IS5 in R: Relationships Between Categorical Variables–Contingency Tables (Chapter 3)

Nicholas Horton (nhorton@amherst.edu)

2025-01-20

Table of contents

Introduction and background

This document is intended to help describe how to undertake analyses introduced as examples in the Fifth Edition of *Intro Stats* (2018) by De Veaux, Velleman, and Bock. This file as well as the associated Quarto reproducible analysis source file used to create it can be found at http://nhorton.people.amherst.edu/is5.

This work leverages initiatives undertaken by Project MOSAIC (http://www.mosaic-web.org), an NSF-funded effort to improve the teaching of statistics, calculus, science and computing in the undergraduate curriculum. In particular, we utilize the `mosaic` package, which was written to simplify the use of R for introductory statistics courses. A short summary of the R needed to teach introductory statistics can be found in the mosaic package vignettes (https://cran.r-project.org/web/packages/mosaic). A paper describing the mosaic approach was published in the *R Journal*: https://journal.r-project.org/archive/2017/RJ-2017-024.

We begin by loading packages that will be required for our analyses.

```
library(mosaic)
library(tidyverse)
```

Chapter 3: Relationships Between Categorical Variables–Contingency Tables

Section 3.1: Contingency Tables

```
library(janitor)
OKCupid <-
  read_csv(
    "http://nhorton.people.amherst.edu/is5/data/OKCupid_CatsDogs.csv",
    skip = 1
  ) |>
  janitor::clean_names()
```

The `read_csv()` function lists the input variable names by default. These were suppressed using the `message: false` code chunk option to save space. Here we use the `clean_names()` function from the `janitor` package to sanitize the names of the columns (which would otherwise contain special characters or whitespace). You can use the `names()` function to check the cleaned names. We use `skip = 1` because the first line in the original data set is a set of variable labels (e.g., `Col1`, `Col2`).

```
names(OKCupid)
```

```
[1] "cats_dogs_both" "gender"         "drugs_y_n"      "smokes_y_n"
```

The `names()` function is an easy way to see what variables are included in a dataset.

```
glimpse(OKCupid)
```

```
Rows: 59,943
Columns: 4
$ cats_dogs_both <chr> NA, NA, "Has cats", NA, NA, NA, NA, NA, NA, NA, NA, NA,~
$ gender         <chr> "M", "M", "M", "M", "M", "M", "F", "F", "F", "M", "M", ~
$ drugs_y_n      <chr> "No", "Yes", NA, NA, "No", NA, "No", "No", NA, "No", NA~
$ smokes_y_n     <chr> "yes", "no", "no", "no", "no", "no", NA, "no", "yes", "~
```

The `glimpse()` function provides more information.

```
# Table 3.1, page 65
tally(~ cats_dogs_both + gender, margin = TRUE, useNA = "no", data = OKCupid)
```

```
              gender
cats_dogs_both    F    M Total
      Has Both  897  577  1474
      Has cats 3412 2388  5800
      Has dogs 3431 3587  7018
      Total    7740 6552 14292
```

```
# Table 3.2
tally(~ cats_dogs_both + gender,
  format = "percent", margin = TRUE, useNA = "no",
  data = OKCupid
)
```

```
              gender
cats_dogs_both         F         M      Total
      Has Both  6.276238  4.037224  10.313462
      Has cats 23.873496 16.708648  40.582144
      Has dogs 24.006437 25.097957  49.104394
      Total    54.156171 45.843829 100.000000
```

```
tally(cats_dogs_both ~ gender,
  format = "percent", margin = TRUE, useNA = "no",
  data = OKCupid
)
```

```
              gender
cats_dogs_both         F         M
      Has Both  11.589147  8.806471
      Has cats  44.082687 36.446886
      Has dogs  44.328165 54.746642
      Total    100.000000 100.000000
```

```
# Table 3.3
tally(gender ~ cats_dogs_both, format = "percent", margin = TRUE, data = OKCupid)
```

```
       cats_dogs_both
gender   Has Both  Has cats  Has dogs      <NA>
  F      60.85482  58.82759  48.88857  35.87435
  M      39.14518  41.17241  51.11143  64.12565
  Total 100.00000 100.00000 100.00000 100.00000
```

We note that the logical values `TRUE` and `FALSE` are all caps in R, but that code chunk options are all lower-case (e.g., `message: false`).

