Dropping and moving columns

DATA MANIPULATION IN JULIA

Katerina Zahradova
Instructor

How to select most columns

```
# Selecting all columns from chocolates except the review_date column
select(chocolates, :company, :bean_origin, :REF, :cocoa,
        :company_location, :ratings, :bean_type, :bean_location)
```

```
ArgumentError: column name "ratings" not found in the data frame;
    existing most similar names are: "rating"
...
```

Not() operator

```
# Selecting all columns from chocolates except the review_date column
select(chocolates, Not(:review_date))
```

```
1795×9 DataFrame
Row company   bean_origin REF   cocoa   company_location rating  bean_type bean_location
    String    String      Int64 Float64 String31         Float64 String31? String31?

-------------------------------------------------------------------------------------
1   A. Morin Agua Grande 1876  63.0    France           3.75    Sao Tome
...
```

select() and not() pitfalls

```
# Dropping the same column twice
select!(chocolates, Not(:review_date))


# Several lines of code later ...


select(chocolates, Not(:review_date))
```

```
ArgumentError: column name :review_date not found in the data frame
```

Dropping column that is not there safely

```julia
# Using Cols
select(chocolates, Not(Cols(==("review_date"))))
```

```
1795×9 DataFrame
Row company   bean_origin REF   cocoa   company_location rating  bean_type bean_location
    String    String      Int64 Float64 String31         Float64 String31? String31?
...
```

```julia
# Using regex
select(chocolates, Not(r"review\_date"))
```

```
1795×9 DataFrame
Row company   bean_origin REF   cocoa   company_location rating  bean_type bean_location
    String    String      Int64 Float64 String31         Float64 String31? String31?
...
```

Reordering columns

```
# Pre-selected chocolates
first(chocolates)
```

```
1795x5 DataFrame
Row  company   review_date  cocoa    rating   bean_location
     String    Int64        Float64  Float64  String31?

----------------------------------------------------------
1    A. Morin  2016         63.0     3.75     Sao Tome
```

Move it to the left

```
# Moving cocoa to the left
select(chocolates, :cocoa, :)
```

```
1795x5 DataFrame
Row  cocoa    company   review_date  rating   bean_location
     Float64  String    Int64        Float64  String31?

-----------------------------------------------------------
1    63.0     A. Morin  2016         3.75     Sao Tome
...
```

Move them to the left

```julia
# Moving cocoa and rating to the left
select(chocolates, :cocoa, :rating, :)
```

```
1795x5 DataFrame

Row  cocoa     rating   company   review_date  bean_location
     Float64   Float64  String    Int64        String31?
------------------------------------------------------------
1    63.0      3.75     A. Morin  2016         Sao Tome
...
```

Move it right

```
# Moving company to the right
select(chocolates, Not(:company), :company)
```

```
1795x5 DataFrame
Row  review_date  cocoa    rating   bean_location  company
     Int64        Float64  Float64  String31?      String

----------------------------------------------------------
1    2016         63.0     3.75     Sao Tome       A. Morin
...
```

Move them all around

```julia
# Combine the two moves
select(chocolates, :cocoa, :rating, Not(:company), :company)
```

```
1795x5 DataFrame
Row cocoa    rating   bean_location  review_date  company
    Float64  Float64  String31       Int64        String

-----------------------------------------------------------
1   63.0     3.75     Sao Tome       2016         A. Morin
...
```

Move and drop at the same time

```
# Reorder and drop review_date
# Combine the two moves
select(chocolates, :cocoa, :rating, Not([:company, :review_date]), :company)
```

```
1795x4 DataFrame
Row cocoa    rating   bean_location  company
    Float64  Float64  String31       String

----------------------------------------------------------
1   63.0     3.75     Sao Tome       A. Morin
...
```

Cheat sheet

Dropping columns:

```
# Drop col1 and col 2
select(df, Not([:col1, "col 2"]))
```

Dropping columns safely:

```
# Drop col1 and col 2 that might not exist
select(df, Not(r"col1"), Not(Cols(==("col 2"))))
```

Moving columns to the left

```
# Move col1 and col 2 to the left
select(df, :col1, "col 2", :)
```

Moving columns to the right

```
# Move col1 and col 2 to the right
select(df, Not([:col1, "col 2"]) ,:col1, "col 2")
```

Let's practice!

