Providing relocation assistance

PROGRAMMING WITH DPLYR

Dr. Chester Ismay

Educator, Data Scientist, and R/Python Consultant

The column names of world_bank_data

```
names(world_bank_data)
```

```
 [1] "iso"                 "country"               "continent"
 [4] "region"              "year"                  "infant_mortality_rate"
 [7] "fertility_rate"      "perc_electric_access"  "perc_college_complete"
[10] "perc_cvd_crd_70"     "unemployment_rate"     "perc_rural_pop"
```

Moving with select()

```
reordered_wb <- world_bank_data %>%
    select(iso:year,
    matches("^perc"),
    everything())
```

```
names(reordered_wb)
```

```
 [1] "iso"                    "country"                "continent"
 [4] "region"                 "year"                   "perc_electric_access"
 [7] "perc_college_complete"  "perc_cvd_crd_70"        "perc_rural_pop"
[10] "infant_mortality_rate"  "fertility_rate"         "unemployment_rate"
```

Using last_col() instead

```
world_bank_data %>%
    select(iso:year,
    matches("^perc"),
    infant_mortality_rate:last_col()) %>%
    names()
```

```
 [1] "iso"                    "country"              "continent"
 [4] "region"                 "year"                 "perc_electric_access"
 [7] "perc_college_complete"  "perc_cvd_crd_70"      "perc_rural_pop"
[10] "infant_mortality_rate"  "fertility_rate"       "unemployment_rate"
```

A simpler way with relocate() and .after

```
world_bank_data %>%
    relocate(matches("^perc"), .after = year) %>%
    names()
```

```
 [1] "iso"                    "country"                "continent"
 [4] "region"                 "year"                   "perc_electric_access"
 [7] "perc_college_complete"  "perc_cvd_crd_70"        "perc_rural_pop"
[10] "infant_mortality_rate"  "fertility_rate"         "unemployment_rate"
```

relocate() with .before

```
world_bank_data %>%
  relocate(matches("^perc"),
           .before = infant_mortality_rate) %>%
  names()
```

```
 [1] "iso"                    "country"              "continent"
 [4] "region"                 "year"                 "perc_electric_access"
 [7] "perc_college_complete"  "perc_cvd_crd_70"      "perc_rural_pop"
[10] "infant_mortality_rate"  "fertility_rate"       "unemployment_rate"
```

Let's practice!

PROGRAMMING WITH DPLYR

That has crossed the line

PROGRAMMING WITH DPLYR

Dr. Chester Ismay

Educator, Data Scientist, and R/Python Consultant

Creating a new column based on another

```
world_bank_data %>%
  mutate(prop_perc_college_complete = perc_college_complete / 100,
         .keep = "used")
```

```
# A tibble: 300 x 2
   perc_college_complete prop_perc_college_complete
                   <dbl>                      <dbl>
 1                  7.26                     0.0726
 2                 20.4                      0.204
 3                 18.0                      0.180
# ... with 297 more rows
```

Say hello to across()

```
world_bank_data %>%
  mutate(across(.cols = perc_college_complete,
                .fns = ~ .x / 100,
                .names = "prop_{.col}"),
         .keep = "used")
```

```
# A tibble: 300 x 2
   perc_college_complete prop_perc_college_complete
                   <dbl>                      <dbl>
 1                  7.26                     0.0726
 2                 20.4                      0.204
 3                 18.0                      0.180
# ... with 297 more rows
```