Example 3.1: Exploring Marginal Distributions

We begin by reading and tallying the data.

```
SuperBowl <-
  read_csv(
    "http://nhorton.people.amherst.edu/is5/data/Watch_the_Super_bowl.csv",
    skip = 1
  )
tally(~ Plan + Sex, data = SuperBowl)
```

```
             Sex
Plan          Female Male
  Commercials    156   81
  Game           200  279
  Wont Watch     160  132
```

Example 3.2: Exploring Percentages: Children and First-Class Ticket Holders First?

We do the same for the Titanic data.

```
Titanic <- read_csv("http://nhorton.people.amherst.edu/is5/data/Titanic.csv")
tally(~ Class + Survived, format = "percent", margin = TRUE, data = Titanic)
```

```
       Survived
Class        Alive       Dead      Total
  1       9.103261   5.570652  14.673913
  2       5.389493   7.518116  12.907609
  3       8.152174  24.003623  32.155797
  Crew    9.601449  30.661232  40.262681
  Total  32.246377  67.753623 100.000000
```

```
tally(Class ~ Survived, format = "percent", margin = TRUE, data = Titanic)
```

```
       Survived
Class        Alive       Dead
  1      28.230337   8.221925
  2      16.713483  11.096257
  3      25.280899  35.427807
  Crew   29.775281  45.254011
  Total 100.000000 100.000000
```

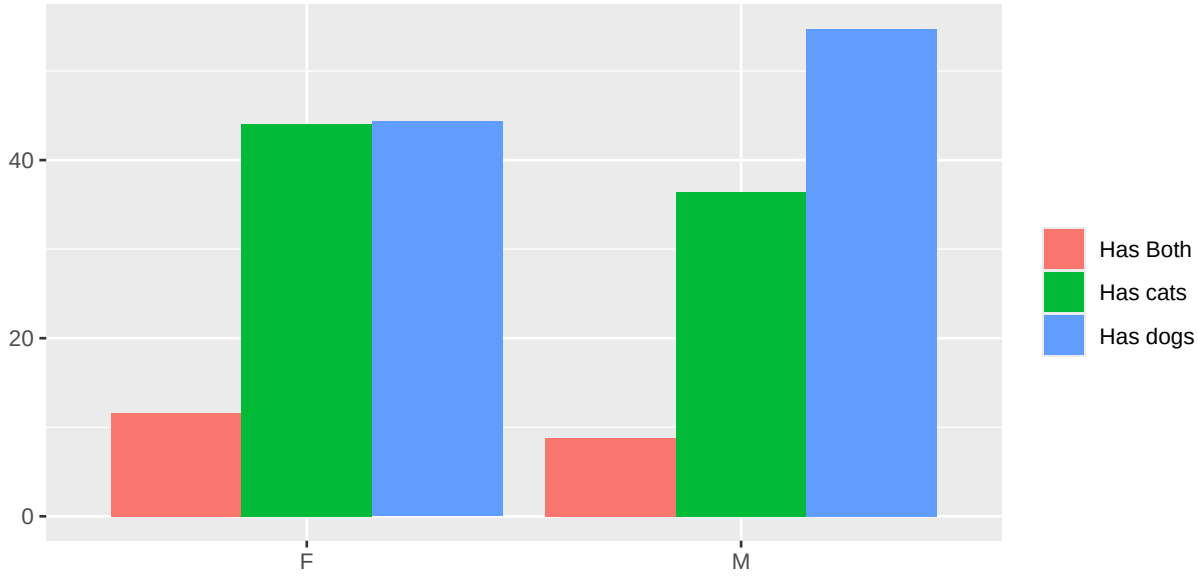
```
tally(Survived ~ Class, format = "percent", margin = TRUE, data = Titanic)
```

```
        Class
Survived         1         2         3      Crew
   Alive  62.03704  41.75439  25.35211  23.84702
   Dead   37.96296  58.24561  74.64789  76.15298
   Total 100.00000 100.00000 100.00000 100.00000
```

