NC STATE UNIVERSITY

Manipulating Data: Reshaping Data

Justin Post

Recap/Next Up!

- Data manipulation idea
- Documenting with Markdown
- Logical statements
- `dplyr`
- Creating new variables
    - Conditional execution (if then)
    - For loops
    - Vectorized functions
- Reshaping data

Reshaping Data

Long vs Wide format data

Reshaping Data

`tidyr` package

Easily allows for two very important actions

- `pivot_longer()` - lengthens data by increasing the number of rows and decreasing the number of columns
    - Most important as analysis methods often prefer this form
- `pivot_wider()` - widens data by increasing the number of columns and decreasing the number of rows

Reshaping Data

`tidyr` package

- Data in 'Wide' form

```
tempsData <- read_table2(file = "https://www4.stat.ncsu.edu/~online/datasets/cityTemps.txt")
tempsData
```

```
## # A tibble: 6 x 8
##   city        sun   mon   tue   wed   thr   fri   sat
##   <chr>     <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 atlanta      81    87    83    79    88    91    94
## 2 baltimore    73    75    70    78    73    75    79
## 3 charlotte    82    80    75    82    83    88    93
## 4 denver       72    71    67    68    72    71    58
## 5 ellington    51    42    47    52    55    56    59
## 6 frankfort    70    70    72    70    74    74    79
```

Manipulating Data

```
## # A tibble: 6 x 8
##   city        sun   mon   tue   wed   thr   fri   sat
##   <chr>     <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 atlanta      81    87    83    79    88    91    94
## 2 baltimore    73    75    70    78    73    75    79
## 3 charlotte    82    80    75    82    83    88    93
## 4 denver       72    71    67    68    72    71    58
## 5 ellington    51    42    47    52    55    56    59
## 6 frankfort    70    70    72    70    74    74    79
```

- Switch to 'Long' form with `pivot_longer()`
  - `cols` = columns to pivot to longer format (`cols = 2:8`)
  - `names_to` = new name(s) for columns created (`names_to = "day"`)
  - `values_to` = new name(s) for data values (`values_to = "temp"`)

Manipulating Data

- Switch to 'Long' form with `pivot_longer()`
  - `cols` = columns to pivot to longer format (`cols = 2:8`)
  - `names_to` = new name(s) for columns created (`names_to = "day"`)
  - `values_to` = new name(s) for data values (`values_to = "temp"`)

```
tempsData %>% pivot_longer(cols = 2:8, names_to = "day", values_to = "temp")
```

```
## # A tibble: 42 x 3
##   city    day    temp
##   <chr>   <chr> <dbl>
## 1 atlanta sun      81
## 2 atlanta mon      87
## 3 atlanta tue      83
## 4 atlanta wed      79
## 5 atlanta thr      88
## # … with 37 more rows
```

Reshaping Data

- Switch to 'Long' form with `pivot_longer()`
- Can provide columns in many ways!

```
newTempsData <- tempsData %>%
  pivot_longer(cols = sun:sat, names_to = "day", values_to = "temp")
newTempsData
```

```
## # A tibble: 42 x 3
##   city    day    temp
##   <chr>   <chr> <dbl>
## 1 atlanta sun      81
## 2 atlanta mon      87
## 3 atlanta tue      83
## 4 atlanta wed      79
## 5 atlanta thr      88
## # … with 37 more rows
```

Reshaping Data

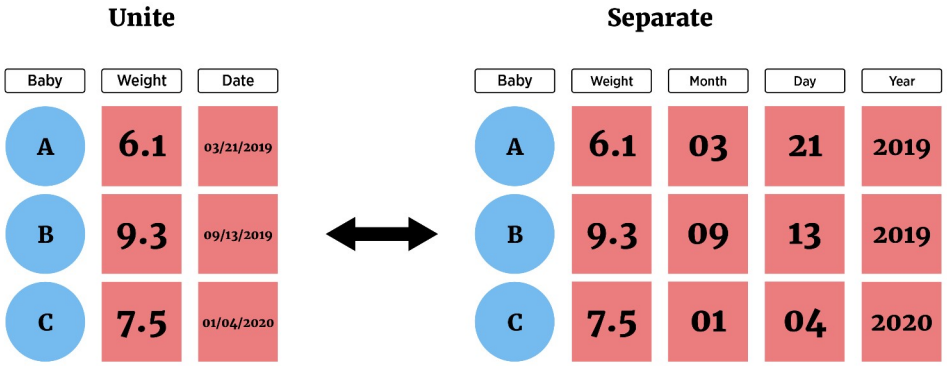
- Switch to 'Wide' form with `pivot_wider()`
  - `names_from` = column(s) to get the names used in the output columns (`names_from = "day"`)
  - `values_from` = column(s) to get the cell values from (`values_from = "temp"`)

```
newTempsData %>% pivot_wider(names_from = "day", values_from = "temp")
```

```
## # A tibble: 6 x 8
##   city        sun   mon   tue   wed   thr   fri   sat
##   <chr>     <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 atlanta      81    87    83    79    88    91    94
## 2 baltimore    73    75    70    78    73    75    79
## 3 charlotte    82    80    75    82    83    88    93
## 4 denver       72    71    67    68    72    71    58
## 5 ellington    51    42    47    52    55    56    59
## 6 frankfort    70    70    72    70    74    74    79
```

