Explore data with tidyverse

Seun Odeyemi

October 9, 2018

Contents

```
knitr::opts_knit$set(root.dir = "~/tidyverse/datasets/")
```

```
devtools::session_info()
```

```
## Session info -------------------------------------------------------------
```

```
##  setting  value
##  version  R version 3.4.3 (2017-11-30)
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  English_United States.1252
##  tz       America/New_York
##  date     2018-10-09
```

```
## Packages -----------------------------------------------------------------
```

```
##  package   * version date       source
##  backports   1.1.2   2017-12-13 CRAN (R 3.4.3)
##  base      * 3.4.3   2017-12-06 local
##  compiler    3.4.3   2017-12-06 local
##  datasets  * 3.4.3   2017-12-06 local
##  devtools    1.13.4  2017-11-09 CRAN (R 3.4.3)
##  digest      0.6.15  2018-01-28 CRAN (R 3.4.3)
##  evaluate    0.10.1  2017-06-24 CRAN (R 3.4.3)
##  graphics  * 3.4.3   2017-12-06 local
##  grDevices * 3.4.3   2017-12-06 local
##  htmltools   0.3.6   2017-04-28 CRAN (R 3.4.3)
##  knitr       1.19    2018-01-29 CRAN (R 3.4.3)
##  magrittr    1.5     2014-11-22 CRAN (R 3.4.3)
##  memoise     1.1.0   2017-04-21 CRAN (R 3.4.3)
##  methods   * 3.4.3   2017-12-06 local
##  pillar      1.1.0   2018-01-14 CRAN (R 3.4.3)
##  Rcpp        0.12.15 2018-01-20 CRAN (R 3.4.3)
##  rlang       0.2.2   2018-08-16 CRAN (R 3.4.4)
##  rmarkdown   1.8.8   2018-01-28 Github (rstudio/rmarkdown@90475fe)
##  rprojroot   1.3-2   2018-01-03 CRAN (R 3.4.3)
##  stats     * 3.4.3   2017-12-06 local
##  stringi     1.1.6   2017-11-17 CRAN (R 3.4.2)
##  stringr     1.2.0   2017-02-18 CRAN (R 3.4.3)
##  tibble      1.4.2   2018-01-22 CRAN (R 3.4.3)
##  tools       3.4.3   2017-12-06 local
##  utils     * 3.4.3   2017-12-06 local
##  withr       2.1.1   2017-12-19 CRAN (R 3.4.3)
##  yaml        2.1.16  2017-12-12 CRAN (R 3.4.3)
```

Load Packages

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.4.4
```

```
## Warning: package 'tidyr' was built under R version 3.4.4
```

```
## Warning: package 'readr' was built under R version 3.4.4
```

```
## Warning: package 'purrr' was built under R version 3.4.4
```

```
## Warning: package 'forcats' was built under R version 3.4.4
```

```
library(skimr)
```

```
## Warning: package 'skimr' was built under R version 3.4.4
```

Import Data

```
bakeoff <- read_csv("bakeoff.csv",
                    na = c("", "NA", "UNKNOWN"))
```

```
## Parsed with column specification:
## cols(
##   series = col_integer(),
##   episode = col_integer(),
##   baker = col_character(),
##   signature = col_character(),
##   technical = col_integer(),
##   showstopper = col_character(),
##   result = col_character(),
##   uk_airdate = col_date(format = ""),
##   us_season = col_integer(),
##   us_airdate = col_date(format = "")
## )
```

```
bakeoff %>% filter(is.na(showstopper))
```

```
## # A tibble: 21 x 10
##    series episode baker  signature technical showstopper result uk_airdate
##     <int>   <int> <chr>  <chr>         <int> <chr>       <chr>  <date>
##  1      1       1 Edd    Caramel ~         1 <NA>        IN     2010-08-17
##  2      1       1 Jasmi~ Fresh Ma~        NA <NA>        IN     2010-08-17
##  3      1       6 Miran~ Lemon Cu~        NA <NA>        RUNNE~ 2010-09-21
##  4      2       1 Ian    Apple an~        10 <NA>        IN     2011-08-16
##  5      2       1 Jason  "Lemon M~         6 <NA>        IN     2011-08-16
##  6      2       1 Urvas~ Cherry B~         7 <NA>        IN     2011-08-16
##  7      2       1 Yasmin Cardamom~         5 <NA>        IN     2011-08-16
##  8      2       1 Holly  "Cherry ~         1 <NA>        SB     2011-08-16
##  9      2       2 Ben    Chorizo,~         1 <NA>        IN     2011-08-23
## 10      2       2 Ian    "Stilton~         2 <NA>        IN     2011-08-23
## # ... with 11 more rows, and 2 more variables: us_season <int>,
## #   us_airdate <date>
```