Section 3.2: Conditional Distributions

See displays on 68-69.

```
OKdata <- tally(
  cats_dogs_both ~ gender,
  format = "percent", useNA = "no",
  data = OKCupid
) |>
  data.frame()
# Figure 3.2, page 69
gf_col(Freq ~ gender, fill = ~ cats_dogs_both, position = "dodge", data = OKdata) |>
  gf_labs(x = "", y = "", fill = "")
```



Example 3.3: Finding Conditional Distributions: Watching the Super Bowl

We can calculate conditional probabilities from tables using `mosaic::tally()`.

```
tally(~ Plan + Sex, margin = TRUE, data = SuperBowl)
```

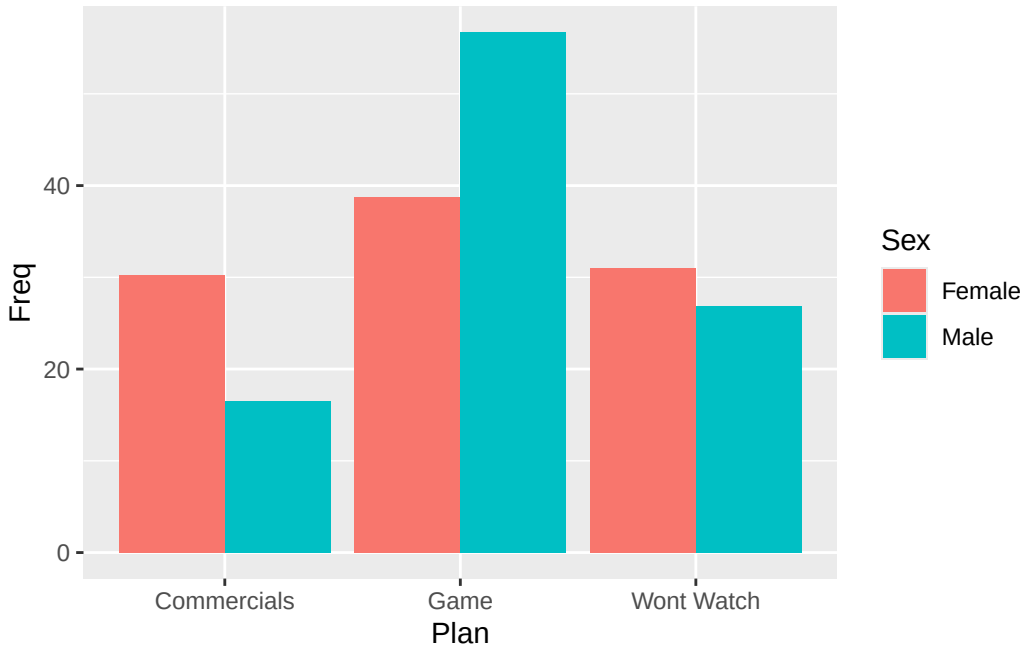
```
             Sex
Plan          Female Male Total
  Commercials    156   81   237
  Game           200  279   479
  Wont Watch     160  132   292
  Total          516  492  1008
```

```
tally(Plan ~ Sex, format = "percent", data = SuperBowl)
```

```
             Sex
Plan            Female     Male
  Commercials 30.23256 16.46341
  Game        38.75969 56.70732
  Wont Watch  31.00775 26.82927
```

Example 3.4: Looking for Associations Between Variables: Still Watching the Super Bowl

```
Superdata <- tally(Plan ~ Sex, format = "percent", data = SuperBowl) |>
  data.frame()
gf_col(Freq ~ Plan, fill = ~Sex, position = "dodge", data = Superdata)
```



