DataManipulation

Anita Dhillon

2023-03-07

The classic way of running code

For example, I want the square root of the mean of a sequence of numbers

Nested Code

```
numbers <- 1:300
mean(numbers)
```

```
## [1] 150.5
```

```
sqrt(mean(numbers))
```

```
## [1] 12.26784
```

Sequential Code

In this case we create intermediate variables

```
numbers <- 300:546
numbers <- 1:300
 numbers_mean <- mean(numbers)
 sqrt(x = numbers_mean)
```

```
## [1] 12.26784
```

Piping Code

It can be implemented in R using the package `magrittr` It is a dependency of `dplye`, so it is installed along

```
 library(magrittr)
```

The original symbol of the pipe is `%>%` But we also have a new symbol that is similiar to bash `l>` The purpose of pipes is to reduce the max need of intermediate variables for the mean example

```
1:300 %>% mean() %>% sqrt()
```

```
## [1] 12.26784
```

Pipes with the surveys data set

```
surveys <- read.csv(file = "../data-raw/surveys.csv")
str(surveys)
```

```
## 'data.frame':    35549 obs. of  9 variables:
##  $ record_id      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ month          : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ day            : int  16 16 16 16 16 16 16 16 16 16 ...
##  $ year           : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ plot_id        : int  2 3 2 7 3 1 2 1 1 6 ...
##  $ species_id     : chr  "NL" "NL" "DM" "DM" ...
##  $ sex            : chr  "M" "M" "F" "M" ...
##  $ hindfoot_length: int  32 33 37 36 35 14 NA 37 34 20 ...
##  $ weight         : int  NA NA NA NA NA NA NA NA NA NA ...
```

Calculate the mean of the year column using pipes

```
surveys$year %>% mean()
```

```
## [1] 1990.475
```

Calculate the mean of the weight column

```
surveys$weight %>% mean(na.rm = TRUE)
```

```
## [1] 42.67243
```

Excercise 1

Load surveys.csv into R using read.csv().

```
surveys <- read.csv(file = "../data-raw/surveys.csv")
str(surveys)
```

```
## 'data.frame':    35549 obs. of  9 variables:
##  $ record_id      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ month          : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ day            : int  16 16 16 16 16 16 16 16 16 16 ...
##  $ year           : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ plot_id        : int  2 3 2 7 3 1 2 1 1 6 ...
##  $ species_id     : chr  "NL" "NL" "DM" "DM" ...
##  $ sex            : chr  "M" "M" "F" "M" ...
##  $ hindfoot_length: int  32 33 37 36 35 14 NA 37 34 20 ...
##  $ weight         : int  NA NA NA NA NA NA NA NA NA NA ...
```

Use select() to create a new data frame object called surveys1 with just the year, month, day, and species_id columns in that order.

```
surveys1 <- select(surveys, year, month, day, species_id)
str(surveys1)
```

```
## 'data.frame':    35549 obs. of  4 variables:
##  $ year      : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ month     : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ day       : int  16 16 16 16 16 16 16 16 16 16 ...
##  $ species_id: chr  "NL" "NL" "DM" "DM" ...
```

Create a new data frame called surveys2 with the year, species_id, and weight in kilograms of each individual, with no null weights. Use mutate(), select(), and filter() with !is.na(). The weight in the table is given in grams so you will need to create a new column called "weight_kg" for weight in kilograms by dividing the weight column by 1000.

```
surveys2 <- select(surveys, year ,species_id, weight)
str(surveys2)
```

```
## 'data.frame':    35549 obs. of  3 variables:
##  $ year      : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ species_id: chr  "NL" "NL" "DM" "DM" ...
##  $ weight    : int  NA NA NA NA NA NA NA NA NA NA ...
```

```
surveys2 <- mutate(surveys2, weight_kg = weight/1000)
str(surveys2)
```

```
## 'data.frame':    35549 obs. of  4 variables:
##  $ year      : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ species_id: chr  "NL" "NL" "DM" "DM" ...
##  $ weight    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ weight_kg : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
surveys2 <- filter(surveys2, !is.na(weight_kg))
str(surveys2)
```

```
## 'data.frame':    32283 obs. of  4 variables:
##  $ year      : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ species_id: chr  "DM" "DM" "DM" "DM" ...
##  $ weight    : int  40 48 29 46 36 52 8 22 35 7 ...
##  $ weight_kg : num  0.04 0.048 0.029 0.046 0.036 0.052 0.008 0.022 0.035 0.007 ...
```

```
surveys2 <- select(surveys2, year, species_id, weight_kg)
colnames(surveys2)
```

```
## [1] "year"       "species_id" "weight_kg"
```

