Factors

David Gerard

2019-02-25

Learning Objectives

- Manipulating factors.
- Chapter 15 of RDS.
- Wrangling Categorical Data in R.
- 8.2: Chimeras of the R Inferno
- Factors with forcats Cheat Sheet.

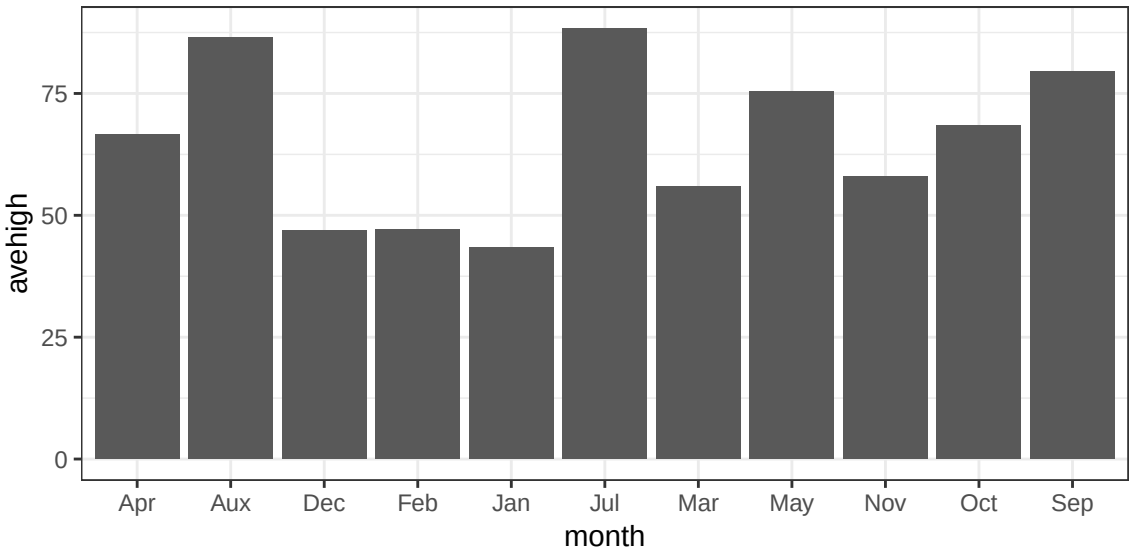
Factors

- A "factor" is R's way to say that a variable is categorical (places observational/experimental units into different groups or categories based on its values.).
- A factor is different from a character in that:
    1. There is a small predefined set of "levels" (possible values) of a factor, but not of a character.
    2. There is an ordering for the levels of a factor
        - Useful when determining the order to plot something.
        - Useful when doing orderd logistic regression.
- Consider the following data frame for average highs in DC for each month.

```
library(tidyverse)
dcclimate <- tribble(~month, ~avehigh,
                     ##----/---------
                     "Jan",  43.4,
                     "Feb",  47.1,
                     "Mar",  55.9,
                     "Apr",  66.6,
                     "May",  75.4,
                     "Jul",  88.4,
                     "Aux",  86.5,
                     "Sep",  79.5,
                     "Oct",  68.4,
                     "Nov",  57.9,
                     "Dec",  46.8)
```

- The weather for June is missing and the 3-letter abbreviation for August is incorrect. We would like to notice both of these.
- Also, when we plot the data, we would prefer the order to be the same as that for the order of the months of the year.

```
ggplot(dcclimate, aes(x = month, y = avehigh)) +
  geom_col()
```



- Factors help us with all of these issues.
- You have to be **very** careful about factors.

```
x  <- c("51", "32", "15", "2", "32")
xf <- factor(x)
as.numeric(x)
```

```
## [1] 51 32 15  2 32
```

```
as.numeric(xf)
```

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

```
as.numeric("Hello")
```

```
## Warning: NAs introduced by coercion
```

```
## [1] NA
```

```
as.numeric(factor("Hello"))
```

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

```
fac1 <- factor(c("x1", "x2", "x3"))
fac2 <- factor(c("y1", "y2", "y3"))
c(fac1, fac2)
```