Examining Contingency Tables

See displays on page 72.

```
FishDiet <- read_csv("http://nhorton.people.amherst.edu/is5/data/Fish_diet.csv", skip = 1) |>
  janitor::clean_names()
tally(~ diet_counts + cancer_counts, margins = TRUE, data = FishDiet)
```

```
           cancer_counts
diet_counts   No  Yes Total
   Large     507   42   549
   Moderate 2769  209  2978
   Never     110   14   124
   Small    2420  201  2621
   Total    5806  466  6272
```

Random Matters

See display on page 74.

```
Nightmares <-
  read_csv("http://nhorton.people.amherst.edu/is5/data/Nightmares.csv", skip = 1)
glimpse(Nightmares)
```

```
Rows: 63
Columns: 2
$ Side  <chr> "L", "L", "L", "L", "L", "L", "L", "L", "L", "L", "L", "L", "L",~
$ Dream <chr> "N", "N", "N", "N", "N", "N", "N", "N", "N", "n", "n", "n", "n",~
```

```
Nightmares <- Nightmares |>   # recode the `Dream` variable
  mutate(Dream = ifelse(Dream == "N", "Nightmare", "SweetDreams"))
glimpse(Nightmares)
```

```
Rows: 63
Columns: 2
$ Side  <chr> "L", "L", "L", "L", "L", "L", "L", "L", "L", "L", "L", "L", "L",~
$ Dream <chr> "Nightmare", "Nightmare", "Nightmare", "Nightmare", "Nightmare",~
```

Now we can calculate the contingency table.

```
tally(~ Dream + Side, margins = TRUE, data = Nightmares)
```

```
             Side
Dream          L  R Total
  Nightmare    9  6    15
  SweetDreams 13 35    48
  Total       22 41    63
```

Section 3.3: Displaying Contingency Tables

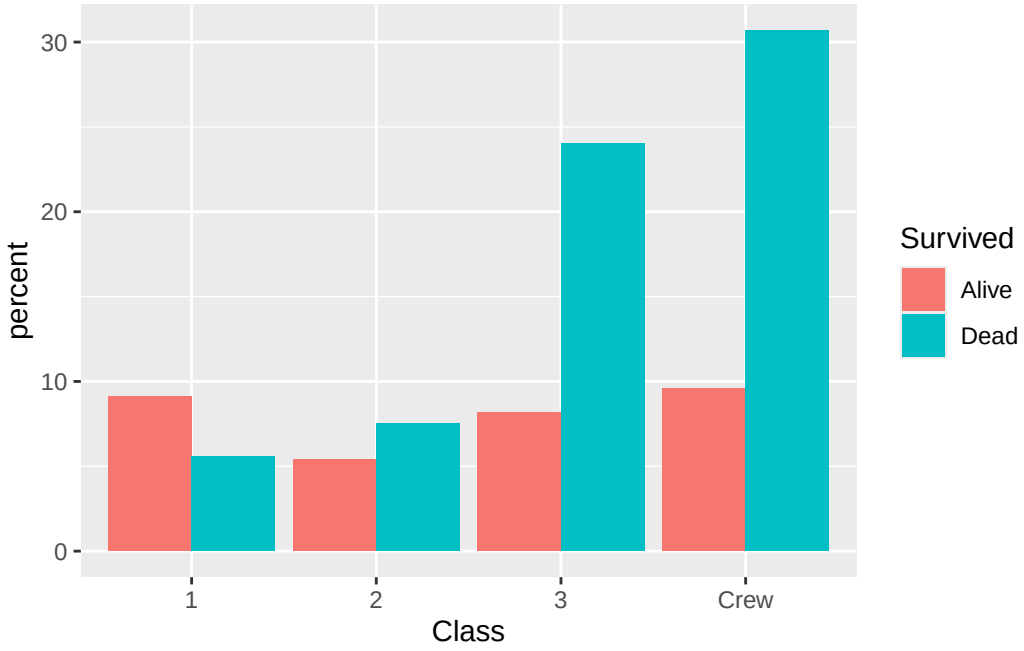
```
tally(~ Class + Survived, format = "count", data = Titanic)
```

```
      Survived
Class  Alive Dead
  1      201  123
  2      119  166
  3      180  530
  Crew   212  677
```

