Data Wrangling R

Art Tay

Loading Packages

```
# Option 1: Individual Packages (not recommended).
library(dplyr)

# Option 2: tidyverse (okay).
library(tidyverse)

# Option 3: tidymodels (preferred).
library(tidymodels)
tidymodels_prefer() # Can be used to avoid conflicts with other packages.
```

- Tidyverse Documentation
- Tidymodels Documentation
- Data Science in R (Free Textbook)

Pipes

- Pipes are a cleaner way to preform data operations.
- New in `R >=4.1` "`|>`" is a native pipe operator.
- `%>%` comes from tidyverse and can also be used.
- **Syntax:** object pipe function(args) <-> function(object, args)
- **Chaining:** Data |> function1(args) |> function2(args) = function2(function1(data, args1), args2)

Examples:

```
# Simple example.
a <- 5
b <- a |> sum(2)
b
```

```
[1] 7
```

```
c <- sum(a, 2)
c
```

```
[1] 7
```

```r
# Pipes work with all named functions.
t_test <- rnorm(n = 10, mean = 1, sd = 1) |> t.test()
t_test
```

```
	One Sample t-test

data:  rnorm(n = 10, mean = 1, sd = 1)
t = 3.5964, df = 9, p-value = 0.005781
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 0.4220256 1.8530964
sample estimates:
mean of x 
 1.137561 
```

```r
# You can use anonymous functions.
d <- 5 |> {\(x) x * 7}()
d
```

```
[1] 35
```

Practice:

- Generate 10 samples from a normal distribution, add 1, then test if the mean is different from 2.

```r
t_test_practice <- (rnorm(n = 10) + 1) |> t.test(mu = 2)

t_test_practice_2 <- rnorm(n = 10) |> {\(x) x + 1}() |> t.test(mu = 2)
```

Quick Look at the Data

```r
data <- read.csv("./Cholesterol_R.csv")
data <- data |> rename("ID" = contains("ID"))

head(data)
```

```
  ID Before After4weeks After8weeks Margarine
1  1   6.42        5.83        5.75         B
2  2   6.76        6.20        6.13         A
3  3   6.56        5.83        5.71         B
4  4   4.80        4.27        4.15         A
5  5   8.43        7.71        7.67         B
6  6   7.49        7.12        7.05         A
```

```r
str(data)
```

```
'data.frame':	18 obs. of  5 variables:
 $ ID         : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Before     : num  6.42 6.76 6.56 4.8 8.43 7.49 8.05 5.05 5.77 3.91 ...
 $ After4weeks: num  5.83 6.2 5.83 4.27 7.71 7.12 7.25 4.63 5.31 3.7 ...
 $ After8weeks: num  5.75 6.13 5.71 4.15 7.67 7.05 7.1 4.67 5.33 3.66 ...
 $ Margarine  : chr  "B" "A" "B" "A" ...
```

Subseting

Examples:

```
# Selecting columns by name.
measurements <- data |> select(Before, After4weeks, After8weeks)
head(measurements)
```

```
  Before After4weeks After8weeks
1   6.42        5.83        5.75
2   6.76        6.20        6.13
3   6.56        5.83        5.71
4   4.80        4.27        4.15
5   8.43        7.71        7.67
6   7.49        7.12        7.05
```

```
# Selecting columns based on a pattern.
after <- data |> select(starts_with("After"))
head(after)
```

```
  After4weeks After8weeks
1        5.83        5.75
2        6.20        6.13
3        5.83        5.71
4        4.27        4.15
5        7.71        7.67
6        7.12        7.05
```

```
# Lots of options.
tidyselect::starts_with()
tidyselect::ends_with()
tidyselect::contains()
tidyselect::matches()
tidyselect::num_range()
tidyselect::everything()
tidyselect::one_of()
tidyselect::all_of()
tidyselect::any_of()
```

```
# Slicing rows.
first3 <- data |> slice(1:3)
first3
```

```
  ID Before After4weeks After8weeks Margarine
1  1   6.42        5.83        5.75         B
2  2   6.76        6.20        6.13         A
3  3   6.56        5.83        5.71         B
```

```
# Filter rows based on a condition.
A_above_6 <- data |> filter(After8weeks >= 6 & Margarine == "A")
head(A_above_6)
```

```
  ID Before After4weeks After8weeks Margarine
```

```
1  2   6.76        6.20        6.13          A
2  6   7.49        7.12        7.05          A
3 14   7.67        7.11        6.96          A
4 15   7.34        6.84        6.82          A
```

Practice

- Find the patient IDs with baseline (before) measurements below 5.

```
data |> filter(Before < 5) |> select(ID)
```

```
  ID
1  4
2 10
```