```
# convert the "series" variable type from integer to factor
bakeoff <- bakeoff %>% mutate(series = as.factor(series))
```

Getting summary statistics

Arrange and glimpse

- Question: On which data date did the first episode of the show air in the US?

```
bakeoff %>% arrange(us_airdate) %>% glimpse()
```

```
## Observations: 549
## Variables: 10
## $ series      <fct> 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,...
## $ episode     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2,...
## $ baker       <chr> "Ali", "Beca", "Christine", "Deborah", "Frances", ...
## $ signature   <chr> "Rose and Pistachio Cake", "Grapefruit Sandwich Ca...
## $ technical   <int> 11, 8, 3, 9, 7, 4, 6, 10, 2, 5, 12, 13, 1, 9, 11, ...
## $ showstopper <chr> "Chocolate, Raspberry and Passion Fruit Engagement...
## $ result      <chr> "IN", "IN", "IN", "IN", "IN", "IN", "IN", "IN", "I...
## $ uk_airdate  <date> 2013-08-20, 2013-08-20, 2013-08-20, 2013-08-20, 2...
## $ us_season   <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ us_airdate  <date> 2014-12-28, 2014-12-28, 2014-12-28, 2014-12-28, 2...
```

- Answer: 2014-12-28

Summarizing data with the skimr function skim

```
bakeoff %>%
  skim() %>%  # no argument needed here
  summary() # no argument needed here
```

```
## A skim object
##
## Number of Rows: 549
## Number of Columns: 10
##
## Column type frequency
## character: 4
## Date: 2
## factor: 1
## integer: 3
```

Counting data

- How many distinct series are available in the data?

```
bakeoff %>% distinct(series)
```

```
## # A tibble: 8 x 1
##   series
##   <fct>
## 1 1
## 2 2
## 3 3
## 4 4
## 5 5
## 6 6
## 7 7
## 8 8
```

There are eight distinct series.

Counting data with the dplyr function count

- How can we count the number of distinct bakers for each series?

```
bakeoff %>% count(series)
```

```
## # A tibble: 8 x 2
##   series     n
##   <fct>  <int>
## 1 1         36
## 2 2         60
## 3 3         76
## 4 4         78
## 5 5         74
## 6 6         75
## 7 7         75
## 8 8         75
```

```
# Using count produces the same output as the code below

bakeoff %>%
  group_by(series) %>%
  summarize(n = n())
```

```
## # A tibble: 8 x 2
##   series     n
##   <fct>  <int>
## 1 1         36
## 2 2         60
## 3 3         76
## 4 4         78
## 5 5         74
## 6 6         75
## 7 7         75
## 8 8         75
```

The above table shows us the number of bakers that appeared in each series.

Count rows by two variables

Here, I had to create a new variable called `aired_us` based on the conditional statement that checks if `us_season` is populated. If `us_season` is not populated assign *FALSE* to `aired_us`; if populated assign *TRUE*.

```
bakeoff <- bakeoff %>% mutate(aired_us =
                      ifelse(is.na(us_season), FALSE,
                             ifelse(us_season != '', TRUE, NA_complex_)))
```

Now, we can go ahead and answer the question: how many series have aired in the US?

```
bakeoff %>%
  count(aired_us, series)
```

```
## # A tibble: 8 x 3
##   aired_us series     n
##   <lgl>    <fct>  <int>
## 1 F        1         36
```

```
## 2 F        2         60
## 3 F        3         76
## 4 F        8         75
## 5 T        4         78
## 6 T        5         74
## 7 T        6         75
## 8 T        7         75
```

We can see that series 4, 5, 6, and 7 have aired in the US.

What if we are interested in checking for the proportion of bakers in each series for the whole show?

