Chapter 5: Variables and Manipulation

Answers to Exercises

Brian Fogarty

Contents

Answers to Exercise 1

Exercise 1.a

There are a variety of ways we can do this. Below, we first use the `replace_na()` function to make "Did Not Vote" a missing value and then use `recode()` to rename the two other levels. We can do both within a single `mutate()` function; though we could do it in two separate `mutate()` functions.

```
library(tidyverse)
vf_england <- read_csv("VF England.csv")

vf_england %>%
  count(vote2017_dum)
```

```
# A tibble: 3 x 2
  vote2017_dum     n
  <chr>        <int>
1 Loser          919
2 Winner         757
3 <NA>           358
```

```
vf_england <- vf_england %>%
                mutate(vote2017_dum1 =
                         replace_na(vote2017_dum,"Did Not Vote"),
                         recode(vote2017_dum,
                                  "Loser"="Not Conservative Vote",
```

```
                                    "Winner"="Conservative Vote"))

vf_england %>%
  count(vote2017_dum1)
```

```
# A tibble: 3 x 2
  vote2017_dum1     n
  <chr>         <int>
1 Did Not Vote    358
2 Loser           919
3 Winner          757
```

Exercise 1.b

The level of measurement for the recoded version of `vote2017_dum` is nominal; same as the original version.

Answers to Exercise 2

Exercise 2.a

```
vf_england %>%
  count(vfalter)
```

```
# A tibble: 7 x 2
  vfalter                        n
  <chr>                      <int>
1 Agree                        166
2 Disagree                     253
3 Neither agree nor disagree   769
4 Slightly agree               364
5 Slightly disagree            266
6 Strongly agree               121
7 Strongly disagree             95
```

Based on the ordering of the categories seen above, `vfalter` is a nominal variable.

Exercise 2.b

To re-arrange the values of `vfalter`, we use the `factor()` function from the `forcats` package and the `levels=` option to order the labels. We also wrap everything with the `mutate()` function.

```
vf_england <- vf_england %>%
      mutate(vfalter1 = factor(vfalter,
               levels=c("Strongly disagree","Disagree",
                      "Slightly disagree","Neither agree nor disagree",
                      "Slightly agree","Agree","Strongly agree")))

vf_england %>%
  count(vfalter1)
```

```
# A tibble: 7 x 2
  vfalter1                       n
  <fct>                      <int>
1 Strongly disagree             95
```

```
2 Disagree                      253
3 Slightly disagree             266
4 Neither agree nor disagree    769
5 Slightly agree                364
6 Agree                         166
7 Strongly agree                121
```

Exercise 2.c

The level of measurement of the reordered version of `vfalter` is ordinal.

Answers to Exercise 3

We'll use the `fct_collapse()` function to collapse the values and then the `na_if()` function to make "Neither agree nor disagree" missing. We'll do this using two `mutate()` functions.

```
vf_england <- vf_england %>%
                mutate(vfalter2 = fct_collapse(vfalter1,
                        "Disagree" = c("Strongly disagree","Disagree",
                                       "Slightly disagree"),
                        "Agree" = c("Strongly agree","Agree",
                                    "Slightly agree"))) %>%
                mutate(vfalter2 = na_if(vfalter2, "Neither agree nor disagree"))

vf_england %>%
  count(vfalter2)
```

```
# A tibble: 3 x 2
  vfalter2     n
  <fct>    <int>
1 Disagree   614
2 Agree      651
3 <NA>       769
```

Answers to Exercise 4

First, we need to read-in the `simd2020.csv` dataset using the `read_csv()` function. Then, using `mutate()` to create the variable `pct_depress`, we simply need to multiply DEPRESS by 100 to convert it to a percentage.

```
simd <- read_csv("simd2020.csv", na = "*")

summary(simd$DEPRESS)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
 0.0000  0.1495  0.1868  0.1907  0.2276  0.4718       1
```

```
simd <- simd %>%
          mutate(pct_depress = DEPRESS*100)

summary(simd$pct_depress)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   0.00   14.95   18.68   19.07   22.76   47.18       1
```

Answers to Exercise 5

We'll use `case_when()` and a series of less than/greater than conditions to create the categorical version. We also need to reorder the levels to get the correct order. We'll do this using two `mutate()` functions.

```
simd <- simd %>%
          mutate(pct_depress_cat = case_when(
            pct_depress <= 10 ~ "Low",
            pct_depress > 10 & pct_depress <= 20 ~ "Medium",
            pct_depress > 20 ~ "High"
          )) %>%
          mutate(pct_depress_cat = factor(pct_depress_cat,
                                levels = c("Low","Medium","High")))

simd %>%
  count(pct_depress_cat)
```

```
# A tibble: 4 x 2
  pct_depress_cat     n
  <fct>           <int>
1 Low               175
2 Medium           3926
3 High             2874
4 <NA>                1
```