <- c("John Doe", "Jane Doe", "Steve Graves")
temperature <- c(98.1, 98.6, 101.4)
flu_status <- c(FALSE, FALSE, TRUE)
```

2. Display the content of the vectors.

```
subject_name
(subject_name) # () is equivalent to the identity function
print(subject_name)
head(subject_name)
show(subject_name)

temperature
flu_status
```

```
[1] "John Doe"     "Jane Doe"     "Steve Graves"
[1] "John Doe"     "Jane Doe"     "Steve Graves"
[1] "John Doe"     "Jane Doe"     "Steve Graves"
[1] "John Doe"     "Jane Doe"     "Steve Graves"
[1] "John Doe"     "Jane Doe"     "Steve Graves"
[1]  98.1  98.6 101.4
[1] FALSE FALSE  TRUE
```

3. Display the type of vector for each of the three vectors.

```
class(subject_name)
typeof(subject_name)
mode(subject_name)
str(subject_name)

class(temperature)
class(flu_status)
```

```
[1] "character"
[1] "character"
```

```
[1] "character"
 chr [1:3] "John Doe" "Jane Doe" "Steve Graves"
[1] "numeric"
[1] "logical"
```

4. Extract the `temperature` (`98.6`) of the 2nd patient, John Doe, using the index operator.

   ```
   temperature[2]
   temperature[which(temperature==98.6)]
   temperature[c(FALSE,TRUE,FALSE)]
   ```

   ```
   [1] 98.6
   [1] 98.6
   [1] 98.6
   ```

5. Extract the `temperature` values of the other two patients by excluding John Doe's temperature from the printout.

   ```
   temperature[-2]
   temperature[c(TRUE,FALSE,TRUE)]
   temperature[-c(FALSE,TRUE,FALSE)]
   temperature[c(TRUE,FALSE,TRUE)]
   ```

   ```
   [1]  98.1 101.4
   [1]  98.1 101.4
   [1]  98.6 101.4
   [1]  98.1 101.4
   ```

4 DONE Factors

1. Create a `factor` vector named `gender` for the three patients, with the values `"MALE"` or `"FEMALE"`.

   ```
   gender <- factor(c("MALE","FEMALE","MALE"))
   gender
   ```

   ```
   [1] MALE   FEMALE MALE
   Levels: FEMALE MALE
   ```

2. Create a `factor` for blood type called `blood`. John, Jane and Steve have blood type `"O"`, `"AB"` and `"A"`, respectively. Since there are four blood types, add another `level`, `"B"` inside the definition of `blood`.

   ```
   blood <- factor(c("O","AB","A"),
   levels = c("A","B","AB","O"))
   blood
   ```

   ```
   [1] O  AB A
   Levels: A B AB O
   ```

3. Create an *ordered* `factor` for severity of patient symptoms called `symptoms`, with the values `"SEVERE"`, `"MILD"` and `"MODERATE"`.

   ```
   symptoms <- factor(c("SEVERE","MILD","MODERATE"),
      levels=c("MILD","MODERATE","SEVERE"),
      ordered=TRUE)
   symptoms
   ```

   ```
   [1] SEVERE   MILD     MODERATE
   Levels: MILD < MODERATE < SEVERE
   ```

4. Test whether each patient's symptoms are more severe than moderate.

   ```
   symptoms > "MODERATE"
   ```

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

5 DONE Lists

1. Create a `list` named `subject_1` with *named* components for all of the first patient's data: `name`, `temperature`, `flu_status`, `gender`, `blood`, and `symptoms`. Print the list.

   ```
   subject_1 <- list(name=subject_name[1],
     temperature=temperature[1],
     flu_status=flu_status[1],
     gender=gender[1],
     blood=blood[1],
     symptoms=symptoms[1])
   subject_1
   ```

```
$name
[1] "John Doe"

$temperature
[1] 98.1

$flu_status
[1] FALSE

$gender
[1] MALE
Levels: FEMALE MALE

$blood
[1] O
Levels: A B AB O

$symptoms
[1] SEVERE
Levels: MILD < MODERATE < SEVERE
```

2. Extract the `temperature` of the patient from the list `subject_1`.

```
subject_1[["temperature"]]
subject_1[[2]]
subject_1$temperature
subject_1[2]
```

```
[1] 98.1
[1] 98.1
[1] 98.1
$temperature
[1] 98.1
```

3. Extract the temperature and the flu status of the patient from the list `subject_1` with one command.