Sorting

Examples

```
# Sort the data based on a single column.
data_sort_baseline <- data |> arrange(desc(Before))
head(data_sort_baseline)
```

```
  ID Before After4weeks After8weeks Margarine
1  5   8.43        7.71        7.67         B
2  7   8.05        7.25        7.10         B
3 14   7.67        7.11        6.96         A
4  6   7.49        7.12        7.05         A
5 15   7.34        6.84        6.82         A
6 16   6.85        6.40        6.29         B
```

```
# Sort thr data based on multiple columns.
data_sort_many <- data |> arrange(Before, Margarine)
head(data_sort_many)
```

```
  ID Before After4weeks After8weeks Margarine
1 10   3.91        3.70        3.66         A
2  4   4.80        4.27        4.15         A
3  8   5.05        4.63        4.67         A
4 17   5.13        4.52        4.45         A
5 18   5.73        5.13        5.17         B
6  9   5.77        5.31        5.33         B
```

Practice

- Sort Margarine in alphabetical order, then the IDs in ascending order.

```
data_sort_practice <- data |> arrange(Margarine, ID)
data_sort_practice
```

```
   ID Before After4weeks After8weeks Margarine
1   2   6.76        6.20        6.13         A
2   4   4.80        4.27        4.15         A
```

```
3   6   7.49        7.12        7.05         A
4   8   5.05        4.63        4.67         A
5   10  3.91        3.70        3.66         A
6   13  6.17        5.56        5.51         A
7   14  7.67        7.11        6.96         A
8   15  7.34        6.84        6.82         A
9   17  5.13        4.52        4.45         A
10  1   6.42        5.83        5.75         B
11  3   6.56        5.83        5.71         B
12  5   8.43        7.71        7.67         B
13  7   8.05        7.25        7.10         B
14  9   5.77        5.31        5.33         B
15 11   6.77        6.15        5.96         B
16 12   6.44        5.59        5.64         B
17 16   6.85        6.40        6.29         B
18 18   5.73        5.13        5.17         B
```

Summarizing

Example

```
# Creating a table of summary statistics.
summary_table <- data |> select(-ID) |> group_by(Margarine) |>
    summarize(
        across(where(is.numeric),
               list(mean = mean, sd = sd)
        )
    )

summary_table
```

```
# A tibble: 2 x 7
  Margarine Before_mean Before_sd After4weeks_mean After4weeks_sd
  <chr>           <dbl>     <dbl>            <dbl>          <dbl>
1 A                6.04     1.36              5.55          1.32
2 B                6.78     0.919             6.13          0.864
# i 2 more variables: After8weeks_mean <dbl>, After8weeks_sd <dbl>
```

Practice

- Find the median for each column for patients with even ID numbers.

```
summary_table_practice <- data |> filter(ID %% 2 == 0) |> select(-ID) |>
    summarize(
        across(where(is.numeric),
            list(median = median)
        )
    )
```

Feature Engineering

Examples

```
# Create simple features.
data <- data |> mutate(
    diff_4wk = After4weeks - Before,
    diff_8wk = After8weeks - Before
)

head(data)
```

```
  ID Before After4weeks After8weeks Margarine diff_4wk diff_8wk
1  1   6.42        5.83        5.75         B    -0.59    -0.67
2  2   6.76        6.20        6.13         A    -0.56    -0.63
3  3   6.56        5.83        5.71         B    -0.73    -0.85
4  4   4.80        4.27        4.15         A    -0.53    -0.65
5  5   8.43        7.71        7.67         B    -0.72    -0.76
6  6   7.49        7.12        7.05         A    -0.37    -0.44
```

```
# Function based example: normalization.
norm_func <- function(x){
    x <- x - mean(x)
    return(x / sd(x))
}

normalized <- data |> select(Before) |>
    mutate(
        across(where(is.numeric), list(norm = norm_func, log = log))
    )

head(normalized)
```

```
  Before Before_norm Before_log
1   6.42   0.0102614   1.859418
2   6.76   0.2957149   1.911023
3   6.56   0.1278011   1.880991
4   4.80  -1.3498404   1.568616
5   8.43   1.6977952   2.131797
6   7.49   0.9086003   2.013569
```

Practice

- Find the percentage change between baseline and 8 weeks.

```
data_percent_change <- data |> mutate(percent_change = After8weeks / Before - 1)
```

