# Dealing with Factors

Factors are variables in R which take on a limited number of different values; such variables are often referred to as [categorical variables](https://en.wikipedia.org/wiki/Categorical_variable). One of the most important uses of factors is in statistical modeling; since categorical variables enter into statistical models such as lm and glm differently than continuous variables, storing data as factors insures that the modeling functions will treat such data correctly.

One can think of a factor as an integer vector where each integer has a label[[1]](#footnote-23). In fact, factors are built on top of integer vectors using two attributes: the class() "factor", which makes them behave differently from regular integer vectors, and the levels(), which defines the set of allowed values[[2]](#footnote-25).

In this chapter I will cover the basics of dealing with factors which includes [Creating, converting & inspecting factors](#factors_creating), [Ordering levels](#factors_ordering), [Revaluing levels](#factors_revalue), and [Dropping levels](#factors_dropping).

## Creating, converting & inspecting factors

Factor objects can be created with the factor() function:

# create a factor string  
gender <- factor(c("male", "female", "female", "male", "female"))  
gender  
## [1] male female female male female  
## Levels: female male  
  
# inspect to see if it is a factor class  
class(gender)  
## [1] "factor"  
  
# show that factors are just built on top of integers  
typeof(gender)  
## [1] "integer"  
  
# See the underlying representation of factor  
unclass(gender)  
## [1] 2 1 1 2 1  
## attr(,"levels")  
## [1] "female" "male"  
  
# what are the factor levels?  
levels(gender)  
## [1] "female" "male"  
  
# show summary of counts  
summary(gender)  
## female male   
## 3 2

If we have a vector of character strings or integers we can easily convert to factors:

group <- c("Group1", "Group2", "Group2", "Group1", "Group1")  
str(group)  
## chr [1:5] "Group1" "Group2" "Group2" "Group1" "Group1"  
  
# convert from characters to factors  
as.factor(group)  
## [1] Group1 Group2 Group2 Group1 Group1  
## Levels: Group1 Group2

## Ordering levels

When creating a factor we can control the ordering of the levels by using the levels argument:

# when not specified the default puts order as alphabetical  
gender <- factor(c("male", "female", "female", "male", "female"))  
gender  
## [1] male female female male female  
## Levels: female male  
  
# specifying order  
gender <- factor(c("male", "female", "female", "male", "female"),   
 levels = c("male", "female"))  
gender  
## [1] male female female male female  
## Levels: male female

We can also create ordinal factors in which a specific order is desired by using the ordered = TRUE argument. This will be reflected in the output of the levels as shown below in which low < middle < high:

ses <- c("low", "middle", "low", "low", "low", "low", "middle", "low", "middle",  
 "middle", "middle", "middle", "middle", "high", "high", "low", "middle",  
 "middle", "low", "high")  
  
# create ordinal levels  
ses <- factor(ses, levels = c("low", "middle", "high"), ordered = TRUE)  
ses  
## [1] low middle low low low low middle low middle middle  
## [11] middle middle middle high high low middle middle low high   
## Levels: low < middle < high  
  
# you can also reverse the order of levels if desired  
factor(ses, levels = rev(levels(ses)))  
## [1] low middle low low low low middle low middle middle  
## [11] middle middle middle high high low middle middle low high   
## Levels: high < middle < low

## Revalue levels

To recode factor levels I usually use the revalue() function from the plyr package.

plyr::revalue(ses, c("low" = "small", "middle" = "medium", "high" = "large"))  
## [1] small medium small small small small medium small medium medium  
## [11] medium medium medium large large small medium medium small large   
## Levels: small < medium < large

Note that Using the :: notation allows you to access the revalue() function without having to fully load the plyr package.

## Dropping levels

When you want to drop unused factor levels, use droplevels():

ses2 <- ses[ses != "middle"]  
  
# lets say you have no observations in one level  
summary(ses2)  
## low middle high   
## 8 0 3  
  
# you can drop that level if desired  
droplevels(ses2)  
## [1] low low low low low low high high low low high  
## Levels: low < high

1. <https://leanpub.com/rprogramming> [↑](#footnote-ref-23)
2. <http://adv-r.had.co.nz/Data-structures.html> [↑](#footnote-ref-25)