Data Transformations and Queries

Using R

Dr. Austin Brown

Kennesaw State University

Simple Data Transformations/Queries

- **Objective**: Learn to perform basic data transformations such as selecting columns, filtering rows, and creating new columns.
- **Importance**: Essential for cleaning and preparing data for analysis.
- **Key Points**
  - **select() Function**: From the dplyr package, used to select specific columns.
  - **Syntax**: `select(data, column_names)`
  - **Example**: Code snippet showing how to select specific columns.

Simple Data Transformations/Queries

- In many instances, you may need to perform simple data transformations or queries to extract specific information from your dataset.
- This could involve selecting specific columns, filtering rows based on certain conditions, or creating new columns based on existing data.
- In R, the `dplyr` package provides a set of functions that make these operations easy and intuitive.

Selecting Columns in R with `dplyr`

- In D2L, download the HEART.csv file and upload to your RStudio Cloud project folder.
- Go ahead and import the data using `read.csv` as we have done already.
- Now, suppose instead of working with the full dataframe, I want to only focus on a few specific columns:
    - `Chol_Status`
    - `BP_Status`
    - `Weight_Status`
    - `Smoking_Status`

Selecting Columns in R with `dplyr`

- While there are multiple ways of creating a new dataframe which contains only these four columns, one of the most straightforward ways is to use the `select()` function from the `dplyr` package.
- The `select()` function allows you to choose specific columns from a dataframe and create a new dataframe with only those columns.
- This can be useful when you have a large dataset with many columns, but you are only interested in a subset of them.
- Let's see how this works with the HEART dataset.

Selecting Columns in R with `dplyr`

```r
## Read in the HEART.csv dataset ##
heart <- read.csv("HEART.csv")
## Install the dplyr package if you haven't already ##
install.packages('dplyr')
## Load the dplyr package ##
library(dplyr)
## Select our specific columns ##
selected_columns <- select(heart,
                           Chol_Status,
                           BP_Status,
                           Weight_Status,
                           Smoking_Status)
```

Selecting Columns in R with `dplyr`

- As we can see in the code snippet above, we first read in the HEART.csv dataset using the `read.csv()` function.
- Next, we load the `dplyr` package using the `library()` function.
- Finally, we use the `select()` function to create a new dataframe called `selected_columns` from the existing `heart` dataframe that contains only the columns `Chol_Status`, `BP_Status`, `Weight_Status`, and `Smoking_Status`.

Filtering Rows in R with `dplyr`

- Not only can we select columns, but we can also filter rows based on specific conditions.
- For example, in the HEART dataset, we may want to filter out all rows where the `Chol_Status` is `High`.
  - That is, we want to keep only the rows where `Chol_Status` is not `High`.
- To do this, we can use the `filter()` function from the `dplyr` package.

Filtering Rows in R with `dplyr`

- The `filter()` function allows you to filter rows based on specific conditions.
- The basic syntax is `filter(data, condition)`, where:
  - `data` is the dataframe you are working with.
  - `condition` is the condition you want to filter on.
- Let's try out our filtering operation on the HEART dataset.

Filtering Rows in R with `dplyr`

```r
## Filter out rows where Chol_Status is High ##
filtered_rows <- filter(heart,
                        Chol_Status != "High")
```

Filtering Rows in R with `dplyr`

- Note, in the code snippet above, we use the `filter()` function to create a new dataframe called `filtered_rows` from the existing `heart` dataframe.
- We specify the condition `Chol_Status != "High"` to filter out all rows where the `Chol_Status` is `High`.
- We use the `!=` operator to indicate "not equal to".

Creating New Columns in R with `dplyr`

- Many times, you may need to create new columns based on existing data in your dataset.
- For example, in the HEART dataset, we may want to create a new column called `BMI` that calculates the Body Mass Index for each individual.
- To do this, we can use the `mutate()` function from the `dplyr` package.

Creating New Columns in R with `dplyr`

- The `mutate()` function allows you to create new columns based on existing columns in your dataframe.
- The basic syntax is `mutate(data, new_column = expression)`, where:
  - `data` is the dataframe you are working with.
  - `new_column` is the name of the new column you want to create.
  - `expression` is the calculation or transformation you want to apply to create the new column.
- Let's create a new column called `BMI` in the HEART dataset.

Creating New Columns in R with dplyr

```r
## Add BMI Column to heart ##
new_column <- mutate(heart,
                     BMI = (Weight / (Height)^2)*703)
## Use str to verify ##
str(new_column)
```

```
'data.frame':   5209 obs. of  18 variables:
 $ Status        : chr  "Dead" "Dead" "Alive" "Alive" ...
 $ DeathCause    : chr  "Other" "Cancer" "" "" ...
 $ AgeCHDdiag    : int  NA NA NA NA NA NA NA NA NA NA ...
 $ Sex           : chr  "Female" "Female" "Female" "Female" ...
 $ AgeAtStart    : int  29 41 57 39 42 58 36 53 35 52 ...
 $ Height        : num  62.5 59.8 62.2 65.8 66 ...
 $ Weight        : int  140 194 132 158 156 131 136 130 194 129 ...
 $ Diastolic     : int  78 92 90 80 76 92 80 80 68 78 ...
 $ Systolic      : int  124 144 170 128 110 176 112 114 132 124 ...
 $ MRW           : int  121 183 114 123 116 117 110 99 124 106 ...
 $ Smoking       : int  0 0 10 0 20 0 15 0 0 5 ...
 $ AgeAtDeath    : int  55 57 NA NA NA NA NA 77 NA 82 ...
 $ Cholesterol   : int  NA 181 250 242 281 196 196 276 211 284 ...
 $ Chol_Status   : chr  "" "Desirable" "High" "High" ...
 $ BP_Status     : chr  "Normal" "High" "High" "Normal" ...
 $ Weight_Status : chr  "Overweight" "Overweight" "Overweight" "Overweight" ...
 $ Smoking_Status: chr  "Non-smoker" "Non-smoker" "Moderate (6-15)" "Non-smoker
 $ BMI           : num  25.2 38.2 23.9 25.7 25.2 ...
```