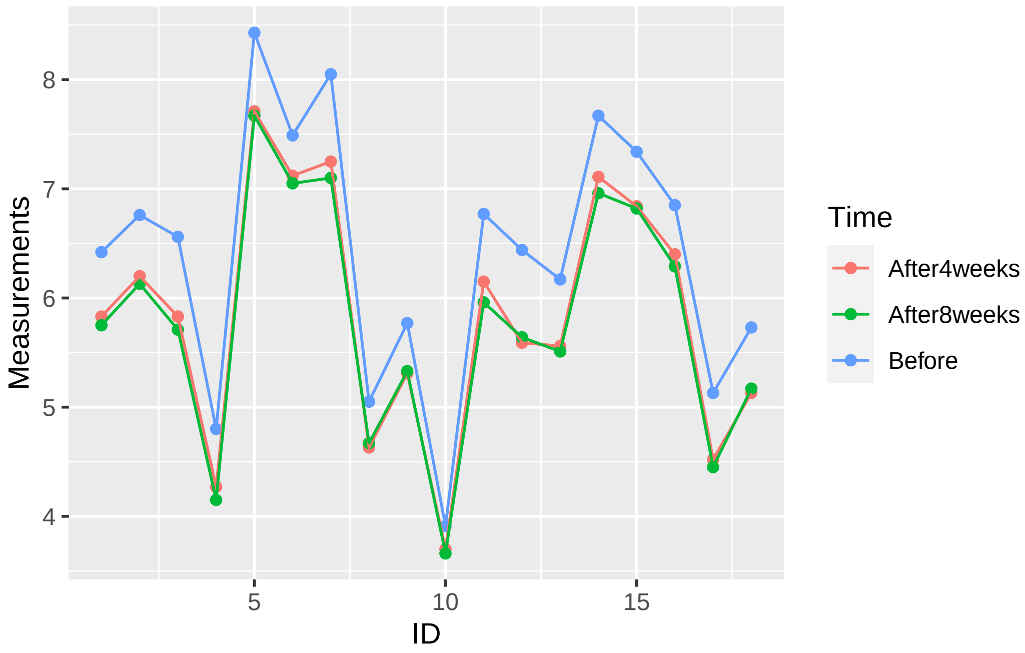
Pivoting

Examples

```
# Pivot to long format. (Good for plotting).
data_long <- data |>
    pivot_longer(cols = c(Before, starts_with("After")),
                 names_to = "Time", values_to = "Measurements")
head(data_long)
```

```
# A tibble: 6 x 6
     ID Margarine diff_4wk diff_8wk Time        Measurements
  <int> <chr>        <dbl>    <dbl> <chr>              <dbl>
1     1 B           -0.59     -0.67 Before              6.42
2     1 B           -0.59     -0.67 After4weeks         5.83
3     1 B           -0.59     -0.67 After8weeks         5.75
4     2 A           -0.560    -0.63 Before              6.76
5     2 A           -0.560    -0.63 After4weeks         6.2
6     2 A           -0.560    -0.63 After8weeks         6.13
```

```
# Why?
data_long |> ggplot(aes(x = ID, y = Measurements, color = Time)) +
               geom_point() + geom_line()
```



```
# Pivot to wide format.
data_wide <- data_long |> pivot_wider(names_from = Time,
                                      values_from = Measurements)
head(data_wide)
```

```
# A tibble: 6 x 7
```

```
     ID Margarine diff_4wk diff_8wk Before After4weeks After8weeks
  <int> <chr>        <dbl>    <dbl>  <dbl>       <dbl>       <dbl>
1     1 B            -0.59    -0.67   6.42        5.83        5.75
2     2 A            -0.560   -0.63   6.76        6.2         6.13
3     3 B            -0.730   -0.85   6.56        5.83        5.71
4     4 A            -0.53    -0.650  4.8         4.27        4.15
5     5 B            -0.72    -0.76   8.43        7.71        7.67
6     6 A            -0.37    -0.440  7.49        7.12        7.05
```

```r
# Why?
t.test(data_wide$Before, data_wide$After8weeks)
```

```
    Welch Two Sample t-test

data:  data_wide$Before and data_wide$After8weeks
t = 1.6443, df = 33.796, p-value = 0.1094
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.1485269  1.4063046
sample estimates:
mean of x mean of y
 6.407778  5.778889
```

Practice

- Create a bar plot with time groups on the x axis and the average 8 week difference on the y axis.

```r
plot_practice <- data |>
    pivot_longer(cols = c(Before, starts_with("After")),
                 names_to = "Time", values_to = "Measurements") |>
    group_by(Time) |>
    summarise(avg_8_diff = mean(Measurements)) |>
    ggplot(aes(x = Time, y = avg_8_diff)) +
    geom_bar(stat = "identity")
```