DATA MANIPULATION IN JULIA

Manipulating columns

DATA MANIPULATION IN JULIA

Katerina Zahradova
Instructor

Applying functions

- Functions taking whole columns
  - Features determined by whole column, e.g., mean, minimum, etc.
  - In Julia, functions such as maximum are written in full as `maximum()` , not just `max()`
- Functions working on individual lines

Options

- `select()`
- `transform()`
- `combine()`

To mutate a DataFrame in place:

- `select!()` , `transform!()` , `combine!()`

select()

```julia
# Selecting columns
select(penguins, :species, :body_mass_g)
```

```
333x2 DataFrame
Row species   body_mass_g
    String15  Int64

---------------------------
1   Adelie    3750
2   Adelie    3800
3   Adelie    3250
...
```

```julia
# Selecting and renaming columns
select(penguins, :species, :body_mass_g => :weight_g)
```

```
333x2 DataFrame
Row species   weight_g
    String15  Int64

---------------------------
1   Adelie    3750
2   Adelie    3800
3   Adelie    3250
...
```

select()

```julia
# Select columns and apply functions
select(penguins, :species, :body_mass_g => mean)
```

```
333x2 DataFrame
Row species   body_mass_g_mm
    String15  Float64

---------------------------
1   Adelie    4207.06
2   Adelie    4207.06
3   Adelie    4207.06
...
```

transform()

```julia
# Adding column with maximum of body_mass_g
transform(penguins, :body_mass_g => maximum)
```

```
333x8 DataFrame
Row species   island    ...  body_mass_g  sex      body_mass_g_maximum
    String15  String15  ...  Int64        String7  Float64

-------------------------------------------------------------------
1   Adelie    Torgersen ...  3750         MALE     4207.06
2   Adelie    Torgersen ...  3800         FEMALE   4207.06
...
```

combine()

```julia
# Combining penguins with maximum of body_mass_g
combine(penguins, :body_mass_g => maximum)
```

```
1×1 DataFrame
Row  body_mass_g_mean
     Float64

──────────────────────────
1    4207.06
```

How to handle multiples

```
# Using multiple functions on a column
combine(penguins, :body_mass_g .=> [mean, minimum, maximum])
```

```
Row  body_mass_g_mean  body_mass_g_minimum  body_mass_g_maximum
     Float64           Float64              Float64

---------------------------------------------------------------
1    4207.06           2700                 6300
```

```
# Passing multiple columns to a function
select(penguins, [:body_mass_g, :flipper_length_mm] .=> mean)
```

```
Row  body_mass_g_mean  flipper_length_mm_mean
     Float64           Float64

-------------------------------------------
1    4207.06           200.967
2    4207.06           200.967
...
```

Cheat sheet

- `select()` :
  - Only includes specified columns
  - Same number of rows; same value is broadcasted over all rows
- `transform()` :
  - Keeps all columns and adds new ones
  - Same number of rows same value is broadcasted over all rows
- `combine()` :
  - Only includes specified columns
  - Does not broadcast the values over all rows

Let's practice!

DATA MANIPULATION IN JULIA

Creating new columns

DATA MANIPULATION IN JULIA

Katerina Zahradova
Instructor

Columns vs. rows

Column

- Single number for all rows
- E.g., mean, median, sum, ...

Rows

- Values depend on data in each individual row

=> using `ByRow()`

Flipper length to inches

```julia
# Convert mm to inches
transform(penguins, :flipper_length_mm => ByRow(x -> x/25.4) => :flipper_length_inch)
```

```
333x8 DataFrame
Row species    island     ...  body_mass_g  sex      flipper_lenght_inch
    String15   String15   ...  Int64        String7  Float64

-------------------------------------------------------------------
1   Adelie     Torgersen ...  3750         MALE     7.12598
2   Adelie     Torgersen ...  3800         FEMALE   7.32283
...
```

Culmen depth and length ratio

```
# Select columns and calculate their ratio
select(penguins,:culmen_depth_mm, :culmen_length_mm,
   [:culmen_depth_mm, :culmen_length_mm] => ByRow((x, y) -> x/y) => :culmen_ratio)
```

```
333x3 DataFrame
Row  culmen_depth_mm  culmen_length_mm  culmen_ratio
     Float64          Float64           Float64

----------------------------------------------------
1    18.7             39.1              0.478261
2    17.4             39.5              0.440506
...
```

New column from a vector

```julia
# Vector id_vec that we want to add

# Using [] and :
penguins[:, :id_colon] = id_vec

# Using [] and !
penguins[!, :id_exclamation] = id_vec

# Using .
penguins.id_dot = id_vec
```

Copy or not

```
penguins[:, :id_colon] = id_vec
penguins[!, :id_exclamation] = id_vec
penguins.id_dot = id_vec

# Change first element
id_vec[1] = 27
select(penguins, :species, r"id")
```

```
333x4 DataFrame
Row  species   id_colon  id_exclamation  id_dot
     String15  Int64     Int64           Int64
-----------------------------------------------------
1    Adelie    25        27              27
...
```

Copy or not

```
penguins[:, :id_colon] = id_vec         # copies values to the DataFrame
penguins[!, :id_exclamation] = id_vec  # references id_vec
penguins.id_dot = id_vec               # references id_vec


# Change first element
id_vec[1] = 27
select(penguins, :species, r"id")
```

```
333x4 DataFrame
Row  species   id_colon  id_exclamation  id_dot
     String15  Int64     Int64           Int64

-----------------------------------------------------
1    Adelie    25        27              27
...
```

Let's practice!

DATA MANIPULATION IN JULIA