Time-Series data

Teodor Chakarov

2022-04-14

Tutorium in R

Exercise: Time-Series data - Number 6

By: Teodor Chakarov 12141198

Dowload data

```
# Here are the URLs! As you can see they're just normal strings
csv_url <- "http://s3.amazonaws.com/assets.datacamp.com/production/course_1561/datasets/chickwts.csv"
tsv_url <- "http://s3.amazonaws.com/assets.datacamp.com/production/course_3026/datasets/tsv_data.tsv"

# Read a file in from the CSV URL and assign it to csv_data
csv_data <- read.csv(csv_url)

# Read a file in from the TSV URL and assign it to tsv_data
tsv_data <- read.delim(tsv_url)


# Examine the objects with head()
head(csv_data)
```

```
##   weight      feed
## 1    179 horsebean
## 2    160 horsebean
## 3    136 horsebean
## 4    227 horsebean
## 5    217 horsebean
## 6    168 horsebean
```

```
# Download the file with download.file()
download.file(url = csv_url, destfile = "feed_data.csv")

# Read it in with read.csv()
csv_data <- read.csv("feed_data.csv")
```

```
# Add a new column: square_weight
csv_data$square_weight <- csv_data$weight^2

# Save it to disk with saveRDS()
saveRDS(object = csv_data, file = "modified_feed_data.RDS")

# Read it back in with readRDS()
modified_feed_data <- readRDS(file = "modified_feed_data.RDS")
```

```r
# Examine modified_feed_data
str(modified_feed_data)
```

```
## 'data.frame':    71 obs. of  3 variables:
##  $ weight       : int  179 160 136 227 217 168 108 124 143 140 ...
##  $ feed         : chr  "horsebean" "horsebean" "horsebean" "horsebean" ...
##  $ square_weight: num  32041 25600 18496 51529 47089 ...
```

```r
# Load pageviews
library(pageviews)

# Get the pageviews for "Hadley Wickham"
hadley_pageviews <- article_pageviews(project = "en.wikipedia", "Hadley Wickham")

# Examine the resulting object
str(hadley_pageviews)
```

```
## 'data.frame':    1 obs. of  8 variables:
##  $ project    : chr "wikipedia"
##  $ language   : chr "en"
##  $ article    : chr "Hadley_Wickham"
##  $ access     : chr "all-access"
##  $ agent      : chr "all-agents"
##  $ granularity: chr "daily"
##  $ date       : POSIXct, format: "2015-10-01"
##  $ views      : num 53
```

GET and POST request

```r
#Load the httr package
library(httr)

# Make a GET request to http://httpbin.org/get
get_result <- GET("http://httpbin.org/get")

# Print it to inspect it
get_result
```

```
## Response [http://httpbin.org/get]
##   Date: 2022-05-04 18:50
##   Status: 200
##   Content-Type: application/json
##   Size: 368 B
## {
##   "args": {},
##   "headers": {
##     "Accept": "application/json, text/xml, application/xml, */*",
##     "Accept-Encoding": "deflate, gzip",
##     "Host": "httpbin.org",
##     "User-Agent": "libcurl/7.64.1 r-curl/4.3.2 httr/1.4.2",
##     "X-Amzn-Trace-Id": "Root=1-6272cb0a-6e1f62407c645cbc7b59e27c"
##   },
##   "origin": "77.119.210.152",
## ...
```

```r
# Load the httr package
library(httr)

# Make a POST request to http://httpbin.org/post with the body "this is a test"
post_result <- POST("http://httpbin.org/post", body = "this is a test")

# Print it to inspect it
post_result
```

```
## Response [http://httpbin.org/post]
##   Date: 2022-05-04 18:50
##   Status: 200
##   Content-Type: application/json
##   Size: 475 B
## {
##   "args": {},
##   "data": "this is a test",
##   "files": {},
##   "form": {},
##   "headers": {
##     "Accept": "application/json, text/xml, application/xml, */*",
##     "Accept-Encoding": "deflate, gzip",
##     "Content-Length": "14",
##     "Host": "httpbin.org",
## ...
```

```r
url <- "https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/en.wikipedia.org/all-access/all

# Make a GET request to url and save the results
pageview_response <- GET(url)

# Call content() to retrieve the data the server sent back
pageview_data <- content(pageview_response)

# Examine the results with str()
str(pageview_data)
```

```
## List of 1
##  $ items:List of 2
##   ..$ :List of 7
##   .. ..$ project    : chr "en.wikipedia"
##   .. ..$ article    : chr "Hadley_Wickham"
##   .. ..$ granularity: chr "daily"
##   .. ..$ timestamp  : chr "2017010100"
##   .. ..$ access     : chr "all-access"
##   .. ..$ agent      : chr "all-agents"
##   .. ..$ views      : int 45
##   ..$ :List of 7
##   .. ..$ project    : chr "en.wikipedia"
##   .. ..$ article    : chr "Hadley_Wickham"
##   .. ..$ granularity: chr "daily"
##   .. ..$ timestamp  : chr "2017010200"
##   .. ..$ access     : chr "all-access"
##   .. ..$ agent      : chr "all-agents"
```

```
##   .. ..$ views      : int 86
```