```
subject_1[c("temperature","flu_status")]
```

```
$temperature
[1] 98.1

$flu_status
[1] FALSE
```

6 DONE Data frames

1. Combine the features `subject_name`, `temperature`, `flu_status`, `gender`, `blood` and `symptoms` into a data frame.

   ```
   pt_data <- data.frame(subject_name,
         temperature,
         flu_status,
         gender,
         blood,
         symptoms)
   pt_data
   pt_data[,]
   ```

   ```
     subject_name temperature flu_status gender blood symptoms
   1     John Doe        98.1      FALSE   MALE     O   SEVERE
   2     Jane Doe        98.6      FALSE FEMALE    AB     MILD
   3 Steve Graves       101.4       TRUE   MALE     A MODERATE
     subject_name temperature flu_status gender blood symptoms
   1     John Doe        98.1      FALSE   MALE     O   SEVERE
   2     Jane Doe        98.6      FALSE FEMALE    AB     MILD
   3 Steve Graves       101.4       TRUE   MALE     A MODERATE
   ```

2. Extract the `subject_name` vector from `pt_data`, with the names of the three patients.

   ```
   pt_data$subject_name
   ```

   ```
   [1] "John Doe"     "Jane Doe"     "Steve Graves"
   ```

3. Extract `temperature` and `flu_status` of all patients from `pt_data` with one command.

```
pt_data[c("temperature","flu_status")]
pt_data[2:3]
```

```
  temperature flu_status
1        98.1      FALSE
2        98.6      FALSE
3       101.4       TRUE
  temperature flu_status
1        98.1      FALSE
2        98.6      FALSE
3       101.4       TRUE
```

4. Extract the `temperature` of `John Doe` from the data frame. John's data are in row 1 and column 2 of `pt_data`.

   ```
   pt_data[1,2]
   ```

   ```
   [1] 98.1
   ```

5. What if you don't know the row and column number but only that `John Doe` is a name in the feature vector `subject_name`, and that his temperature is in the feature vector `temperature`?

   ```
   pt_data[subject_name=="John Doe","temperature"]
   ```

   ```
   [1] 98.1
   ```

6. Extract the data from the first and third row, and the second and fourth column of the data frame `pt_data`.

   ```
   pt_data[c(1,3),c(2,4)]
   ```

   ```
     temperature gender
   1        98.1   MALE
   3       101.4   MALE
   ```

7. Copy all columns of `pt_data` to another data frame `df` except the `subject_name` column and print `df`.

```
df <- pt_data[,-1]
df
```

```
  temperature flu_status gender blood symptoms
1        98.1      FALSE   MALE     O   SEVERE
2        98.6      FALSE FEMALE    AB     MILD
3       101.4       TRUE   MALE     A MODERATE
```

8. Name the patient records according to the patient's names, John Doe, Jane Doe and Steve Graves, then print `df`.

```
rownames(df) <- c("John Doe", "Jane Doe", "Steve Graves")
df
```

```
             temperature flu_status gender blood symptoms
John Doe            98.1      FALSE   MALE     O   SEVERE
Jane Doe            98.6      FALSE FEMALE    AB     MILD
Steve Graves       101.4       TRUE   MALE     A MODERATE
```

9. Extract the `temperature` of `John Doe` from the data frame `df` using the row and column names.

```
df["John Doe","temperature"]
```

```
[1] 98.1
```

10. Extract `gender` and `blood` type of John Doe and Steve Graves from the data frame `df`.

```
df[c("John Doe","Steve Graves"),c("gender","blood")]
```

```
             gender blood
John Doe       MALE     O
Steve Graves   MALE     A
```

11. Extract `gender` and `blood` type of John Doe and Steve Graves from the data frame `pt_data` by **removing** all data that you do not want.

```
pt_data[-2,-c(2,3,6)]
```

```
  subject_name gender blood
1     John Doe   MALE     O
3 Steve Graves   MALE     A
```

12. Add a new column `temp_c` to `pt_data` that contains the `temperature` in degrees Celsius: the conversion formula is: 1 C = (1 F - 32) * (5/9).

```
pt_data$temp_c <- (pt_data$temperature - 32) * (5/9)
```

13. Print the temperature of all patients in Fahrenheit and Celsius.

```
pt_data[c("temperature","temp_c")]
```

```
  temperature  temp_c
1        98.1 36.7222
2        98.6 37.0000
3       101.4 38.5556
```

14. Change the display of digits so that both temperature columns show only one digit after the decimal point.

```
options(digits=3)
pt_data[c("temperature","temp_c")]
options(digits=6)
format(pt_data[c("temperature","temp_c")],digits=3)
```

```
  temperature temp_c
1        98.1   36.7
2        98.6   37.0
3       101.4   38.6
  temperature temp_c
1        98.1   36.7
2        98.6   37.0
3       101.4   38.6
```