Computing across multiple columns

```
world_bank_with_prop <- world_bank_data %>%
  mutate(across(.cols = starts_with("perc"),
                .fns = ~ .x / 100,
                .names = "prop_{.col}"),
         .keep = "used")
```

```
glimpse(world_bank_with_prop)
```

```
Rows: 300
Columns: 8
$ perc_electric_access       <dbl> 100.000000, 100.000000, 100.000000, 100.000000...
$ perc_college_complete      <dbl> 7.25665, 20.35655, 18.04557, 7.57332, 7.69954, 8.94607...
$ perc_cvd_crd_70            <dbl> 15.8, 26.2, 28.1, 15.5, 15.0, 14.8, 14.0, 23.8, 25.6, 33.3, 14.2...
$ perc_rural_pop             <dbl> 45.601, 35.615, 30.834, 44.956, 44.334, 43.713, 43.093, 2.910....
$ prop_perc_electric_access  <dbl> 1.00000000, 1.00000000, 1.00000000, 1.00000000...
$ prop_perc_college_complete <dbl> 0.0725665, 0.2035655, 0.1804557, 0.0757332, 0.0769954...
$ prop_perc_cvd_crd_70       <dbl> 0.158, 0.262, 0.281, 0.155, 0.150, 0.148, 0.140, 0.238, 0.256...
$ prop_perc_rural_pop        <dbl> 0.45601, 0.35615, 0.30834, 0.44956, 0.44334, 0.43713, 0.43093...
```

Tweaking column names

- Change `prop_perc` to `prop`

```
names(world_bank_new_cols) <- sub(
  pattern = "prop_perc",
  replacement = "prop",
  x = names(world_bank_new_cols),
)
names(world_bank_new_cols)
```

```
[1] "perc_electric_access"  "perc_college_complete" "perc_cvd_crd_70"
[4] "perc_rural_pop"        "prop_electric_access"  "prop_college_complete"
[7] "prop_cvd_crd_70"       "prop_rural_pop"
```

across() with summarize()

```
world_bank_data %>%
  filter(year == 2015) %>%
  summarize(across(.cols = ends_with("rate"),
                   .fns = median,
                   .names = "median_{.col}"))
```

```
# A tibble: 1 x 3
  median_infant_mortality_rate median_fertility_rate median_unemployment_rate
                         <dbl>                  <dbl>                     <dbl>
1                          5.7                   1.87                      6.40
```

count() how many rows are in each combination

```
world_bank_data %>%
  count(country, continent)
```

```
# A tibble: 101 x 3
   country     continent     n
   <chr>       <fct>     <int>
 1 Albania     Europe        2
 2 Angola      Africa        1
 3 Armenia     Asia          3
 4 Australia   Oceania       4
 5 Austria     Europe        3
# ... with 96 more rows
```

count() with across() and introducing where()

```
world_bank_data %>%
  count(across(
    .cols = !where(is.numeric)
))
```

```
# A tibble: 101 x 5
   iso   country              continent region                  n
   <chr> <chr>                <fct>     <fct>                 <int>
 1 AGO   Angola               Africa    Middle Africa             1
 2 ALB   Albania              Europe    Southern Europe           2
 3 ARE   United Arab Emirates Asia      Western Asia              1
# ... with 98 more rows
```

Sorted result

```
world_bank_data %>%
  count(across(.cols = !where(is.numeric)),
        sort = TRUE)
```

```
# A tibble: 101 x 5
   iso   country    continent region                n
   <chr> <chr>      <fct>     <fct>                 <int>
 1 PRT   Portugal   Europe    Southern Europe          17
 2 BGR   Bulgaria   Europe    Eastern Europe           12
 3 SGP   Singapore  Asia      South-Eastern Asia       11
 4 COL   Colombia   Americas  South America            10
 5 ECU   Ecuador    Americas  South America            10
# ... with 96 more rows
```

Let's practice!

PROGRAMMING WITH DPLYR

Animal crossing: new rowwise's

PROGRAMMING WITH DPLYR

Dr. Chester Ismay

Educator, Data Scientist, and R/Python Consultant

Building up to rowwise()