Checking url

```
fake_url <- "http://google.com/fakepagethatdoesnotexist"

# Make the GET request
request_result <- GET(fake_url)

# Check request_result
if(http_error(request_result)){
    warning("The request failed")
} else {
    content(request_result)
}
```

```
## Warning: The request failed
```

```
# Construct a directory-based API URL to `http://swapi.co/api`,
# looking for person `1` in `people`
directory_url <- paste("http://swapi.co/api", "people", 1, sep = "/")

# Make a GET call with it
result <- GET(directory_url)
```

Constructing queries

```
# Create list with nationality and country elements
query_params <- list(nationality = "americans",
    country = "antigua")

# Make parameter-based call to httpbin, with query_params
parameter_response <- GET("https://httpbin.org/get", query = query_params)

# Print parameter_response
parameter_response
```

```
## Response [https://httpbin.org/get?nationality=americans&country=antigua]
##   Date: 2022-05-04 18:50
##   Status: 200
##   Content-Type: application/json
##   Size: 468 B
## {
##   "args": {
##     "country": "antigua",
##     "nationality": "americans"
##   },
##   "headers": {
##     "Accept": "application/json, text/xml, application/xml, */*",
##     "Accept-Encoding": "deflate, gzip",
##     "Host": "httpbin.org",
##     "User-Agent": "libcurl/7.64.1 r-curl/4.3.2 httr/1.4.2",
## ...
```

REST API

```
url <- "https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/en.wikipedia/all-access/all-age
```

```r
# Add the email address and the test sentence inside user_agent()
server_response <- GET(url, user_agent("my@email.address this is a test"))
```

```r
# Construct a vector of 2 URLs
urls <- c("http://httpbin.org/status/404", "http://httpbin.org/status/301")

for(url in urls){
    # Send a GET request to url
    result <- GET(url)
    # Delay for 5 seconds between requests
    Sys.sleep(5)
}
```

```r
get_pageviews <- function(article_title){
  url <- paste(
    "https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/en.wikipedia/all-access/all-agents
    # Include article title
    article_title,
    "daily/2015100100/2015103100",
    sep = "/"
  )
  url
}
```

```r
get_pageviews <- function(article_title){
  url <- paste(
    "https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/en.wikipedia/all-access/all-agents
    article_title,
    "daily/2015100100/2015103100",
    sep = "/"
  )
  response <- GET(url, user_agent("my@email.com this is a test"))
  # Is there an HTTP error?
  if(http_error(response)){
    # Throw an R error
    stop("the request failed")
  }
  # Return the response's content
  content(response)
}
```

```r
get_pageviews <- function(article_title){
  url <- paste(
    "https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/en.wikipedia/all-access/all-agents
    article_title,
    "daily/2015100100/2015103100",
    sep = "/"
  )
  response <- GET(url, user_agent("my@email.com this is a test"))
  # Is there an HTTP error?
  if(http_error(response)){
    # Throw an R error
     stop("the request failed")
  }
  response
```

```
}
```

```
get_pageviews <- function(article_title){
  url <- paste(
    "https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/en.wikipedia/all-access/all-agents
    article_title,
    "daily/2015100100/2015103100",
    sep = "/"
  )
  response <- GET(url, user_agent("my@email.com this is a test"))
  # Is there an HTTP error?
  if(http_error(response)){
    # Throw an R error
    stop("the request failed")
  }
  # Return the response's content
  content(response)
}
```