Joins

Examples

```r
# Include additional information.
info_A <- data |> select(ID, Margarine) |> filter(Margarine == "A") |>
    mutate(After12weeks = rnorm(length(ID), mean = 3, sd = 1))
head(info_A)
```

```
  ID Margarine After12weeks
1  2         A     2.026017
2  4         A     2.928692
3  6         A     1.906760
4  8         A     2.139485
5 10         A     2.237223
6 13         A     2.745113
```

```
# Left join - maintains all the information on the left table.
data_left <- data |> left_join(info_A, join_by(ID, Margarine))
head(data_left)
```

```
  ID Before After4weeks After8weeks Margarine diff_4wk diff_8wk After12weeks
1  1   6.42        5.83        5.75         B    -0.59    -0.67           NA
2  2   6.76        6.20        6.13         A    -0.56    -0.63     2.026017
3  3   6.56        5.83        5.71         B    -0.73    -0.85           NA
4  4   4.80        4.27        4.15         A    -0.53    -0.65     2.928692
5  5   8.43        7.71        7.67         B    -0.72    -0.76           NA
6  6   7.49        7.12        7.05         A    -0.37    -0.44     1.906760
```

```
# Right join - maintains all the information on the right table.
data_right <- data |> right_join(info_A, join_by(ID, Margarine))
head(data_right)
```

```
  ID Before After4weeks After8weeks Margarine diff_4wk diff_8wk After12weeks
1  2   6.76        6.20        6.13         A    -0.56    -0.63     2.026017
2  4   4.80        4.27        4.15         A    -0.53    -0.65     2.928692
3  6   7.49        7.12        7.05         A    -0.37    -0.44     1.906760
4  8   5.05        4.63        4.67         A    -0.42    -0.38     2.139485
5 10   3.91        3.70        3.66         A    -0.21    -0.25     2.237223
6 13   6.17        5.56        5.51         A    -0.61    -0.66     2.745113
```

```
# Inner v. Full Joins
full_ex_1 <- data.frame(ID = 1:10, value_1 = rnorm(10))
full_ex_2 <- data.frame(ID = 6:15, value_2 = rnorm(10))

full_ex_1 |> left_join(full_ex_2, join_by(ID))
```

```
   ID     value_1     value_2
1   1 -0.33878578          NA
2   2 -0.47538017          NA
3   3 -0.28835249          NA
4   4  2.33931073          NA
5   5 -1.36536669          NA
6   6  0.49142987 -0.05060684
7   7  0.47420285  0.36681646
8   8 -0.57437554  0.54348327
9   9  0.73461252  0.08616754
10 10  0.03344151  0.65897144
```

```
full_ex_1 |> inner_join(full_ex_2, join_by(ID))
```

```
  ID     value_1     value_2
1  6  0.49142987 -0.05060684
2  7  0.47420285  0.36681646
3  8 -0.57437554  0.54348327
4  9  0.73461252  0.08616754
5 10  0.03344151  0.65897144
```

```
full_ex_1 |> full_join(full_ex_2, join_by(ID))
```

```
   ID     value_1     value_2
1   1 -0.33878578          NA
2   2 -0.47538017          NA
3   3 -0.28835249          NA
4   4  2.33931073          NA
5   5 -1.36536669          NA
6   6  0.49142987 -0.05060684
7   7  0.47420285  0.36681646
8   8 -0.57437554  0.54348327
9   9  0.73461252  0.08616754
10 10  0.03344151  0.65897144
11 11          NA  0.01532030
12 12          NA  0.46015989
13 13          NA -0.24378432
14 14          NA -0.89058840
15 15          NA  1.69796996
```

```
# Additional patients to include.
data_2 <- data |> select(-c(ID, starts_with("diff"))) |>
                  mutate(Margarine = ifelse(Margarine == "A", "C", "D")) |>
                  mutate(across(where(is.numeric),
                                function(x){x + rnorm(length(x))}
                         )
                  ) |>
                  mutate(ID = 19:36)

data_combined <- data |> bind_rows(data_2)
head(data_combined)
```

```
  ID Before After4weeks After8weeks Margarine diff_4wk diff_8wk
1  1   6.42        5.83        5.75         B    -0.59    -0.67
2  2   6.76        6.20        6.13         A    -0.56    -0.63
3  3   6.56        5.83        5.71         B    -0.73    -0.85
4  4   4.80        4.27        4.15         A    -0.53    -0.65
5  5   8.43        7.71        7.67         B    -0.72    -0.76
6  6   7.49        7.12        7.05         A    -0.37    -0.44
```

