-Ladies Istanbul

Data Manipulation with dplyr Package

November 5 - @Binovist

istanbul@rladies.org
hazel@rladies.org

For Beginners

Getting Started

R commands;

- are case sensitive
- can be separated either by a semi-colon (‘;’), or by a newline
- #comment

Objects;

- variables, arrays of numbers, character strings, functions

Need Help;

- ?summary
- help(summary)
- example(summary)

```
• #this is R-Ladies Istanbul
• A <- 10 ; a <- 3
• print(paste("A is", A))
• print(paste("a is", a))
• cat("A and a are equal? = ", A == a)

• myNumbers <- c(1:10)
• rep(myNumbers, times = 3)
• twice <- rep(myNumbers, each = 2)

• ls()
• rm(a)
• print(twice)
```

Assignments, Basic Operators

Assignments

- use <- or -> symbol combination

Basic arithmetic operators

- +,-,*,/,^, %%

Logical operators

- <, >, <=, >=, ==, !=, !x, x & y, x | y

Others

- sum, sqrt, min, max, mean, var, sd, abs, summary

Basic Operators

`is.na(x)`	test if x is NA
`!is.na(x)`	test if x is not Na
`x %in% y`	test if x is in y
`!(x %in% y)`	test if x is not in y
`!x`	not x

- `x <- c(1:15)`
- `dailyshow <- read.csv(file = "https://raw.githubusercontent.com/../daily_show_guests.csv", sep = ";", header = TRUE)`
- `seq(from = 2, to = 100, by = 2) -> y`
- `sum(x), min(x), max(x), mean(x), var(x), sqrt(x), sd(x), length(x)`

Save your codes

&

Keep track of them

R Script or R Markdown

- R Script: File -> New File -> R Script
- R Markdown: File -> New File -> R Markdown

R Script

- Type your code in the R Script
- Use curser + Run or highlight + Run
- Use # for comments
- Run the codes: Cmd + Enter or Highlight + Run

R Markdown

- Helps to create of dynamics documents, presentation and reports
- Fully reproducible
- Source: http://rmarkdown.rstudio.com/

Packages

&

Loading Data

Install.Packages & Library

Install the Packages by running the codes in the Console

- `install.packages("readr")`
- `install.packages("dplyr")`
- `install.packages("tidyr")`

Then load the packages by running the following codes

- `library(readr)`
- `library(dplyr)`
- `library(tidyr)`

!!Important!!

Example: use dplyr::select(data, v1,…)

tidyverse: Easily Install Tidyverse Packages

- broom, dplyr, tidyr, ggplot2, lubridate
- magrittr, purrr, modelr, readxl
- stringr, forcats, tibble

Tidy Data

In **tidy data**

- Each variable forms a column
- Each observation forms a row
- Each type of observational unit forms a table

tidyr package is helpful for converting messy data to tidy data

Today we'll start with tidy data, later we'll learn how to deal with messy data!!

Let's Go!

- `install.packages("dplyr")`
- `install.packages("hflights")`
- `library(dplyr)`
- `library(hflights)`

Write *?hflights* in Console and look what has come to Help section.

Understanding the Data

```
tbls <- tbl_df(hflights)
glimpse(tbls)
?hflights
```

- `str(hflights)`
- `glimpse(hflights)`
- `names(hflights)`
- `ncol(hflights); nrow(hflights)`
- `head(hflights); tail(hflights, n = 10)`
- `summary(hflights)`
- `View(hflights)`
- `print(hflights)` *#use carefully when your data is big!*