JSON

```
rev_history <- function (title, format = "json") {
    if (title != "Hadley Wickham") {
        stop("rev_history() only works for `title = \"Hadley Wickham\"`")
    }
    if (format == "json") {
        resp <- readr::readRDS("had_rev_json.rds")
    }
    else if (format == "xml") {
        resp <- readr::readRDS("had_rev_xml.rds")
    }
    else {
        stop("Invalid format supplied, try \"json\" or \"xml\"")
    }
    resp
}


# Get revision history for "Hadley Wickham"
#resp_json <- rev_history("Hadley Wickham")

# Check http_type() of resp_json
#http_type(resp_json)

# Examine returned text with content()
#content(resp_json, as = "text")

# Parse response with content()
#content(resp_json, as = "parsed")

# Parse returned text with fromJSON()
library(jsonlite)
#fromJSON(content(resp_json, as = "text"))
```

```
resp_json <- pageview_response

# Check http_type() of resp_json
http_type(resp_json)
```

```
## [1] "application/json"
```

Manipulating JSON

```
url <- "https://en.wikipedia.org/w/api.php?action=query&titles=Hadley%20Wickham&prop=revisions&rvprop=ti

resp_json <- GET(url)
# Load rlist
library(rlist)
str(content(resp_json), max.level = 4)
```

```
## List of 3
##  $ continue:List of 2
##   ..$ rvcontinue: chr "20150528042700|664370232"
##   ..$ continue  : chr "||"
##  $ warnings:List of 2
##   ..$ main     :List of 1
##   .. ..$ *: chr "Subscribe to the mediawiki-api-announce mailing list at <https://lists.wikimedia.org
##   ..$ revisions:List of 1
##   .. ..$ *: chr "Because \"rvslots\" was not specified, a legacy format has been used for the output
##  $ query   :List of 1
##   ..$ pages:List of 1
##   .. ..$ 41916270:List of 4
##   .. .. ..$ pageid   : int 41916270
##   .. .. ..$ ns       : int 0
##   .. .. ..$ title    : chr "Hadley Wickham"
##   .. .. ..$ revisions:List of 5
```

Use list.select() to pull out the user and timestamp elements from each revision, store in user_tim

```
# Store revision list
revs <- content(resp_json)$query$pages$`41916270`$revisions
# Extract the user element
user_time <- list.select(revs, user, timestamp)

# Print user_time
user_time
```

```
## [[1]]
## [[1]]$user
## [1] "214.28.226.251"
##
## [[1]]$timestamp
## [1] "2015-01-14T17:12:45Z"
##
##
## [[2]]
## [[2]]$user
## [1] "73.183.151.193"
```

```
##
## [[2]]$timestamp
## [1] "2015-01-15T15:49:34Z"
##
##
## [[3]]
## [[3]]$user
## [1] "FeanorStar7"
##
## [[3]]$timestamp
## [1] "2015-01-24T16:34:31Z"
##
##
## [[4]]
## [[4]]$user
## [1] "KasparBot"
##
## [[4]]$timestamp
## [1] "2015-04-26T19:18:17Z"
##
##
## [[5]]
## [[5]]$user
## [1] "Spkal"
##
## [[5]]$timestamp
## [1] "2015-05-06T18:24:57Z"
```

```
list.stack(user_time)
```

```
##             user            timestamp
## 1 214.28.226.251 2015-01-14T17:12:45Z
## 2 73.183.151.193 2015-01-15T15:49:34Z
## 3    FeanorStar7 2015-01-24T16:34:31Z
## 4      KasparBot 2015-04-26T19:18:17Z
## 5          Spkal 2015-05-06T18:24:57Z
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Pull out revision list
revs <- content(resp_json)$query$pages$`41916270`$revisions

# Extract user and timestamp
revs %>%
  bind_rows %>%
  select(user, timestamp)
```

```
## # A tibble: 5 x 2
##   user           timestamp
##   <chr>          <chr>
## 1 214.28.226.251 2015-01-14T17:12:45Z
## 2 73.183.151.193 2015-01-15T15:49:34Z
## 3 FeanorStar7    2015-01-24T16:34:31Z
## 4 KasparBot      2015-04-26T19:18:17Z
## 5 Spkal          2015-05-06T18:24:57Z
```