- Identifying how many missing values are in a vector
  - `is.na()`
  - `sum()`

```
some_vector <- c(5, NA, 2, NA, 10)
is.na(some_vector)
```

```
[1] FALSE  TRUE FALSE  TRUE FALSE
```

```
sum(is.na(vec_test))
```

```
[1] 2
```

```
glimpse(world_bank_data)
```

```
Rows: 300
Columns: 12
$ iso                   <chr> "PRT", "ARM", "BGR", "PRT", "PRT", "PRT", "PRT",...
$ country               <chr> "Portugal", "Armenia", "Bulgaria", "Portugal", "...
$ continent             <fct> Europe, Asia, Europe, Europe, Europe, Europe, Eu...
$ region                <fct> Southern Europe, Western Asia, Eastern Europe, S...
$ year                  <dbl> 2000, 2001, 2001, 2001, 2002, 2003, 2004, 2004, ...
$ infant_mortality_rate <dbl> 5.5, 25.3, 17.1, 5.2, 4.7, 4.3, 4.0, 9.2, 17.2, ...
$ fertility_rate        <dbl> 1.47, 1.20, 1.20, 1.46, 1.45, 1.44, 1.43, 2.70, ...
$ perc_electric_access  <dbl> 100.00000, 100.00000, 100.00000, 100.00000, 100....
$ perc_college_complete <dbl> 7.25665, 20.35655, 18.04557, 7.57332, 7.69954, 8...
$ perc_cvd_crd_70       <dbl> 15.8, 26.2, 28.1, 15.5, 15.0, 14.8, 14.0, 23.8, ...
$ unemployment_rate     <dbl> 3.81, 10.91, 19.92, 3.83, 4.50, 6.13, 6.32, 0.71...
$ perc_rural_pop        <dbl> 45.601, 35.615, 30.834, 44.956, 44.334, 43.713, ...
```

rowwise() with c_across()

```
world_bank_data %>%
  rowwise() %>%
  mutate(num_missing = sum(is.na(
    c_across(infant_mortality_rate:last_col())))
  )) %>%
  select(country:year, num_missing) %>%
  arrange(desc(num_missing))
```

How many are missing?

```
# A tibble: 300 x 5
# Rowwise:
   country                continent region                       year num_missing
   <chr>                  <fct>     <fct>                       <dbl>       <int>
 1 Australia              Oceania   Australia and New Zealand  2016           2
 2 Austria                Europe    Western Europe             2016           2
 3 Azerbaijan             Asia      Western Asia               2016           2
 4 Bahrain                Asia      Western Asia               2016           2
 5 Bangladesh             Asia      Southern Asia              2016           2
# ... with 295 more rows
```

```
world_bank_data %>%
  filter(country == "Australia", year == 2016) %>%
  glimpse()
```

```
Rows: 1
Columns: 12
$ iso                  <chr> "AUS"
$ country              <chr> "Australia"
$ continent            <fct> Oceania
$ region               <fct> Australia and New Zealand
$ year                 <dbl> 2016
$ infant_mortality_rate <dbl> NA
$ fertility_rate       <dbl> NA
$ perc_electric_access  <dbl> 100
$ perc_college_complete <dbl> 30.02981
$ perc_cvd_crd_70      <dbl> 9
$ unemployment_rate    <dbl> 5.71
$ perc_rural_pop       <dbl> 14.2
```

if_any() with filter()

```
world_bank_data %>%
    filter(if_any(
      .cols = starts_with("perc"),
      .fns = ~ .x < 5)) %>%
    select(country, year, starts_with("perc"))
```

```
# A tibble: 60 x 6
   country    year perc_electric_access perc_college_complete perc_cvd_crd_70 perc_rural_pop
   <chr>     <dbl>                <dbl>                 <dbl>           <dbl>          <dbl>
 1 Qatar      2004                100                   20.9             23.8           2.91
 2 Pakistan   2005                 83.8                  3.92            33.3          66.0
 3 Singapore  2006                100                   19.6             12.5           0
 4 Honduras   2007                 73.5                  4.23            17.3          50.1
 5 Qatar      2007                100                   25.1             21.4           2.08
# ... with 55 more rows
```

if_all()

```
world_bank_data %>%
  filter(if_all(
    .cols = starts_with("perc"),
    .fns = ~ .x >= 25)) %>%
  select(country, year, starts_with("perc"))
```

```
# A tibble: 4 x 6
  country  year perc_electric_access perc_college_complete perc_cvd_crd_70 perc_rural_pop
  <chr>   <dbl>                <dbl>                 <dbl>           <dbl>          <dbl>
1 Russia   2010                  100                  59.3            30.9           26.3
2 Georgia  2012                  100                  30.2            25.3           43.7
3 Georgia  2014                  100                  30.9            25.1           42.9
4 Georgia  2016                  100                  32.8            26.5           42.2
```

Let's practice!

PROGRAMMING WITH DPLYR