Grouping & Summarizing DataFrames

Author: Cole Brookson **Date:** 20 July 2022

To obtain a useful summary of a dataframe is useful when trying to get summary statistics or the like from a dataframe. Let's use an example dataframe that describes the masses of Bison.

```
# load libraries
library(lterdatasampler)
library(tidyverse)

bison <- lterdatasampler::knz_bison
head(bison)
```

```
## # A tibble: 6 x 8
##   data_code rec_year rec_month rec_day animal_code animal_sex animal_w~1 anima~2
##   <chr>        <dbl>     <dbl>   <dbl> <chr>       <chr>           <dbl>   <dbl>
## 1 CBH01         1994        11       8 813         F                 890    1981
## 2 CBH01         1994        11       8 834         F                1074    1983
## 3 CBH01         1994        11       8 B-301       F                1060    1983
## 4 CBH01         1994        11       8 B-402       F                 989    1984
## 5 CBH01         1994        11       8 B-403       F                1062    1984
## 6 CBH01         1994        11       8 B-502       F                 978    1985
## # ... with abbreviated variable names 1: animal_weight, 2: animal_yob
```

It might be useful to understand how the weights of these animals differs by sex. Let's first check the `animal_sex` column to see what values it takes and what type of data the column is:

```
str(bison$animal_sex)
```

```
##  chr [1:8325] "F" "F" "F" "F" "F" "F" "F" "F" "F" "F" "F" "F" "F" "F" "F" "F" ...
```

```
unique(bison$animal_sex)
```

```
## [1] "F" "M"
```

So there are two values, and the column is a character type.

Grouping with One Variable

Let's try to get the mean and standard deviation for each sex. We could do this by making two different dataframes and then calculating those values, but it might be easier to make this into one separate object that has all the information we need.

To do this we can use the `group_by()` function from the `dplyr` package to help us. We will need to group the dataset by the variable we're interested in, and then we can use the `summarize()` function also from `dplyr` to get our values of interest.

To understand how the summarize function works, we can run `?summarize()` in the console. We see that we will get a new dataframe, with one row for each combination of grouping variables. Here we are only using one grouping variable, and there are two levels to the variable, so we should only have two rows left.

Summarize will make a new column or set of columns, that we can create with base summary functions like `summarize(mean = mean())`.

Note that when using `summarize()`, all columns *not* involved in the grouping will be removed from the resulting dataframe. So, in our case, we will only have one column (`animal_weight`) left from our original dataframe.

So to group our dataframe and then use `summarize()` to get our mean and standard deviation for our measures of animal weight, we can pipe these commands together.

```
mass_by_sex = bison %>%
  dplyr::group_by(animal_sex) %>%
  dplyr::summarize(mean_mass = mean(animal_weight, na.rm = TRUE),
                   std_dev = sd(animal_weight, na.rm = TRUE))
mass_by_sex
```

```
## # A tibble: 2 x 3
##   animal_sex mean_mass std_dev
##   <chr>          <dbl>   <dbl>
## 1 F               762.    282.
## 2 M               728.    420.
```

Note here that we've used `na.rm = TRUE` to ensure that when `mean()` and `sd()` calculate their values, if there are any `NA` values in the data, they are ignored.

Grouping with Multiple Variables

In theory, we can use `summarize()` grouped by as many variables as we want. To demonstrate this, we can repeat our measurement above, but now also grouping my the month the weight was measured in. We may assume that the animals' weights will fluctuate throughout the year, and it could be useful to understand how this differs by sex (if at all).

So we will again use `group_by()` and `summarize()` to perform this task, but now grouping by both `animal_sex` and `rec_month`:

```
month_sex_weight = bison %>%
  dplyr::group_by(animal_sex, rec_month) %>%
  dplyr::summarize(mean_mass = mean(animal_weight, na.rm = TRUE),
                   std_dev = sd(animal_weight, na.rm = TRUE))
```

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

```
month_sex_weight
```

```
## # A tibble: 4 x 4
## # Groups:   animal_sex [2]
##   animal_sex rec_month mean_mass std_dev
##   <chr>          <dbl>     <dbl>   <dbl>
## 1 F                 10      771.    281.
## 2 F                 11      751.    283.
## 3 M                 10      756.    436.
## 4 M                 11      701.    402.
```

In our little observational look here, we can see that for both sexes, the average weights dropped between October (month 10) and November (month 11).