XML Structure

```
library(xml2)

url <- "https://en.wikipedia.org/w/api.php?action=query&titles=Hadley%20Wickham&prop=revisions&rvprop=ti


# Load xml2
library(xml2)

# Get XML revision history
resp_xml <- GET(url)

# Check response is XML
http_type(resp_xml)
```

```
## [1] "text/xml"
```

```
# Examine returned text with content()
rev_text <- content(resp_xml, as="text")

# Turn rev_text into an XML document
rev_xml <- read_xml(rev_text)

# Examine the structure of rev_xml
xml_structure(rev_xml)
```

```
## <api>
##   <continue [rvcontinue, continue]>
##   <warnings>
##     <main [space]>
##       {text}
##     <revisions [space]>
##       {text}
##   <query>
##     <pages>
##       <page [_idx, pageid, ns, title]>
##         <revisions>
##           <rev [user, anon, timestamp, contentformat, contentmodel, comment, space]>
##             {text}
##           <rev [user, anon, timestamp, contentformat, contentmodel, comment, space]>
##             {text}
##           <rev [user, timestamp, contentformat, contentmodel, comment, space]>
##             {text}
##           <rev [user, timestamp, contentformat, contentmodel, comment, space]>
##             {text}
```

```
##           <rev [user, timestamp, contentformat, contentmodel, comment, space]>
##             {text}
```

XPATHS

```
# Find all nodes using XPATH "/api/query/pages/page/revisions/rev"
xml_find_all(rev_xml, "/api/query/pages/page/revisions/rev")
```

```
## {xml_nodeset (5)}
## [1] <rev user="214.28.226.251" anon="" timestamp="2015-01-14T17:12:45Z" conte ...
## [2] <rev user="73.183.151.193" anon="" timestamp="2015-01-15T15:49:34Z" conte ...
## [3] <rev user="FeanorStar7" timestamp="2015-01-24T16:34:31Z" contentformat="t ...
## [4] <rev user="KasparBot" timestamp="2015-04-26T19:18:17Z" contentformat="tex ...
## [5] <rev user="Spkal" timestamp="2015-05-06T18:24:57Z" contentformat="text/x- ...
```

```
# Find all rev nodes anywhere in document
rev_nodes <- xml_find_all(rev_xml, "//rev")

# Use xml_text() to get text from rev_nodes
xml_text(rev_nodes)
```

```
## [1] "'''Hadley Mary Helen Wickham III''' is a  [[statistician]] from [[New Zealand]] who is currentl
## [2] "'''Hadley Wickham''' is a  [[statistician]] from [[New Zealand]] who is currently Chief Scientis
## [3] "'''Hadley Wickham''' is a  [[statistician]] from [[New Zealand]] who is currently Chief Scientis
## [4] "'''Hadley Wickham''' is a  [[statistician]] from [[New Zealand]] who is currently Chief Scientis
## [5] "'''Hadley Wickham''' is a  [[statistician]] from [[New Zealand]] who is currently Chief Scientis
```

Extracting XML attributes

```
# All rev nodes
rev_nodes <- xml_find_all(rev_xml, "//rev")

# The first rev node
first_rev_node <- xml_find_first(rev_xml, "//rev")

# Find all attributes with xml_attrs()
xml_attrs(first_rev_node)
```

```
##                   user                   anon              timestamp
##       "214.28.226.251"                     "" "2015-01-14T17:12:45Z"
##          contentformat          contentmodel                comment
##          "text/x-wiki"            "wikitext"                     ""
##                  space
##             "preserve"
```

```
# Find user attribute with xml_attr()
xml_attr(first_rev_node, "user")
```

```
## [1] "214.28.226.251"
```

```
# Find user attribute for all rev nodes
xml_attr(rev_nodes, "user")
```

```
## [1] "214.28.226.251" "73.183.151.193" "FeanorStar7"    "KasparBot"
## [5] "Spkal"
```

```
# Find anon attribute for all rev nodes
xml_attr(rev_nodes, "anon")
```

```
## [1] "" "" NA NA NA
```

