DATA MANIPULATION IN R WITH DATA.TABLE

Selecting columns from a data.table

Matt Dowle, Arun Srinivasan
Instructors, DataCamp

General form of data.table syntax (Recap)

Second argument `j` is used to *select* (and compute on) columns

```
# General form of data.table syntax
DT[i, j, by]
   |  |  |
   |  |  --> grouped by what?
   |  -----> what to do?
   --------> on which rows?
```

Using column names to select columns

`j` argument accepts a character vector of column names

```
ans <- batrips[, c("trip_id", "duration")]
head(ans, 2)
   trip_id duration
1:  139545      435
2:  139546      432
```

Using column names to select columns

```
batrips_df <- as.data.frame(batrips)
ans <- batrips_df[, "trip_id"]
head(ans, 2)

# The result is a vector,
# not a data.frame
[1] 139545, 139546
```

```
ans <- batrips[, "trip_id"]
# Still a data.table, not a vector
head(ans, 2)
   trip_id
1:  139545
2:  139546
```

Using column numbers to select columns

Column numbers instead of names work just fine

```
ans <- batrips[, c(2, 4)]
head(ans, 2)
   duration           start_station
1:      435 San Francisco City Hall
2:      432 San Francisco City Hall
```

However, we consider this a *bad practice*

```
# If the order of columns changes, the result is wrong
batrips[, c(2, 4)]

# The result is always correct, no matter the order
batrips[, c("duration", "start_station")]
```

Deselecting columns with character vectors

- `-c("col1", "col2", ...)` *deselects* the specified columns
- Convenience feature only in data.table
- Using `!` instead of `-` works the same way

```
# Select all cols *except* those shown below
ans <- batrips[, -c("start_date", "end_date", "end_station")]
head(ans, 1)
   trip_id duration           start_station start_terminal end_terminal
1:  139545      435 San Francisco City Hall             58           65
   bike_id subscription_type zip_code
1:     473        Subscriber    94612
```

Selecting columns the data.table way

Remember how columns were used as if they are variables in `i` argument in the last chapter?

```
# Recap the "i" argument
# All trips more than an hour
batrips[duration > 3600]
```

Similarly, you can use a *list of variables* (column names) to select columns

```
ans <- batrips[, list(trip_id, dur = duration)]
head(ans, 2)
   trip_id      dur
1:  139545      435
2:  139546      432
```

Selecting columns the data.table way

When selecting a single column, not wrapping the variable by `list()` returns a `vector`

```
# Select a single column and return a data.table
ans <- batrips[, list(trip_id)]
head(ans ,2)
   trip_id
1:  139545
2:  139546

# Select a single column and return a vector
ans <- batrips[, trip_id]
head(ans, 2)
[1] 139545 139546
```

Selecting columns the data.table way

`.()` is an alias to `list()`, for convenience

```
# .() is the same as list()
ans <- batrips[, .(trip_id, duration)]
head(ans, 2)
   trip_id duration
1:  139545      435
2:  139546      432
```

DATA MANIPULATION IN R WITH DATA.TABLE

Let's practice!

DATA MANIPULATION IN R WITH DATA.TABLE

Computing on columns the data.table way

Matt Dowle, Arun Srinivasan
Instructors, DataCamp

Computing on columns

Since columns can be referred to as variables, you can *compute* directly on them in `j`

```
# Compute mean of duration column using the data.table way
ans <- batrips[, mean(duration)]
[1] 1131.967

# Compute mean of duration column using the data.frame way
ans <- mean(batrips[, "duration"])
[1] 1131.967
```

Computing on rows and columns

Combining `i` and `j` is *straightforward*

```
# Compute mean of duration column for "Japantown" start station
batrips[start_station == "Japantown", mean(duration)]
[1] 2464.331
```

Special symbol .N in j

- `.N` can be used in `j` as well
- Particularly useful to get the number of rows after filtering in `i`

```
# How many trips started from "Japantown"?
batrips[start_station == "Japantown", .N]
[1] 902

# Compare this to the data.frame way
nrow(batrips[batrips$start_station == "Japantown", ])
[1] 902
```

DATA MANIPULATION IN R WITH DATA.TABLE

Let's practice!

DATA MANIPULATION IN R WITH DATA.TABLE

Advanced computations in j

Matt Dowle, Arun Srinivasan
Instructors, DataCamp

Compute in j and return a data.table

Recall that you can select multiple columns using `.()`

```
# Recap: Select trip_id and duration columns
ans <- batrips[, .(trip_id, dur = duration)]
head(ans, 2)
   trip_id      dur
1:  139545      435
2:  139546      432
```

You can compute on multiple columns and return a data.table the same way

```
# Get mean and median of duration
batrips[, .(mn_dur = mean(duration), med_dur = median(duration))]
      mn_dur med_dur
1:  1131.967     511
```

Question

- How would you perform this operation using the data frame way?
- Is your code straightforward and clear?

```
# Get mean and median of duration
batrips[, .(mn_dur = mean(duration), med_dur = median(duration))]
      mn_dur med_dur
1:  1131.967     511
```

Combining with i

Together with `i`, you can compute on columns in `j` only for those rows that satisfy a condition

```
batrips[start_station == "Japantown", .(mn_dur = mean(duration),
                                        med_dur = median(duration))]
     mn_dur med_dur
1: 2464.331     782
```

Question

- How would you perform this operation using the data frame way?
- Is your code straightforward and clear?

```
batrips[start_station == "Japantown", .(mn_dur = mean(duration),
                                        med_dur = median(duration))]
     mn_dur med_dur
1: 2464.331     782
```

DATA MANIPULATION IN R WITH DATA.TABLE

Let's practice!