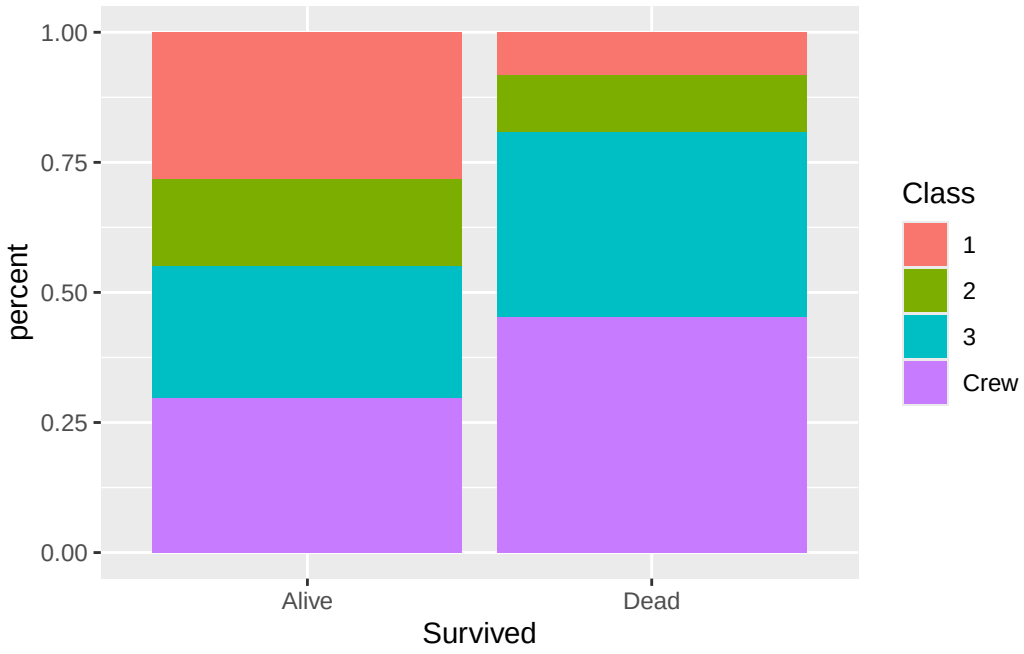
```
tally(~ Class + Survived, format = "percent", data = Titanic)
```

```
      Survived
Class      Alive      Dead
  1     9.103261  5.570652
  2     5.389493  7.518116
  3     8.152174 24.003623
  Crew  9.601449 30.661232
```