Understanding the Data

```
> glimpse(tbls)
Observations: 227,496
Variables: 21
$ Year              <int> 2011, 2011, 2011, 2011, 2011, 2011, 2011, 2011, 2011,...
$ Month             <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ DayofMonth        <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16...
$ DayOfWeek         <int> 6, 7, 1, 2, 3, 4, 5, 6, 7, 1, 2, 3, 4, 5, 6, 7, 1, 2,...
$ DepTime           <int> 1400, 1401, 1352, 1403, 1405, 1359, 1359, 1355, 1443,...
$ ArrTime           <int> 1500, 1501, 1502, 1513, 1507, 1503, 1509, 1454, 1554,...
$ UniqueCarrier     <chr> "AA", "AA", "AA", "AA", "AA", "AA", "AA", "AA", "AA",...
$ FlightNum         <int> 428, 428, 428, 428, 428, 428, 428, 428, 428, 428, 428...
$ TailNum           <chr> "N576AA", "N557AA", "N541AA", "N403AA", "N492AA", "N2...
$ ActualElapsedTime <int> 60, 60, 70, 70, 62, 64, 70, 59, 71, 70, 70, 56, 63, 6...
$ AirTime           <int> 40, 45, 48, 39, 44, 45, 43, 40, 41, 45, 42, 41, 44, 4...
$ ArrDelay          <int> -10, -9, -8, 3, -3, -7, -1, -16, 44, 43, 29, 5, -9, -...
$ DepDelay          <int> 0, 1, -8, 3, 5, -1, -1, -5, 43, 43, 29, 19, -2, -3, -...
$ Origin            <chr> "IAH", "IAH", "IAH", "IAH", "IAH", "IAH", "IAH", "IAH...
$ Dest              <chr> "DFW", "DFW", "DFW", "DFW", "DFW", "DFW", "DFW", "DFW...
$ Distance          <int> 224, 224, 224, 224, 224, 224, 224, 224, 224, 224, 224...
$ TaxiIn            <int> 7, 6, 5, 9, 9, 6, 12, 7, 8, 6, 8, 4, 6, 5, 6, 12, 8, ...
$ TaxiOut           <int> 13, 9, 17, 22, 9, 13, 15, 12, 22, 19, 20, 11, 13, 15,...
$ Cancelled         <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
$ CancellationCode  <chr> "", "", "", "", "", "", "", "", "", "", "", "", "", "...
$ Diverted          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
```

Understanding the Data

- Year, Month, DayofMonth: date of departure
- DayOfWeek: day of week of departure (useful for removing weekend effects)
- DepTime, ArrTime: departure and arrival times (in local time, hhmm)
- UniqueCarrier: unique abbreviation for a carrier
- FlightNum: flight number
- TailNum: airplane tail number
- ActualElapsedTime: elapsed time of flight, in minutes
- AirTime: flight time, in minutes
- ArrDelay, DepDelay: arrival and departure delays, in minutes
- Origin, Dest origin and destination airport codes
- Distance: distance of flight, in miles
- TaxiIn, TaxiOut: taxi in and out times in minutes
- Cancelled: cancelled indicator: 1 = Yes, 0 = No
- CancellationCode: reason for cancellation: A = carrier, B = weather, C = national air system, D = security
- Diverted: diverted indicator: 1 = Yes, 0 = No

Five Magical Words!

Variables (columns)

- `select`
- `mutate`

Observations (rows)

- `filter`
- `arrange`

Groups

- `summarise`

dplyr package do not change the original data set.

%>% (pipe) operator

- magrittr package
- basically tells R to take the value of that which is to the left and pass it to the right as an argument.
- cmd + shft + m
- kntr + shft + m

```
hflights %>% mutate(diff = TaxiOut -
TaxiIn) %>% filter(!is.na(diff)) %>%
summarise(minimum = min(diff))
```

select

- `select(dataframe, var1, var2,…)`
- `select(dataframe, 1:4, -2)`

Helper functions

- starts_with, ends_with, contains

select function returns a modified copy, doesn't change the data.