```
bakeoff %>%
  count(aired_us, series) %>%
  mutate(prop_bakers = n/sum(n)) %>%
  mutate(prop_bakers = sprintf("%0.3f", prop_bakers))
```

```
## # A tibble: 8 x 4
##   aired_us series     n prop_bakers
##   <lgl>    <fct>  <int> <chr>
## 1 F        1         36 0.066
## 2 F        2         60 0.109
## 3 F        3         76 0.138
## 4 F        8         75 0.137
## 5 T        4         78 0.142
## 6 T        5         74 0.135
## 7 T        6         75 0.137
## 8 T        7         75 0.137
```

The `count` function also ungroups a dataframe for us. For instance, if we group_by then summarize, we will get a surprisly different result from the one we go with the count function.

```
bakeoff %>%
  group_by(aired_us, series) %>%
  summarize(n = n()) %>%
  mutate(prop_bakers = n/sum(n)) %>%
  mutate(prop_bakers = sprintf("%0.3f", prop_bakers))
```

```
## # A tibble: 8 x 4
## # Groups:   aired_us [2]
##   aired_us series     n prop_bakers
##   <lgl>    <fct>  <int> <chr>
## 1 F        1         36 0.146
## 2 F        2         60 0.243
## 3 F        3         76 0.308
## 4 F        8         75 0.304
## 5 T        4         78 0.258
## 6 T        5         74 0.245
## 7 T        6         75 0.248
## 8 T        7         75 0.248
```

Compare this with

```
bakeoff %>%
  group_by(aired_us, series) %>%
  summarize(n = n()) %>%
  ungroup() %>%
  mutate(prop_bakers = n/sum(n)) %>%
```

```
  mutate(prop_bakers = sprintf("%0.3f", prop_bakers))
```

```
## # A tibble: 8 x 4
##   aired_us series     n prop_bakers
##   <lgl>    <fct>  <int> <chr>
## 1 F        1         36 0.066
## 2 F        2         60 0.109
## 3 F        3         76 0.138
## 4 F        8         75 0.137
## 5 T        4         78 0.142
## 6 T        5         74 0.135
## 7 T        6         75 0.137
## 8 T        7         75 0.137
```

They both do the same thing, but using count we have a more parsimonious code.

Count to roll up a level

```
bakeoff %>%
  count(aired_us, series) %>%
  count(aired_us)
```

```
## # A tibble: 2 x 2
##   aired_us    nn
##   <lgl>    <int>
## 1 F            4
## 2 T            4
```

The column name `nn` is the number of series that aired in the US. US viewers have seen half of the series aired in the UK.

```
bakeoff %>%
  count(result)
```

```
## # A tibble: 6 x 2
##   result        n
##   <chr>     <int>
## 1 IN          393
## 2 LEFT          1
## 3 OUT          70
## 4 RUNNER UP    16
## 5 SB           61
## 6 WINNER        8
```

```
bakeoff %>% count(result == "SB")
```

```
## # A tibble: 2 x 2
##   `result == "SB"`     n
##   <lgl>            <int>
## 1 F                  488
## 2 T                   61
```

```
# Count the number of rows by series and episode
bakeoff %>% count(series, episode)
```

```
## # A tibble: 74 x 3
```

```
##    series episode     n
##    <fct>    <int> <int>
##  1 1            1    10
##  2 1            2     8
##  3 1            3     6
##  4 1            4     5
##  5 1            5     4
##  6 1            6     3
##  7 2            1    12
##  8 2            2    11
##  9 2            3    10
## 10 2            4     8
## # ... with 64 more rows
```

```
# Add second count by series
bakeoff %>%
  count(series, episode) %>%
  count(series)
```