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

Creating Factors

- Use `factor()` or `parse_factor()` to create a factor variable
- `parse_factor()` returns better warnings, so I would recommend always using that.

```
monthvec <- c("Jan", "Feb", "Mar", "Apr", "May", "Jun",
              "Jul", "Aug", "Sep", "Oct", "Nov", "Dec")
dcclimate %>%
  mutate(monthfc = factor(month, levels = monthvec)) ->
  dcclimate

dcclimate %>%
  mutate(monthfc2 = parse_factor(month, levels = monthvec)) ->
  dcclimate
```

```
## Warning: 1 parsing failure.
## row col           expected actual
##   7  -- value in level set    Aux
```

```
dcclimate$monthfc
```

```
##  [1] Jan  Feb  Mar  Apr  May  Jul  <NA> Sep  Oct  Nov  Dec
## Levels: Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
```

- If you do not specefy the `levels` argument, R will assume that the levels are the unique values of the vector
  - `factor()` takes the order of the levels to be the same order returned by `sort()`.
  - `parse_factor()` takes the order of the levels to be the same order as the order of the value introduced.

```
x <- c("A", "string", "vector", "is", "a", "string", "vector")
factor(x)
```

```
## [1] A      string vector is     a      string vector
## Levels: a A is string vector
```

```
sort(unique(x))
```

```
## [1] "a"      "A"      "is"     "string" "vector"
```

```
parse_factor(x)
```

```
## [1] A      string vector is     a      string vector
## Levels: A string vector is a
```

- You can always see the levels of a factor (and their order) using the `levels()` function

```
levels(dcclimate$monthfc)
```

```
##  [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov"
## [12] "Dec"
```

- More elegant is the `count()` function:

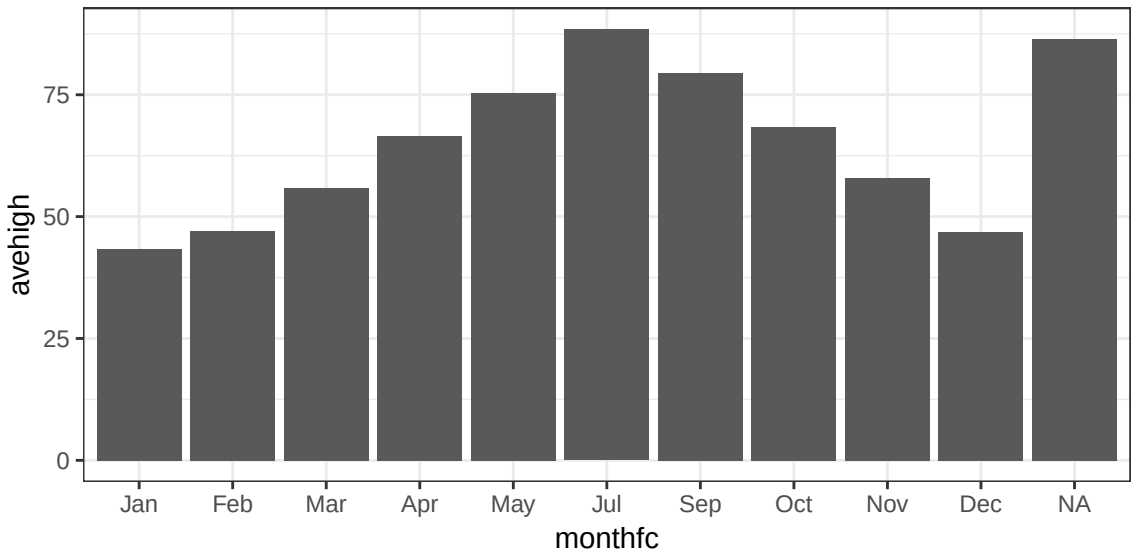
```
dcclimate %>%
  count(monthfc)
```

```
## Warning: Factor `monthfc` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
## # A tibble: 11 x 2
##    monthfc     n
##    <fct>   <int>
##  1 Jan         1
##  2 Feb         1
##  3 Mar         1
##  4 Apr         1
##  5 May         1
##  6 Jul         1
##  7 Sep         1
##  8 Oct         1
##  9 Nov         1
## 10 Dec         1
## 11 <NA>        1
```