Use the filter() function to get all of the rows in the data frame surveys2 for the species ID "SH".

```
surveys2_filtered <- filter(surveys2, species_id == "SH")
str(surveys2_filtered)
```

```
## 'data.frame':    141 obs. of  3 variables:
##  $ year      : int  1978 1982 1982 1986 1987 1987 1987 1987 1987 1988 ...
##  $ species_id: chr  "SH" "SH" "SH" "SH" ...
##  $ weight_kg : num  0.089 0.106 0.052 0.055 0.077 0.078 0.104 0.058 0.052 0.06 ...
```

Excercise 2

```
surveys2 <- select(surveys, year ,species_id, weight) |>
mutate(weight_kg = weight/1000) |>
filter(!is.na(weight_kg)) |>
filter(species_id == "SH")
str(surveys2)
```

```
## 'data.frame':    141 obs. of  4 variables:
##  $ year      : int  1978 1982 1982 1986 1987 1987 1987 1987 1987 1988 ...
##  $ species_id: chr  "SH" "SH" "SH" "SH" ...
##  $ weight    : int  89 106 52 55 77 78 104 58 52 60 ...
##  $ weight_kg : num  0.089 0.106 0.052 0.055 0.077 0.078 0.104 0.058 0.052 0.06 ...
```

```
ds_data <- filter(surveys, species_id == "DS", !is.na(weight))
ds_data_by_year <- arrange(ds_data, year)
ds_weight_by_year <- select(ds_data_by_year, year, weight)
str(ds_weight_by_year)
```

```
## 'data.frame':    2344 obs. of  2 variables:
##  $ year  : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ weight: int  117 121 115 120 118 126 132 113 122 107 ...
```

```
ds_data <- filter(surveys,
                  species_id == "DS",
                  !is.na(weight))
ds_data_by_year <- arrange(ds_data, year)
ds_weight_by_year <- select(ds_data_by_year,
                            year,
                            weight)
```

Exercise 4

```
surveys_DS <- filter(surveys, species_id == "DS", !is.na(weight))
surveys_DS_lm <- lm(weight ~ year, data = surveys_DS)
summary(surveys_DS_lm)
```

```
##
## Call:
```

```
## lm(formula = weight ~ year, data = surveys_DS)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -109.787  -12.440    3.723   14.886   69.886
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -709.1968   263.2510  -2.694  0.00711 **
## year           0.4184     0.1328   3.150  0.00165 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22.86 on 2342 degrees of freedom
## Multiple R-squared:  0.00422,    Adjusted R-squared:  0.003795
## F-statistic: 9.925 on 1 and 2342 DF,  p-value: 0.001651
```