select

- `select(hflights, 1:8)`
- `select(hflights, AirTime, ArrDelay, TaxiIn)`
- `select(hflights, ends_with("Time"))`
- `select(hflights, UniqueCarrier, ends_with("Delay"), starts_with("Cancel"))`
- `select(hflights, contains("Tim"),contains("Del"))`

mutate

Deals with info in your data which is not display

- `mutate(dataframe, new = var1 + var2)`
- `mutate(my_df, x = a + b, y = x + c)`

mutate(dataframe, new_Var = expression)

mutate

- `t1 <- mutate(hflights, ActualGroundTime = ActualElapsedTime - AirTime)`
- `t2 <- mutate(t1, GroundTime = TaxiIn + TaxiOut)`

filter

*Filter out rows, specific type of **observation***

```
filter(dataframe, logicaltest)
```

filter

- `filter(hflights, DepTime < 1700 | ArrTime > 2200)`
- `filter(hflights, Cancelled == 1, DepDelay > 0)`

exp: DayofWeek %in% c(6,7)

select & filter & pipe

- `f1 <- select(hflights, starts_with("Cancel"), DepDelay)`
- `filter(f1, Cancelled == 1)`
- `filter(f1, Cancelled == 1) %>% head()`

Your Turn!

Exercise! Let's practice with 3 verbs!

- filter destination only "JFK" and assign results to myDest
- select variables (from myDest) contain "Time" and "Taxi", and assign results of myDest2
- add a new column which calculates difference between TaxiOut and TaxiIn, assign the results myDest3

arrange

Help order observation *(default ascending)*

- `arrange(dataframe, var1)`
- `arrange(dataframe, var1, desc(var2))`

arrange

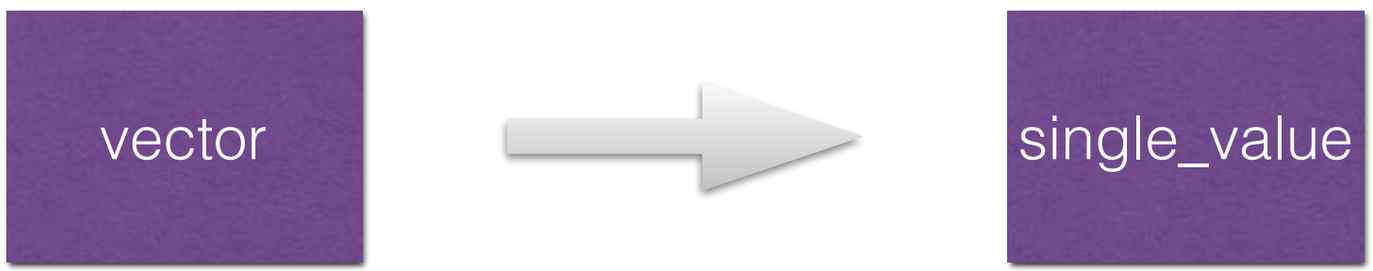
- `a1 <- select(hflights, TailNum, contains("Delay"))`
- `arrange(a1, DepDelay)`

- `arrange(hflights, UniqueCarrier, desc(DepDelay))`
- `arrange(hflights, Total = DepDelay + ArrDelay)`

summarise

Builds a new dataset that contains only the summarising statistics

- `summarise(dataframe, newColname = expression,. . .)`
- `summarise(dataframe, minimum = min(A), avg = mean(B)..)`



summarise

```
a1 <- filter(a1, !is.na(DepDelay))
summarise(a1, min = min(DepDelay),
  max = max(DepDelay), avg = mean(DepDelay),
  med = median(DepDelay))
```

Your Turn!

Try and try and try!

We learnt 5 verbs and how they work with %>% operator

Do it in one row! (remember %>% operator?)

- Use hflights data
- Add new column called diff by calculation difference between TaxiIn and TaxiOut
- Filter values - use only not NA's
- And find average value of diff variable and summarise

And there is more…

You can search for:

- `group_by`
- `rename`
- `sample_n & sample_frac`
- `transmute`
- `slice`

Sources

- Datacamp's dplyr lesson
- Mine Çetinkaya-Rundell's rpubs presentation
- R-Ladies Github
- İsmail Sezen's Github

"Data analysis starts with questions, not techniques"

Thank you for joining us…

R-Ladies Istanbul

istanbul@rladies.org
hazel@rladies.org