

Day 3 Cheatsheet

Subsetting Data in R

Functions

| Library/Package | Piece of code | Example of usage | What it does |
|-----------------|-----------------------------------|--|---|
| Base R | <code>nrow(x); ncol(x)</code> | <code>nrow(x); ncol(x)</code> | Get the number of rows and the number of columns in an object <code>x</code> , respectively. |
| Base R | <code>dim(x)</code> | <code>dim(x)</code> | Get the number of rows <i>and</i> number of columns in an object <code>x</code> |
| dplyr | <code>glimpse(x)</code> | <code>glimpse(mtcars)</code> | Get an overview of data frame <code>x</code> |
| dplyr | <code>slice_sample(x)</code> | <code>slice_sample(mtcars)</code> | See a random subset of the rows of <code>x</code> |
| Base R | <code>data.frame()</code> | <code>df <- data.frame(1:3)</code> | Creates a data frame where the named arguments will be the same length. |
| Base R | <code>tibble()</code> | <code>tibble(mtcars)</code> | Creates a tibble from a data.frame or matrix. |
| tibble | <code>column_to_rownames()</code> | <code>df <- df %>% column_to_rownames('existing_variable_name')</code> | Transforms an existing column into the rownames. |
| tibble | <code>rownames_to_column()</code> | <code>df <- df %>% rownames_to_column('new_variable_name')</code> | Transforms the rownames of a data frame into a column (which is added to the start of the data frame). The string supplied as an argument will be the name of the new column. |
| dplyr | <code>rename()</code> | <code>df <- dplyr::rename(df, MPG = mpg)</code> | Renames designated columns while keeping all variables of the data.frame |
| dplyr | <code>pull()</code> | <code>pull(df, 'existing_variable_name')</code> | Extract a column as a vector |
| dplyr | <code>select()</code> | <code>select(df, 'existing_variable_name')</code> | Selects columns that match the specified argument |
| dplyr | <code>filter()</code> | <code>filter(df, mpg > 20)</code> | Returns a subset of rows matching the conditions of the specified logical argument |

| Library/Package | Piece of code | Example of usage | What it does |
|-----------------|-----------------------------------|--|--|
| Base R | <code>==, <=, >=, !=</code> | <code>filter(df, mpg > 20)</code> | These are binary operators which allow for the comparison of values in an object. They are handy for use with <code>dplyr::filter()</code> |
| Base R | <code>%in%</code> | <code>filter(df, mpg %in% c(20,21,22))</code> | Checks if the given value(s) on the left side of the operator are in the vector or other R object defined on the right side of the operator. It returns a logical TRUE or FALSE statement. |
| dplyr | <code>%>%</code> | <code>df <- df %>% select('new_variable_name')</code> | Filters a data.frame through tidyverse operations |
| dplyr | <code>mutate()</code> | <code>df <- mutate(df, newcol = wt/2.2)</code> | Adds a new column that is a function of existing columns |
| dplyr | <code>relocate()</code> | <code>df_carb <- relocate(.data = df, wt, .before = mpg)</code> | Reorder columns in a data frame or tibble |
| dplyr | <code>arrange()</code> | <code>df <- arrange(df, mpg)</code> | Reorders rows in ascending order. <code>arrange(desc())</code> would reorder rows in descending order. |
| dplyr | <code>case_when()</code> | <code>df <- arrange(df, mpg)</code> | This function allows you to vectorise multiple <code>if_else()</code> statements. If no cases match, NA is returned. |
| Base R | <code>colnames()</code> | <code>colnames(df)</code> | Gets or sets the column names of a matrix or data frame. |

- See `tidyselect` helpers for handy things to use with `select()`.

* This format was adapted from the cheatsheet format from AlexsLemonade.