Reshaping Data

- Separate a column (or combine two columns) using `separate()` and `unite()`



Reshaping Data

- Separate a column (or combine two columns) using `separate()` and `unite()`
- Consider data set on air pollution in Chicago

```
chicagoData <- read_csv("https://www4.stat.ncsu.edu/~online/datasets/Chicago.csv")
chicagoData
```

```
## # A tibble: 1,461 x 11
##       X city  date     death  temp dewpoint  pm10    o3  time season  year
##   <dbl> <chr> <chr>    <dbl> <dbl>    <dbl> <dbl> <dbl> <dbl> <chr>  <dbl>
## 1  3654 chic  1/1/1997   137  36       37.5  13.1  5.66  3654 winter  1997
## 2  3655 chic  1/2/1997   123  45       47.2  41.9  5.53  3655 winter  1997
## 3  3656 chic  1/3/1997   127  40       38    27.0  6.29  3656 winter  1997
## 4  3657 chic  1/4/1997   146  51.5     45.5  25.1  7.54  3657 winter  1997
## 5  3658 chic  1/5/1997   102  27       11.2  15.3 20.8   3658 winter  1997
## # … with 1,456 more rows
```

Manipulating Data

- Can parse with `separate`:

```
chicagoData %>% separate(date, c("Month", "Day", "Year"), sep = "/",
                                                  convert = TRUE, remove = FALSE)
```

```
## # A tibble: 1,461 x 14
##       X city  date  Month   Day  Year death  temp dewpoint  pm10    o3  time
##   <dbl> <chr> <chr> <int> <int> <int> <dbl> <dbl>    <dbl> <dbl> <dbl> <dbl>
## 1  3654 chic  1/1/…     1     1  1997   137  36        37.5  13.1  5.66  3654
## 2  3655 chic  1/2/…     1     2  1997   123  45        47.2  41.9  5.53  3655
## 3  3656 chic  1/3/…     1     3  1997   127  40        38    27.0  6.29  3656
## 4  3657 chic  1/4/…     1     4  1997   146  51.5      45.5  25.1  7.54  3657
## 5  3658 chic  1/5/…     1     5  1997   102  27        11.2  15.3 20.8   3658
## # … with 1,456 more rows, and 2 more variables: season <chr>, year <dbl>
```

Manipulating Data

- Can combine with `unite`:

```
chicagoData %>% separate(date, c("Month", "Day", "Year"), sep = "/",
                                                    convert = TRUE, remove = FALSE) %>%
  unite(MonthDay, Month, Day, sep = "-")
```

```
## # A tibble: 1,461 x 13
##       X city  date  MonthDay  Year death  temp dewpoint  pm10    o3  time season
##   <dbl> <chr> <chr> <chr>    <int> <dbl> <dbl>    <dbl> <dbl> <dbl> <dbl> <chr>
## 1  3654 chic  1/1/… 1-1       1997   137  36       37.5  13.1  5.66  3654 winter
## 2  3655 chic  1/2/… 1-2       1997   123  45       47.2  41.9  5.53  3655 winter
## 3  3656 chic  1/3/… 1-3       1997   127  40       38    27.0  6.29  3656 winter
## 4  3657 chic  1/4/… 1-4       1997   146  51.5     45.5  25.1  7.54  3657 winter
## 5  3658 chic  1/5/… 1-5       1997   102  27       11.2  15.3 20.8   3658 winter
## # … with 1,456 more rows, and 1 more variable: year <dbl>
```

Recap!

- Data manipulation idea
- Documenting with Markdown
- Logical statements
- `dplyr`
- Creating new variables
    - Conditional execution (if then)
    - For loops
    - Vectorized functions
- Reshaping data