In Class Activity- (3/9/23) ~Missed

Data Grouping/ Aggregation

- groups data into variables so that manipulation can be done easily

```
group_by(surveys, year)
```

```
## # A tibble: 35,549 x 9
## # Groups:   year [26]
##    record_id month   day  year plot_id species_id sex   hindfoot_length weight
##        <int> <int> <int> <int>   <int> <chr>      <chr>           <int>  <int>
##  1         1     7    16  1977       2 NL         M                  32     NA
##  2         2     7    16  1977       3 NL         M                  33     NA
##  3         3     7    16  1977       2 DM         F                  37     NA
##  4         4     7    16  1977       7 DM         M                  36     NA
##  5         5     7    16  1977       3 DM         M                  35     NA
##  6         6     7    16  1977       1 PF         M                  14     NA
##  7         7     7    16  1977       2 PE         F                  NA     NA
##  8         8     7    16  1977       1 DM         M                  37     NA
##  9         9     7    16  1977       1 DM         F                  34     NA
## 10        10     7    16  1977       6 PF         F                  20     NA
## # ... with 35,539 more rows
```

```
group_by(surveys, plot_id, year)
```

```
## # A tibble: 35,549 x 9
## # Groups:   plot_id, year [622]
##    record_id month   day  year plot_id species_id sex   hindfoot_length weight
##        <int> <int> <int> <int>   <int> <chr>      <chr>           <int>  <int>
##  1         1     7    16  1977       2 NL         M                  32     NA
##  2         2     7    16  1977       3 NL         M                  33     NA
##  3         3     7    16  1977       2 DM         F                  37     NA
##  4         4     7    16  1977       7 DM         M                  36     NA
```

```
##  5         5     7    16  1977       3 DM         M                 35     NA
##  6         6     7    16  1977       1 PF         M                 14     NA
##  7         7     7    16  1977       2 PE         F                 NA     NA
##  8         8     7    16  1977       1 DM         M                 37     NA
##  9         9     7    16  1977       1 DM         F                 34     NA
## 10        10     7    16  1977       6 PF         F                 20     NA
## # ... with 35,539 more rows
```

```
surveys_by_year <- group_by(surveys, year)
year_counts <- summarize(surveys_by_year, abundance = n())
str(year_counts)
```

```
## tibble [26 x 2] (S3: tbl_df/tbl/data.frame)
##  $ year     : int [1:26] 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 ...
##  $ abundance: int [1:26] 503 1048 719 1415 1472 1978 1673 981 1438 942 ...
```

```
surveys_by_plot_year <- group_by(surveys, plot_id, year)
plot_year_counts <- summarise(surveys_by_plot_year, abundance = n())
```

```
## `summarise()` has grouped output by 'plot_id'. You can override using the
## `.groups` argument.
```

```
str(plot_year_counts)
```

```
## gropd_df [622 x 3] (S3: grouped_df/tbl_df/tbl/data.frame)
##  $ plot_id  : int [1:622] 1 1 1 1 1 1 1 1 1 1 ...
##  $ year     : int [1:622] 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 ...
##  $ abundance: int [1:622] 22 58 27 75 79 109 130 51 102 57 ...
##  - attr(*, "groups")= tibble [24 x 2] (S3: tbl_df/tbl/data.frame)
##   ..$ plot_id: int [1:24] 1 2 3 4 5 6 7 8 9 10 ...
##   ..$ .rows  : list<int> [1:24]
##   .. ..$ : int [1:26] 1 2 3 4 5 6 7 8 9 10 ...
##   .. ..$ : int [1:26] 27 28 29 30 31 32 33 34 35 36 ...
##   .. ..$ : int [1:26] 53 54 55 56 57 58 59 60 61 62 ...
##   .. ..$ : int [1:26] 79 80 81 82 83 84 85 86 87 88 ...
##   .. ..$ : int [1:26] 105 106 107 108 109 110 111 112 113 114 ...
##   .. ..$ : int [1:26] 131 132 133 134 135 136 137 138 139 140 ...