Practice

- Include the following information retaining all patients present in the original dataset measurement. Complete the dataset by replacing any missing (NA) values with the mean for the patients corresponding group (Margarine).

```
data_to_include <- data.frame(
    ID = 6:20,
    Weight = rnorm(n = 15, 100, 10)
)
```

```
data_practice <- data |> left_join(data_to_include) |>
    group_by(Margarine) |>
    mutate(Weight = ifelse(is.na(Weight), mean(Weight, na.rm = T), Weight))
```

Extensions for Modeling

Resampling

```
# Create a 80-20 train/test split.
set.seed(123)
split <- initial_split(data_combined, prop = 0.8) # contains indices.

data_train <- training(split)
data_test <- testing(split)
```

```
# Bootstrap t-tests.
boots <- data_train |> select(ID, Before) |> bootstraps(times = 10)
head(as.data.frame(boots$splits[[1]]))
```

```
  ID   Before
1  5 8.430000
2  5 8.430000
3 20 6.761933
4  4 4.800000
5 24 6.308628
6 12 6.440000
```

```
p_values <- sapply(boots$splits,
    FUN = function(x) {
        p_value <- x |> as.data.frame() |> t.test()
        p_value$p.value
    }
)
p_values
```

```
 [1] 1.990771e-14 2.714639e-14 9.917317e-16 6.688158e-13 9.027094e-14
 [6] 1.217726e-14 6.462299e-14 3.062132e-14 3.848936e-15 2.807957e-13
```

Recipes

- Recipes are a way to bundle data pre-processing (possible trained).
- Easier access to more complex transformations (ex: PCA, Splines, Imputation).
- Avoids data leakage, and is online for incoming data.
- Works well with the resampling and modeling framework.

```
# Recipe work based on variable roles. This makes it easier to preform
# grouped transformations.
recipe_ex <- data_train |>
                recipe( diff_8wk ~ .) |>
                    # Don't use ID for modeling, but keep it in the dataset.
                    update_role(ID, new_role = "ID")

recipe_ex
```

```
# Most dyplr function have an analog in recipes.
recipe_ex <- recipe_ex |>
                step_select(c(diff_8wk, Margarine, Before)) |>
                step_mutate(Margarine = as.factor(Margarine))

recipe_ex
```

```
# Let's take a look at what our cleaned dataset would look like here.
data_train_cleaned <- bake(prep(recipe_ex), new_data = NULL)
head(data_train_cleaned)
```

```
# A tibble: 6 x 3
  diff_8wk Margarine Before
     <dbl> <fct>      <dbl>
1   NA     C           7.45
2   -0.520 A           7.34
3   -0.71  A           7.67
4   -0.85  B           6.56
5   -0.25  A           3.91
6   -0.560 B           5.73
```

```
# Why do we need prep? For learned transformations.
recipe_ex <- recipe_ex |>
               # Imputation - handles missing data.
               step_impute_knn(diff_8wk, neighbors = 1)

recipe_ex |> prep()
```

```
# Grouped transformations.
recipe_ex <- recipe_ex |>
               # Normalize - there is actually a step_normalize.
               step_mutate_at(all_numeric_predictors(), fn = norm_func) |>
               # Create dummy variable.
               step_dummy(all_nominal_predictors()) |>
               prep()
recipe_ex
```

```
data_train_cleaned <- bake(recipe_ex, new_data = NULL)
data_train_cleaned
```

```
# A tibble: 28 x 5
   diff_8wk Before Margarine_B Margarine_C Margarine_D
      <dbl>  <dbl>       <dbl>       <dbl>       <dbl>
 1   -0.520  0.768           0           1           0
 2   -0.520  0.693           0           0           0
 3   -0.71   0.922           0           0           0
 4   -0.85   0.151           1           0           0
 5   -0.25  -1.69            0           0           0
 6   -0.560 -0.425           1           0           0
 7   -0.25  -1.39            0           1           0
 8   -0.810  0.297           1           0           0
 9   -0.76   1.45             1           0           0
10   -0.810  0.291            0           1           0
# i 18 more rows
```

```
data_test_cleaned <- bake(recipe_ex, new_data = data_test)
data_test_cleaned
```

```
# A tibble: 8 x 5
  diff_8wk  Before Margarine_B Margarine_C Margarine_D
     <dbl>   <dbl>       <dbl>       <dbl>       <dbl>
1   -0.63   0.260            0           0           0
2   -0.440  1.10             0           0           0
3   -0.66  -0.420            0           0           0
4   -0.560  0.364            1           0           0
5   -0.68  -1.62             0           0           0
6   -0.71   1.29             0           0           1
7   -0.560 -1.05             0           1           0
8   -0.85   0.0714           0           0           1
```