Video 5: Factors, Matrices, DataFrames

Stats 102A

Miles Chen, PhD

Factors

A **factor** is a vector used to represent categorical values. It is internally stored as an integer vector with levels and class attributes.

```
gender <- c("M", "F", "F", "X", "M", "F")
gender_fac <- factor(gender)
gender_fac
```

```
[1] M F F X M F
Levels: F M X
```

```
levels(gender_fac)
```

```
[1] "F" "M" "X"
```

```
typeof(gender_fac)
```

```
[1] "integer"
```

Factors

Internally, the factor is an integer vector. When displayed, it replaces the integer with the corresponding level.

```
1 gender_fac
```

```
[1] M F F X M F
Levels: F M X
```

```
1 as.integer(gender_fac)
```

```
[1] 2 1 1 3 2 1
```

```
1 attributes(gender_fac)
```

```
$levels
[1] "F" "M" "X"

$class
[1] "factor"
```

Factors

Watch out! If a vector of numbers get turned into factors, the unique values get stored as levels. This can lead to unexpected results if you aren't careful.

```
x <- c(0, 1, 10, 5)
x_fac <- factor(x)
x_fac
```

```
[1] 0  1  10 5
Levels: 0 1 5 10
```

```
mean(x_fac) # we try to take the mean but it doesn't work
```

```
[1] NA
```

Factors

```
1 # so we coerce to numeric, but the result doesn't make sense
2 mean(as.numeric(x_fac)) # the mean of 0, 1, 10, 5 should be 4
```

```
[1] 2.5
```

```
1 as.numeric(x_fac) # internally, they are stored as integers
```

```
[1] 1 2 4 3
```

```
1 x_fac  # again, x_fac is a factor
```

```
[1] 0  1  10 5
Levels: 0 1 5 10
```

```
1 mean(as.numeric(as.character(x_fac))) # this works
```

```
[1] 4
```

Factors - other rules

You can’t use values that are not in the levels

```
gender_fac[2] <- "male"
gender_fac
```

```
[1] M    <NA> F    X    M    F
Levels: F M X
```

Matrices

A **matrix** in R is an atomic vector with a dimension attribute: a vector for the number of rows and columns.

```
M <- 1:10
M # M is an atomic vector of integers
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
class(M)
```

```
[1] "integer"
```

```
attr(M, "dim") <- c(2, 5) # I set dimension attributes
M # M is now a matrix of integers
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10
```

Matrices

```
attributes(M)  # there's only one attribute: dim
```

```
$dim
[1] 2 5
```

```
class(M) # class is smart enough to figure out that it's a matrix
```

```
[1] "matrix" "array"
```

```
attr(M, "dim") <- NULL # remove the dimension attribute
M # M is back to a vector
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
class(M)
```

```
[1] "integer"
```

Arrays

An **array** in R is an atomice vector where the dimension attribute is a vector longer than 2.

```r
A <- 1:12
attr(A, "dim") <- c(2, 3, 2)
A
```

```
, , 1

     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

, , 2

     [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12
```

Arrays can also be created using `array()`.

Data Frames

A **data frame** in R is internally stored as a list of equal length vectors with a class attribute called `data.frame`.

```
1 trees # data frame
```

```
   Girth Height Volume
1    8.3     70   10.3
2    8.6     65   10.3
3    8.8     63   10.2
4   10.5     72   16.4
5   10.7     81   18.8
6   10.8     83   19.7
7   11.0     66   15.6
8   11.0     75   18.2
9   11.1     80   22.6
10  11.2     75   19.9
11  11.3     79   24.2
12  11.4     76   21.0
13  11.4     76   21.4
14  11.7     69   21.3
```

Data Frames

```
1 class(trees)
```

```
[1] "data.frame"
```

```
1 typeof(trees)
```

```
[1] "list"
```

```
1 str(trees)
```

```
'data.frame':   31 obs. of  3 variables:
 $ Girth : num  8.3 8.6 8.8 10.5 10.7 10.8 11 11 11.1 11.2 ...
 $ Height: num  70 65 63 72 81 83 66 75 80 75 ...
 $ Volume: num  10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6 19.9 ...
```

```
1 as.list(trees)
```

```
$Girth
 [1]  8.3  8.6  8.8 10.5 10.7 10.8 11.0 11.0 11.1 11.2 11.3 11.4 11.4 11.7
12.0
[16] 12.9 12.9 13.3 13.7 13.8 14.0 14.2 14.5 16.0 16.3 17.3 17.5 17.9 18.0
18.0
[31] 20.6

$Height
 [1] 70 65 63 72 81 83 66 75 80 75 79 76 76 69 75 74 85 86 71 64 78 80 74 72
77
[26] 81 82 80 80 80 87

$Volume
 [1] 10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6 19.9 24.2 21.0 21.4 21.3
19.1
```

```r
attributes(trees)
```

```
$names
[1] "Girth"  "Height" "Volume"

$class
[1] "data.frame"

$row.names
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
25
[26] 26 27 28 29 30 31
```