##   .. ..$ : int [1:26] 157 158 159 160 161 162 163 164 165 166 ...
##   .. ..$ : int [1:26] 183 184 185 186 187 188 189 190 191 192 ...
##   .. ..$ : int [1:26] 209 210 211 212 213 214 215 216 217 218 ...
##   .. ..$ : int [1:26] 235 236 237 238 239 240 241 242 243 244 ...
##   .. ..$ : int [1:26] 261 262 263 264 265 266 267 268 269 270 ...
##   .. ..$ : int [1:26] 287 288 289 290 291 292 293 294 295 296 ...
##   .. ..$ : int [1:26] 313 314 315 316 317 318 319 320 321 322 ...
##   .. ..$ : int [1:26] 339 340 341 342 343 344 345 346 347 348 ...
##   .. ..$ : int [1:26] 365 366 367 368 369 370 371 372 373 374 ...
##   .. ..$ : int [1:26] 391 392 393 394 395 396 397 398 399 400 ...
##   .. ..$ : int [1:26] 417 418 419 420 421 422 423 424 425 426 ...
##   .. ..$ : int [1:26] 443 444 445 446 447 448 449 450 451 452 ...
##   .. ..$ : int [1:26] 469 470 471 472 473 474 475 476 477 478 ...
##   .. ..$ : int [1:26] 495 496 497 498 499 500 501 502 503 504 ...
##   .. ..$ : int [1:26] 521 522 523 524 525 526 527 528 529 530 ...
```

```
##   .. ..$ : int [1:26] 547 548 549 550 551 552 553 554 555 556 ...
##   .. ..$ : int [1:26] 573 574 575 576 577 578 579 580 581 582 ...
##   .. ..$ : int [1:24] 599 600 601 602 603 604 605 606 607 608 ...
##   .. ..@ ptype: int(0)
##   ..- attr(*, ".drop")= logi TRUE
```

```
surveys |>
  group_by(plot_id, year) |>
  summarize(abundance = n(), avg_weight = mean(weight))
```

```
## `summarise()` has grouped output by 'plot_id'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 622 x 4
## # Groups:   plot_id [24]
##    plot_id  year abundance avg_weight
##      <int> <int>     <int>      <dbl>
##  1       1  1977        22         NA
##  2       1  1978        58         NA
##  3       1  1979        27         NA
##  4       1  1980        75         NA
##  5       1  1981        79         NA
##  6       1  1982       109         NA
##  7       1  1983       130         NA
##  8       1  1984        51         NA
##  9       1  1985       102         NA
## 10       1  1986        57         NA
## # ... with 612 more rows
```

```
mean(surveys$weight, na.rm = TRUE)
```

```
## [1] 42.67243
```

```
surveys |>
group_by(plot_id, year) |>
summarize(abundance = n(),
          avg_weight = mean(weight, na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'plot_id'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 622 x 4
## # Groups:   plot_id [24]
##    plot_id  year abundance avg_weight
##      <int> <int>     <int>      <dbl>
##  1       1  1977        22       37.8
##  2       1  1978        58       84.1
##  3       1  1979        27       76.4
##  4       1  1980        75       75.7
##  5       1  1981        79       79.9
##  6       1  1982       109       63.1
```

```
##  7       1  1983       130       63.8
##  8       1  1984        51       49.3
##  9       1  1985       102       66.4
## 10       1  1986        57       77.9
## # ... with 612 more rows
```

```
surveys |>
  group_by(plot_id, year) |>
  summarize(abundance = n(),
            avg_weight = mean(weight, na.rm = TRUE)) |>
  filter(!is.na(avg_weight))
```

```
## `summarise()` has grouped output by 'plot_id'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 618 x 4
## # Groups:   plot_id [24]
##    plot_id  year abundance avg_weight
##      <int> <int>     <int>      <dbl>
##  1       1  1977        22       37.8
##  2       1  1978        58       84.1
##  3       1  1979        27       76.4
##  4       1  1980        75       75.7
##  5       1  1981        79       79.9
##  6       1  1982       109       63.1
##  7       1  1983       130       63.8
##  8       1  1984        51       49.3
##  9       1  1985       102       66.4
## 10       1  1986        57       77.9
## # ... with 608 more rows
```