Wrap up:

```
get_revision_history <- function(article_title){
  # Get raw revision response
  rev_resp <- resp_xml

  # Turn the content() of rev_resp into XML
  rev_xml <- read_xml(content(rev_resp, "text"))

  # Find revision nodes
  rev_nodes <- xml_find_all(rev_xml, "//rev")

  # Parse out usernames
  user <- xml_attr(rev_nodes, "user")

  # Parse out timestamps
  timestamp <- readr::parse_datetime(xml_attr(rev_nodes, "timestamp"))

  # Parse out content
  content <- xml_text(rev_nodes)

  # Return data frame
  data.frame(user = user,
    timestamp = timestamp,
    content = substr(content, 1, 40))
}

# Call function for "Hadley Wickham"
get_revision_history(article_title = "Hadley Wickham")
```

```
##             user           timestamp                                  content
## 1 214.28.226.251 2015-01-14 17:12:45 '''Hadley Mary Helen Wickham III''' is a
## 2 73.183.151.193 2015-01-15 15:49:34 '''Hadley Wickham''' is a  [[statisticia
## 3    FeanorStar7 2015-01-24 16:34:31 '''Hadley Wickham''' is a  [[statisticia
## 4      KasparBot 2015-04-26 19:18:17 '''Hadley Wickham''' is a  [[statisticia
## 5          Spkal 2015-05-06 18:24:57 '''Hadley Wickham''' is a  [[statisticia
```

Web scraping with XPATHs

```
# Load rvest
library(rvest)

# Hadley Wickham's Wikipedia page
test_url <- "https://en.wikipedia.org/wiki/Hadley_Wickham"

# Read the URL stored as "test_url" with read_html()
test_xml <- read_html(test_url)

# Print test_xml
test_xml
```

```
## {html_document}
## <html class="client-nojs" lang="en" dir="ltr">
```

```
## [1] <head>\n<meta http-equiv="Content-Type" content="text/html; charset=UTF-8 ...
## [2] <body class="mediawiki ltr sitedir-ltr mw-hide-empty-elt ns-0 ns-subject  ...
```

```
test_node_xpath <- "//*[contains(concat( \" \", @class, \" \" ), concat( \" \", \"vcard\", \" \" ))]"
test_node_xpath
```

```
## [1] "//*[contains(concat( \" \", @class, \" \" ), concat( \" \", \"vcard\", \" \" ))]"
```

```
# Use html_node() to grab the node with the XPATH stored as `test_node_xpath`
node <- html_node(x = test_xml, xpath = test_node_xpath)

# Print the first element of the result
node[1]
```

```
## $node
## <pointer: 0x000000001287fb90>
```

```
# Extract the name of node
element_name <- html_name(node)

# Print the name
element_name
```

HTML structure

```
## [1] "table"
```

```
second_xpath_val <- "//*[contains(concat( \" \", @class, \" \" ), concat( \" \", \"fn\", \" \" ))]"
```

```
# Extract the element of table_element referred to by second_xpath_val and store it as page_name
page_name <- html_node(x = test_xml, xpath = second_xpath_val)

# Extract the text from page_name
page_title <- html_text(page_name)

# Print page_title
page_title
```

```
## [1] "Hadley Wickham"
```

```
# Turn table_element into a data frame and assign it to wiki_table
wiki_table <- html_table(node)

# Print wiki_table
wiki_table
```

Reforming data

```
## # A tibble: 12 x 2
##    `Hadley Wickham`         `Hadley Wickham`
##    <chr>                    <chr>
##  1 "Hadley Wickham in 2015" "Hadley Wickham in 2015"
##  2 "Born"                   "Hadley Alexander Wickham (1979-10-14) 14 October 1~
##  3 "Alma mater"             "University of Auckland (BSc, MSc)Iowa State Univer~
##  4 "Known for"              "ggplot2[1]tidyverseR packages"
##  5 "Awards"                 "COPSS Presidents' Award (2019)\nFellow of the Amer~
```

```
##  6 "Scientific career"      "Scientific career"
##  7 "Fields"                 "Data science\nVisualization\nStatistics[2]"
##  8 "Institutions"           "RStudio Inc.\nUniversity of Auckland\nStanford Uni~
##  9 "Thesis"                 "Practical tools for exploring data and models (200~
## 10 "Doctoral advisors"      "Dianne Cook\nHeike Hofmann[3]"
## 11 ""                       ""
## 12 "Website"                "hadley.nz"
```

```
# Rename the columns of wiki_table
colnames(wiki_table) <- c("key", "value")

# Remove the empty row from wiki_table
cleaned_table <- subset(wiki_table, !key == "")

# Print cleaned_table
cleaned_table
```

```
## # A tibble: 11 x 2
##    key                    value
##    <chr>                  <chr>
##  1 Hadley Wickham in 2015 "Hadley Wickham in 2015"
##  2 Born                   "Hadley Alexander Wickham (1979-10-14) 14 October 197~
##  3 Alma mater             "University of Auckland (BSc, MSc)Iowa State Universi~
##  4 Known for              "ggplot2[1]tidyverseR packages"
##  5 Awards                 "COPSS Presidents' Award (2019)\nFellow of the Americ~
##  6 Scientific career      "Scientific career"
##  7 Fields                 "Data science\nVisualization\nStatistics[2]"
##  8 Institutions           "RStudio Inc.\nUniversity of Auckland\nStanford Unive~
##  9 Thesis                 "Practical tools for exploring data and models (2008)"
## 10 Doctoral advisors      "Dianne Cook\nHeike Hofmann[3]"
## 11 Website                "hadley.nz"
```