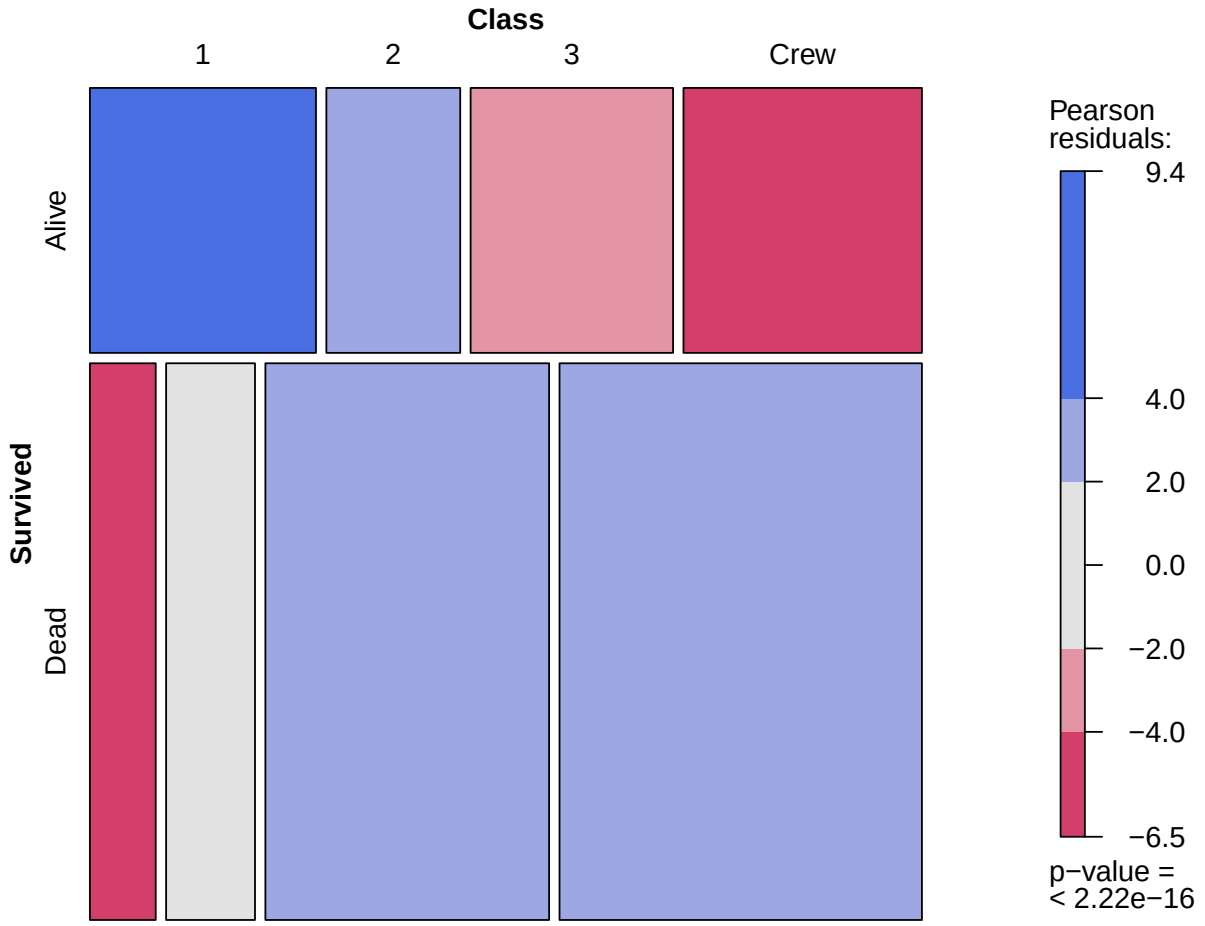
```
# Figure 3.4, page 75
gf_percents(~ Class, fill = ~ Survived, position = position_dodge(), data = Titanic)
```




```
# Figure 3.5
gf_percents(~ Survived, fill = ~ Class, position = "fill", data = Titanic)
```




```
# Figure 3.6, page 76
vcd::mosaic(tally(~ Survived + Class, data = Titanic),
  main = "Mosaic plot of Class by Survival",
  shade = TRUE
)
```



See the mosaic plots on page 77.

Section 3.4: Three Categorical Variables

```
tally(~ gender + cats_dogs_both + drugs_y_n, format = "percent", data = OKCupid)
```

