The basics: 02 Vectors and data types

Ari Anisfeld

9/8/2020

Questions

`ifelse`

We'll start using `ifelse` which is commonly used in data analysis with `mutate()`.

`midwest` is a dataset built into `tidyverse`

1. create a new variable called `poverty_designation` that is "High Poverty" if `percbelowpoverty` is above 10 and is "Low Poverty" otherwise.

   If you pipe your tibble into `count(poverty_designation)`, you should see

   ```
   ## # A tibble: 2 x 2
   ##   poverty_designation     n
   ##   <chr>               <int>
   ## 1 High Poverty          293
   ## 2 Low Poverty           144
   ```

2. Create a new variable that is "Ohio Counties" for observations from Ohio and "Other Midwestern Counties" for the rest of the observations.

3. Create a new variable that is `TRUE` for the observations from the counties "COOK", "WAYNE", "CUYAHOGA", "OAKLAND" or "FRANKLIN" and `FALSE` otherwise. Use the `%in%` operator.

4. In this problem, we'll simulate an election.

```r
election_simulation <-
  tibble(probabilty_vote = runif(1000),
         probability_support = runif(1000))
```

```
a. Using `mutate` and `ifelse` create a new column called `voter` that is 1 if the `probablity_vote` is

b. Create a second column called `supporter` that is 1 if `probablity_support` is over .4 and 0 otherwi

c. Create a third column that equals `TRUE` if `voter` and `supporter` are both equal to 1, that equals
```

Using `if`

We use `if()` when working on "statistical programming" (ie. when not working with tibbles for data analysis). We'll develop a small dice game.

1. Fill in the ... so the code says "You win" if the dice add up to 7 and "You lose" otherwise.

```r
dice <- sample(c(1:6), 2)

if (...) {
  print("You win")
} else {
  print("You lose")
}
```

2. Add an `else if()` block to the code above that says try again if the dice add up to 6 or 8.

Solution

1.
```r
midwest %>%
  mutate(poverty_designation = ifelse(percbelowpoverty > 10, "High Poverty", "Low Poverty")) %>%
  count(poverty_designation)
```

2.
```r
midwest %>%
  mutate(ohio = ifelse(state == "OH", "Ohio Counties", "Other Midwestern Counties"))
```

3.
```r
big_counties <- c("COOK", "WAYNE", "CUYAHOGA", "OAKLAND", "FRANKLIN")

midwest %>%
  mutate(populous_counties = ifelse(county %in% big_counties, 1, 0))
```

4.
```r
simulation <-
  tibble(probabilty_vote = runif(1000),
         probability_support = runif(1000)) %>%
  mutate(voter = ifelse(probabilty_vote > .5, 1, 0),
         supporter = ifelse(probability_support > .4, 1, 0),
         results = case_when(voter == 1 & supporter == 1 ~ TRUE,
                             voter == 1 & supporter == 0 ~ FALSE,
                             TRUE ~ NA))

# An alternative approach takes advantage of the structure of the data

simulation <-
  tibble(probabilty_vote = runif(1000),
         probability_support = runif(1000)) %>%
  mutate(voter = ifelse(probabilty_vote > .5, 1, 0),
         supporter = ifelse(probability_support > .4, 1, 0),
         results = ifelse(voter == 1, supporter * voter, NA))
```

if

~~~
```r
dice <- sample(c(1:6), 2)

if (sum(dice) == 7) {
  print("You win")
} else if(sum(dice) %in% c(6,8)) {
  print ("Try again")
~~~

~~~
} else {
  print("You lose")
}
```

```
## [1] "You lose"
```
~~~