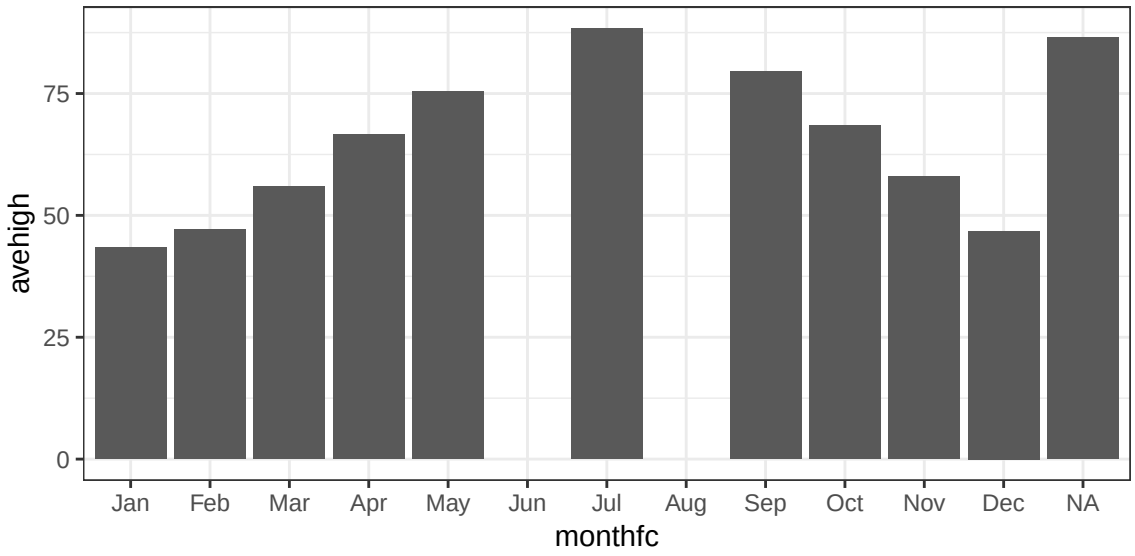
- Once we have a factor variable, the order of the aesthetic map is set in ggplot.

```
ggplot(dcclimate, aes(x = monthfc, y = avehigh)) +
  geom_col()
```



- We can include missing levels by using the `drop = FALSE` argument in the appropriate scale call:

```
ggplot(dcclimate, aes(x = monthfc, y = avehigh)) +
  geom_col() +
  scale_x_discrete(drop = FALSE)
```



forcats

- forcats is an R package which makes two things much easier in R:
  - Changing the order of the levels of the factor variable.
  - Changing the levels of the factor variable.
- forcats is a part of the tidyverse, so you don't need to load it separately when you load the tidyverse.

Changing the Order of the Levels

- Consider the subset of the General Social Survey stored in the `gss_cat` data in forcats.

```
data(gss_cat)
glimpse(gss_cat)
```

```
## Observations: 21,483
## Variables: 9
## $ year    <int> 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000, ...
## $ marital <fct> Never married, Divorced, Widowed, Never married, Divor...
## $ age     <int> 26, 48, 67, 39, 25, 25, 36, 44, 44, 47, 53, 52, 52, 51...
## $ race    <fct> White, White, White, White, White, White, White, White...
## $ rincome <fct> $8000 to 9999, $8000 to 9999, Not applicable, Not appl...
## $ partyid <fct> "Ind,near rep", "Not str republican", "Independent", "...
## $ relig   <fct> Protestant, Protestant, Protestant, Orthodox-christian...
## $ denom   <fct> Southern baptist, Baptist-dk which, No denomination, N...
## $ tvhours <int> 12, NA, 2, 4, 1, NA, 3, NA, 0, 3, 2, NA, 1, NA, 1, 7, ...
```

- You often want to change the order of the levels of a factor to make plots more insightful.

```
gss_cat %>%
  group_by(relig) %>%
  summarize(tvhours_mean = mean(tvhours, na.rm = TRUE)) ->
  tvdat

ggplot(tvdat, aes(x = tvhours_mean, y = relig)) +
  geom_point() +
  xlab("Average TV Hours") +
  ylab("Religion")
```