

# Manipulating Data

Tony Yao-Jen Kuo

## Common data types

## There are 5 essential data types

Python	R
float	numeric or double
str	character
bool	logical
time.struct_time	POSIXct
None	NA or NULL

# Using functions to identify data types

- ▶ Using `type()` in Python
- ▶ Using `class()` or `typeof()` in R

## A quick review on basic data types

# Datetime in Python

```
from time import localtime, strftime  
strftime("%Y-%m-%d %H:%M:%S", localtime())
```

# Datetime in R

```
as.character(Sys.time())
```

## References for `strptime`

- ▶ <https://docs.python.org/3/library/time.html#time.strptime>
- ▶ <https://stat.ethz.ch/R-manual/R-devel/library/base/html/strptime.html>



## Missing value in Python

```
print(type(None))
```

## Missing value in R

```
class(NA)  
class(NULL)
```

## Difference between NA and NULL in R

```
my_vector <- c(1, 2, 3, NA)
sum(my_vector)
my_vector <- c(1, 2, 3, NULL)
sum(my_vector)
```

## Common data structures

## There are 4 essential data structures

Python	R
list	list
dict	named list
ndarray	vector, matrix, array
DataFrame	data.frame

# List in Python

```
my_list = [8.7, True, "Hello Python!"]  
for e in my_list:  
    print(type(e))
```

## List in R

```
my_list <- list(  
  8.7,  
  TRUE,  
  "Hello R!"  
)  
for (e in my_list) {  
  print(class(e))  
}
```

# Dict in Python

```
my_dict = {  
    "my_float": 8.7,  
    "my_bool": True,  
    "my_str": "Hello Python!"  
}
```



## Named list in R

```
my_named_list <- list(  
  my_float = 8.7,  
  my_bool = TRUE,  
  my_str = "Hello Python!"  
)
```

## ndarrays in Python

```
import numpy as np  
my_arr = np.array(range(11, 20))  
print(type(my_arr))
```

## vectors in R

```
my_arr <- 11:19  
class(my_arr)
```

## DataFrame in Python

```
import pandas as pd
numbers = [9, 23, 33, 91, 13]
players = ["Ron Harper", "Michael Jordan", "Scottie Pippen"]
df = pd.DataFrame()
df["number"] = numbers
df["player"] = players
df
```

## data.frame in R

```
numbers <- c(9, 23, 33, 91, 13)
players <- c("Ron Harper", "Michael Jordan", "Scottie Pippen")
df <- data.frame(number = numbers, player = players, stringsAsFactors = FALSE)
View(df)
```

## Manipulating Dataframes

# There are six basic manipulation skills

- ▶ filter
- ▶ select
- ▶ arrange
- ▶ mutate
- ▶ summarise
- ▶ group by

# Dataframes we might use

- ▶ 1995-96 Chicago Bulls roster
- ▶ Lightweight gapminder



# The story behind gapminder data

<https://youtu.be/jbkSRLYSojo>

## Filtering dataframes in Python

```
import pandas as pd
numbers = [9, 23, 33, 91, 13]
players = ["Ron Harper", "Michael Jordan", "Scottie Pippen"]
df = pd.DataFrame()
df["number"] = numbers
df["player"] = players
df.index = ["PG", "SG", "SF", "PF", "C"]
is_trio = df["number"].isin([23, 33, 91])
print(is_trio)
df[is_trio]
```

## Filtering dataframes in R

```
library(dplyr)

numbers <- c(9, 23, 33, 91, 13)
players <- c("Ron Harper", "Michael Jordan", "Scottie Pippen")
df <- data.frame(number = numbers, player = players, stringAsFactors = FALSE)
df %>%
  filter(number %in% c(23, 33, 91))
```

## Selecting from dataframes in Python

```
import pandas as pd
numbers = [9, 23, 33, 91, 13]
players = ["Ron Harper", "Michael Jordan", "Scottie Pippen"]
df = pd.DataFrame()
df["number"] = numbers
df["player"] = players
print(df["player"])
print(type(df["player"]))
df[["player", "number"]]
```

## Selecting from dataframes in R

```
library(dplyr)

numbers <- c(9, 23, 33, 91, 13)
players <- c("Ron Harper", "Michael Jordan", "Scottie Pippen")
df <- data.frame(number = numbers, player = players, stringsAsFactors = FALSE)
df %>%
  select(player, number)
```

## Arranging dataframes in Python

```
import pandas as pd
numbers = [9, 23, 33, 91, 13]
players = ["Ron Harper", "Michael Jordan", "Scottie Pippen"]
df = pd.DataFrame()
df["number"] = numbers
df["player"] = players
print(df.sort_index(ascending=False))
print(df.sort_values(number))
```

## Arranging dataframes in R

```
library(dplyr)

numbers <- c(9, 23, 33, 91, 13)
players <- c("Ron Harper", "Michael Jordan", "Scottie Pippen")
df <- data.frame(number = numbers, player = players, stringsAsFactors = FALSE)
df %>%
  arrange(number)
```

# Mutating dataframes in Python

```
import pandas as pd
numbers = [9, 23, 33, 91, 13]
players = ["Ron Harper", "Michael Jordan", "Scottie Pippen"]
df = pd.DataFrame()
df["number"] = numbers
df["player"] = players
df["last_name"] = df["player"].map(lambda x: x.split()[1])
df
```



## Mutating dataframes in R

```
players <- c("Ron Harper", "Michael Jordan", "Scottie Pippen")
df <- data.frame(number = numbers, player = players, stringsAsFactors = FALSE)
get_last_name <- function(x) {
  split_lst <- strsplit(x, " ")
  name_length <- length(split_lst[[1]])
  last_name <- split_lst[[1]][name_length]
  return(last_name)
}
df$last_name <- sapply(df$player, FUN = get_last_name)
View(df)
```

## Summarising dataframes in Python

```
import pandas as pd
csv_url = "https://storage.googleapis.com/learn_pd_like_tic
df = pd.read_csv(csv_url)
df[df.year == 2007]["pop"].sum()
```

## Summarising dataframes in R

```
library(dplyr)

csv_url <- "https://storage.googleapis.com/learn_pd_like_t
df <- read.csv(csv_url, stringsAsFactors = FALSE)
df %>%
  filter(year == 2007) %>%
  summarise(ttl_pop = sum(as.numeric(pop)))
```

## Grouping values of dataframe in Python

```
import pandas as pd
csv_url = "https://storage.googleapis.com/learn_pd_like_tic
df = pd.read_csv(csv_url)
grouped = df[df.year == 2007].groupby("continent")
grouped["pop"].sum()
```

## Grouping value of dataframe in R

```
library(dplyr)

csv_url <- "https://storage.googleapis.com/learn_pd_like_t
df <- read.csv(csv_url, stringsAsFactors = FALSE)
df %>%
  filter(year == 2007) %>%
  group_by(continent) %>%
  summarise(ttl_pop = sum(as.numeric(pop)))
```

## Summary

# In a nutshell

- ▶ Essential data types
- ▶ Essential data structures
  - ▶ Array-like
  - ▶ Hash-like
  - ▶ Table-like
- ▶ Manipulating table-like dataframes

## Reference



## Further readings

- ▶ <https://www.datainpoint.com/data-science-in-action/06-handling-data-structures.html>
- ▶ <https://www.datainpoint.com/data-science-in-action/07-manipulating-data-frames-basically.html>
- ▶ <https://www.datainpoint.com/data-science-in-action/08-manipulating-data-frames-advancingly.html>