CSS Web Scraping

```
# Select the table elements
html_nodes(test_xml, css = "table")
```

```
## {xml_nodeset (3)}
## [1] <table class="infobox biography vcard"><tbody>\n<tr><th colspan="2" class ...
## [2] <table class="nowraplinks mw-collapsible autocollapse navbox-inner" style ...
## [3] <table class="nowraplinks hlist mw-collapsible autocollapse navbox-inner" ...
```

```
# Select elements with class = "infobox"
html_nodes(test_xml, css = ".infobox")
```

```
## {xml_nodeset (1)}
## [1] <table class="infobox biography vcard"><tbody>\n<tr><th colspan="2" class ...
```

```
# Select elements with id = "firstHeading"
html_nodes(test_xml, css = "#firstHeading")
```

```
## {xml_nodeset (1)}
## [1] <h1 id="firstHeading" class="firstHeading mw-first-heading">Hadley Wickha ...
```

```
# Extract element with class infobox
infobox_element <- html_nodes(test_xml, css = ".infobox")
```

```
# Get tag name of infobox_element
element_name <- html_name(infobox_element)

# Print element_name
element_name
```

```
## [1] "table"
```

```
# Extract element with class fn
page_name <- html_node(x = infobox_element, ".fn")

# Get contents of page_name
page_title <- html_text(page_name)

# Print page_title
page_title
```

```
## [1] "Hadley Wickham"
```