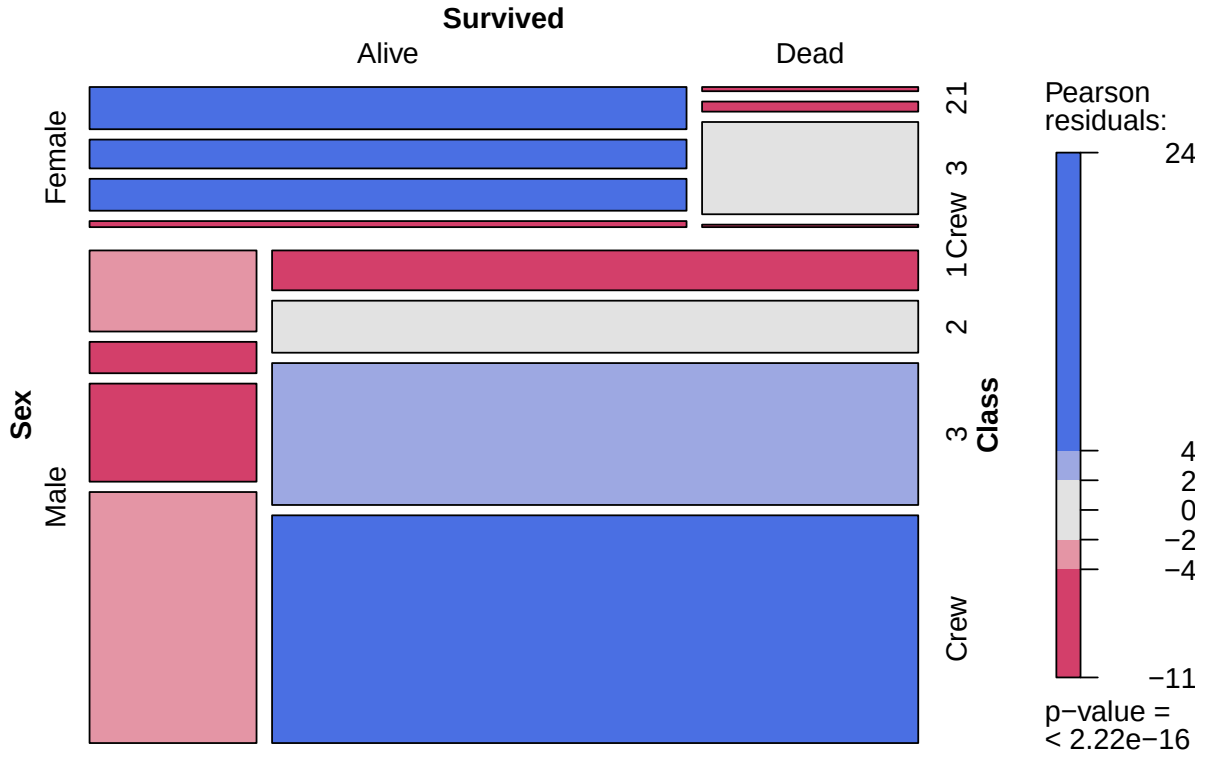
```
, , drugs_y_n = No

      cats_dogs_both
gender    Has Both   Has cats   Has dogs       <NA>
     F  1.0243064  3.4199156  3.9437466 18.0187845
```

```
     M  0.5922293  2.0819779  3.7769214 30.0719016

, , drugs_y_n = Yes

      cats_dogs_both
gender   Has Both   Has cats   Has dogs       <NA>
     F  0.2085314  0.8941828  0.6272626  2.9794972
     M  0.1901807  0.8658225  0.9041923  6.9132342

, , drugs_y_n = NA

      cats_dogs_both
gender   Has Both   Has cats   Has dogs       <NA>
     F  0.2635837  1.3779757  1.1527618  6.3226732
     M  0.1801712  1.0359842  1.3029044 11.8512587
```

Example 3.7: Looking for Associations Among Three Variables at Once

We can repeat the mosaic plot with three variables.

```
vcd::mosaic(tally(~ Sex + Survived + Class, data = Titanic), shade = TRUE)
```



Example 3.8: Simpson's Paradox: Gender Discrimination?

Here we demonstrate how to generate one of the tables on page 80.

```
# Create a dataframe from the counts
# http://mathemathinking.blogspot.com/2012/06/simpsons-paradox.html
Berk <- bind_rows(
  do(512) * data.frame(admit = TRUE, sex = "M", school = "A"),
  do(825 - 512) * data.frame(admit = FALSE, sex = "M", school = "A"),
  do(89) * data.frame(admit = TRUE, sex = "F", school = "A"),
  do(19) * data.frame(admit = FALSE, sex = "F", school = "A")
)
```

As noted previously, the logical values `TRUE` and `FALSE` are all caps in R, but that code chunk options are all lower-case (e.g., `message: false`).

Here, the `do(n)` function is used to create `n` observations with the specified values in `data.frame()`. The `bind_rows()` function can then be used to combine the data frames into one.

```
tally(~ sex + admit, data = Berk)
```

```
   admit
sex TRUE FALSE
  F   89    19
  M  512   313
```

```
tally(admit ~ sex, format = "percent", data = Berk)
```

```
       sex
admit          F        M
  TRUE  82.40741 62.06061
  FALSE 17.59259 37.93939
```