```
## # A tibble: 8 x 2
##   series    nn
##   <fct>  <int>
## 1 1          6
## 2 2          8
## 3 3         10
## 4 4         10
## 5 5         10
## 6 6         10
## 7 7         10
## 8 8         10
```

nn is the number of episodes in each series.

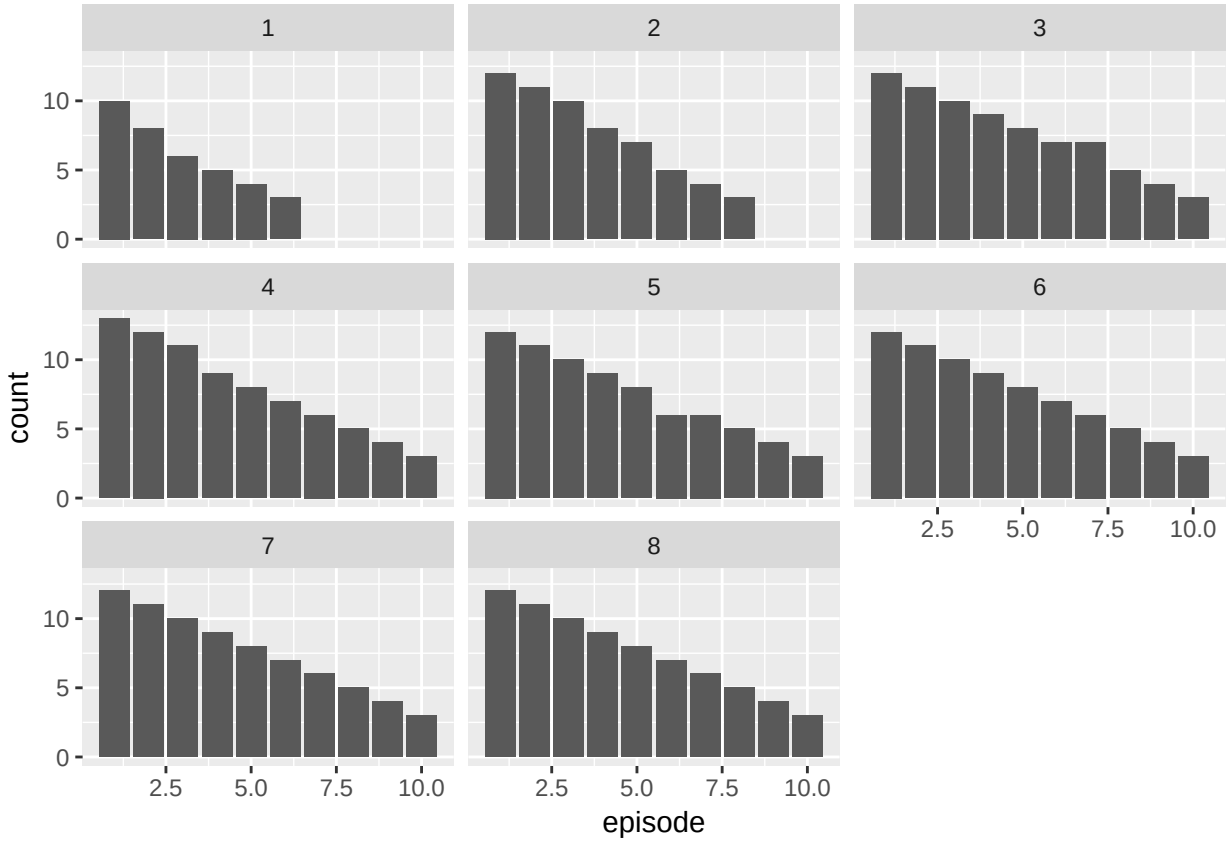
```
# Count the number of rows by series and baker
bakers_by_series <- bakeoff %>%
  count(series, baker)

# Print to view
# bakers_by_series

# Count again by series
bakers_by_series %>% count(series)
```

```
## # A tibble: 8 x 2
##   series    nn
##   <fct>  <int>
## 1 1         10
## 2 2         12
## 3 3         12
## 4 4         13
## 5 5         12
## 6 6         12
## 7 7         12
## 8 8         12
```

```
# Count again by baker
bakers_by_series %>% count(baker, sort = TRUE)
```

```
## # A tibble: 86 x 2
##    baker     nn
##    <chr>  <int>
##  1 Kate       3
##  2 Ian        2
##  3 James      2
##  4 Louise     2
##  5 Mark       2
##  6 Peter      2
##  7 Robert     2
##  8 Tom        2
##  9 Ali        1
## 10 Alvin      1
## # ... with 76 more rows
```

```
ggplot(bakeoff, aes(x = episode)) +
    geom_bar() +
    facet_wrap(~series)
```



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.