Exploring data #1

Summaries for numeric data

Data types and vector classes

Here are a few common vector classes in R:

Class	Example
`character`	"Chemistry", "Physics", "Mathematics"
`numeric`	10, 20, 30, 40
`factor`	Male [underlying number: 1], Female [2]
`Date`	"2010-01-01" [underlying number: 14,610]
`logical`	TRUE, FALSE [underlying numbers: 1, 0]

Numeric vectors

To explore numeric vectors, there are a few base R functions that are very helpful. For example:

Function	Description
`min()`	Minimum of values in the vector
`max()`	Maximum of values in the vector
`mean()`	Mean of values in the vector
`median()`	Median of values in the vector

Simple statistic examples

All of these take, as the main argument, the vector(s) for which you want the statistic.

```
mean(x = beijing_pm$value)
```

```
## [1] 63.18646
```

```
min(x = beijing_pm$value)
```

```
## [1] -999
```

If there are missing values in the vector, you'll need to add an option to say what to do when them (e.g., `na.rm` or `use="complete.obs"`—see help files).

Simple statistic examples

These functions require a **numeric vector** as input.

Remember that you can pull a column from a dataframe as a vector using either `$` or the `pull` function from `dplyr`. Therefore, you can use either of these calls to get the mean weight of the children in the dataset:

```
mean(beijing_pm$value)
```

```
## [1] 63.18646
```

```
beijing_pm %>%
  pull(value) %>%
  mean()
```

```
## [1] 63.18646
```

The `summarize` function

Within a "tidy" workflow, you can use the `summarize` function from the `dplyr` package to create summary statistics for a dataframe. This function inputs a dataframe and outputs a dataframe with the specified summary measures.

The `summarize` function

The basic format for using `summarize` is:

```
## Generic code
summarize(dataframe,
          summary_column_1 = function(existing_columns),
          summary_column_2 = function(existing_columns))
```

The `summarize` function

As an example, to summarize the `beijing_pm` dataset to get the minimum, mean, and maximum $PM_{2.5}$ concentrations, you could run:

```
summarize(.data = beijing_pm,
          min_pm = min(value),
          mean_pm = mean(value),
          max_pm = max(value))
```

```
## # A tibble: 1 x 3
##   min_pm mean_pm max_pm
##    <dbl>   <dbl>  <dbl>
## 1   -999    63.2    684
```

Notice that the output is one row (since the summary was on ungrouped data), with three columns (since we defined three summaries in the `summarize` function).

The `summarize` function

Because the first input to the `summarize` function is a dataframe, you can "pipe into" a `summarize` call. For example, we could have written the code on the previous slide as:

```
beijing_pm %>%
  summarize(min_pm = min(value),
            mean_pm = mean(value),
            max_pm = max(value))
```

As another note, because the output from `summarize` is also a dataframe, we could also "pipe into" another tidyverse function after running `summarize`.