Exercise 6

```
shrub_data <- read_csv("../Data-raw/shrub-volume-data.csv")
```

```
## Rows: 12 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (5): site, experiment, length, width, height
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
select(shrub_data, length)
```

```
## # A tibble: 12 x 1
##    length
##     <dbl>
##  1    2.2
##  2    2.1
```

```
##  3    2.7
##  4    3
##  5    3.1
##  6    2.5
##  7    1.9
##  8    1.1
##  9    3.5
## 10    2.9
## 11    4.5
## 12    1.2
```

```
select(shrub_data, site, experiment)
```

```
## # A tibble: 12 x 2
##     site experiment
##    <dbl>      <dbl>
##  1     1          1
##  2     1          2
##  3     1          3
##  4     2          1
##  5     2          2
##  6     2          3
##  7     3          1
##  8     3          2
##  9     3          3
## 10     4          1
## 11     4          2
## 12     4          3
```

```
mutate(shrub_data, area = length*width)
```

```
## # A tibble: 12 x 6
##     site experiment length width height  area
##    <dbl>      <dbl>  <dbl> <dbl>  <dbl> <dbl>
##  1     1          1    2.2   1.3    9.6  2.86
##  2     1          2    2.1   2.2    7.6  4.62
##  3     1          3    2.7   1.5    2.2  4.05
##  4     2          1    3     4.5    1.5 13.5
##  5     2          2    3.1   3.1    4    9.61
##  6     2          3    2.5   2.8    3    7
##  7     3          1    1.9   1.8    4.5  3.42
##  8     3          2    1.1   0.5    2.3  0.55
##  9     3          3    3.5   2      7.5  7
## 10     4          1    2.9   2.7    3.2  7.83
## 11     4          2    4.5   4.8    6.5 21.6
## 12     4          3    1.2   1.8    2.7  2.16
```

```
arrange(shrub_data,length)
```

```
## # A tibble: 12 x 5
##     site experiment length width height
##    <dbl>      <dbl>  <dbl> <dbl>  <dbl>
```

```
##  1     3          2    1.1   0.5    2.3
##  2     4          3    1.2   1.8    2.7
##  3     3          1    1.9   1.8    4.5
##  4     1          2    2.1   2.2    7.6
##  5     1          1    2.2   1.3    9.6
##  6     2          3    2.5   2.8    3
##  7     1          3    2.7   1.5    2.2
##  8     4          1    2.9   2.7    3.2
##  9     2          1    3     4.5    1.5
## 10     2          2    3.1   3.1    4
## 11     3          3    3.5   2      7.5
## 12     4          2    4.5   4.8    6.5
```

```
filter(shrub_data, height > 5)
```

```
## # A tibble: 4 x 5
##    site experiment length width height
##   <dbl>      <dbl>  <dbl> <dbl>  <dbl>
## 1     1          1    2.2   1.3    9.6
## 2     1          2    2.1   2.2    7.6
## 3     3          3    3.5   2      7.5
## 4     4          2    4.5   4.8    6.5
```

```
filter(shrub_data,height > 5 & width > 2)
```

```
## # A tibble: 2 x 5
##    site experiment length width height
##   <dbl>      <dbl>  <dbl> <dbl>  <dbl>
## 1     1          2    2.1   2.2    7.6
## 2     4          2    4.5   4.8    6.5
```

```
filter(shrub_data, experiment == 1 | experiment == 3)
```

```
## # A tibble: 8 x 5
##    site experiment length width height
##   <dbl>      <dbl>  <dbl> <dbl>  <dbl>
## 1     1          1    2.2   1.3    9.6
## 2     1          3    2.7   1.5    2.2
## 3     2          1    3     4.5    1.5
## 4     2          3    2.5   2.8    3
## 5     3          1    1.9   1.8    4.5
## 6     3          3    3.5   2      7.5
## 7     4          1    2.9   2.7    3.2
## 8     4          3    1.2   1.8    2.7
```

```
shrub_volumes <- mutate(shrub_data, volume = height*width*length)
```