Case Study Introduction

```
# Load httr
library(httr)

# The API url
base_url <- "https://en.wikipedia.org/w/api.php"

# Set query parameters
query_params <- list(action = "parse",
  page = "Hadley Wickham",
  format = "xml")

# Get data from API
resp <- GET(url = base_url, query = query_params)

# Parse response
resp_xml <- content(resp)
```

```
# Read page contents as HTML
page_html <- read_html(xml_text(resp_xml))

# Extract infobox element
infobox_element <- html_node(page_html, ".infobox")

# Extract page name element from infobox
page_name <- html_node(infobox_element, ".fn")

# Extract page name as text
page_title <- html_text(page_name)
```

```
# Your code from earlier exercises
wiki_table <- html_table(infobox_element)
colnames(wiki_table) <- c("key", "value")
cleaned_table <- subset(wiki_table, !key == "")
```

```
# Create a dataframe for full name
name_df <- data.frame(key = "Full name", value = page_title)

# Combine name_df with cleaned_table
wiki_table2 <- rbind(name_df, cleaned_table)

# Print wiki_table
wiki_table2
```

```
##                       key
## 1               Full name
## 2  Hadley Wickham in 2015
## 3                    Born
## 4              Alma mater
## 5               Known for
## 6                  Awards
## 7       Scientific career
## 8                  Fields
## 9            Institutions
## 10                 Thesis
## 11      Doctoral advisors
## 12                Website
##                                                                                                  value
## 1                                                                                        Hadley Wickham
## 2                                                                                  Hadley Wickham in 2015
## 3    Hadley Alexander Wickham (1979-10-14) 14 October 1979 (age 42)Hamilton, New Zealand
## 4                             University of Auckland (BSc, MSc)Iowa State University (PhD)
## 5                                                                  ggplot2[1]tidyverseR packages
## 6  COPSS Presidents' Award (2019)\nFellow of the American Statistical Association (2015)
## 7                                                                          Scientific career
## 8                                                 Data science\nVisualization\nStatistics[2]
## 9              RStudio Inc.\nUniversity of Auckland\nStanford University\nRice University
## 10                                    Practical tools for exploring data and models (2008)
## 11                                                 Dianne Cook\nHeike Hofmann[3]
## 12                                                                               hadley.nz
```

```
library(httr)
library(rvest)
library(xml2)

get_infobox <- function(title){
  base_url <- "https://en.wikipedia.org/w/api.php"

  # Change "Hadley Wickham" to title
  query_params <- list(action = "parse",
    page = title,
    format = "xml")

  resp <- GET(url = base_url, query = query_params)
  resp_xml <- content(resp)

  page_html <- read_html(xml_text(resp_xml))
  infobox_element <- html_node(x = page_html, css =".infobox")
  page_name <- html_node(x = infobox_element, css = ".fn")
```

```
  page_title <- html_text(page_name)

  wiki_table <- html_table(infobox_element)
  colnames(wiki_table) <- c("key", "value")
  cleaned_table <- subset(wiki_table, !wiki_table$key == "")
  name_df <- data.frame(key = "Full name", value = page_title)
  wiki_table <- rbind(name_df, cleaned_table)

  wiki_table
}

# Test get_infobox with "Hadley Wickham"
get_infobox(title = "Hadley Wickham")
```

```
##                       key
## 1               Full name
## 2  Hadley Wickham in 2015
## 3                    Born
## 4              Alma mater
## 5               Known for
## 6                  Awards
## 7       Scientific career
## 8                  Fields
## 9            Institutions
## 10                 Thesis
## 11      Doctoral advisors
## 12                Website
##                                                                                 value
## 1                                                                      Hadley Wickham
## 2                                                                 Hadley Wickham in 2015
## 3    Hadley Alexander Wickham (1979-10-14) 14 October 1979 (age 42)Hamilton, New Zealand
## 4                           University of Auckland (BSc, MSc)Iowa State University (PhD)
## 5                                                  ggplot2[1]tidyverseR packages
## 6  COPSS Presidents' Award (2019)\nFellow of the American Statistical Association (2015)
## 7                                                                  Scientific career
## 8                                          Data science\nVisualization\nStatistics[2]
## 9            RStudio Inc.\nUniversity of Auckland\nStanford University\nRice University
## 10                                Practical tools for exploring data and models (2008)
## 11                                                 Dianne Cook\nHeike Hofmann[3]
## 12                                                                        hadley.nz
```

```
# Try get_infobox with "Ross Ihaka"
get_infobox(title = "Ross Ihaka")
```

```
##                                          key
## 1                                  Full name
## 2  Ihaka at the 2010 New Zealand Open Source Awards
## 3                                       Born
## 4                                 Alma mater
## 5                                  Known for
## 6                                     Awards
## 7                           Scientific career
## 8                                     Fields
## 9                                Institutions
```

```
## 10                                          Thesis
## 11                                Doctoral advisor
## 12                                         Website
##                                                                    value
## 1                                                             Ross Ihaka
## 2                Ihaka at the 2010 New Zealand Open Source Awards
## 3            George Ross Ihaka1954 (age 67-68)Waiuku, New Zealand
## 4  University of AucklandUniversity of California, Berkeley (PhD)
## 5                                          R programming language
## 6                                          Pickering Medal (2008)
## 7                                               Scientific career
## 8                                           Statistical computing
## 9                                          University of Auckland
## 10                                                Ruaumoko (1985)
## 11                                         David R. Brillinger[1]
## 12                                www.stat.auckland.ac.nz/~ihaka/
```

```
# Try get_infobox with "Grace Hopper"
get_infobox(title = "Grace Hopper")
```

```
##                     key
## 1             Full name
## 2  Photograph from 1984
## 3            Birth name
## 4                  Born
## 5                  Died
## 6       Place of burial
## 7            Allegiance
## 8        Service/branch
## 9      Years of service
## 10                 Rank
## 11               Awards
## 12           Alma mater
##
## 1
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11 Defense Distinguished Service Medal Legion of Merit Meritorious Service Medal American Campaign M
## 12
```