# Data Collecting
# – Web scraping  from public websites

Jung PARK, PhD
Research Fellow in Data Science

Version 2018 Aug 09

# Data Collecting

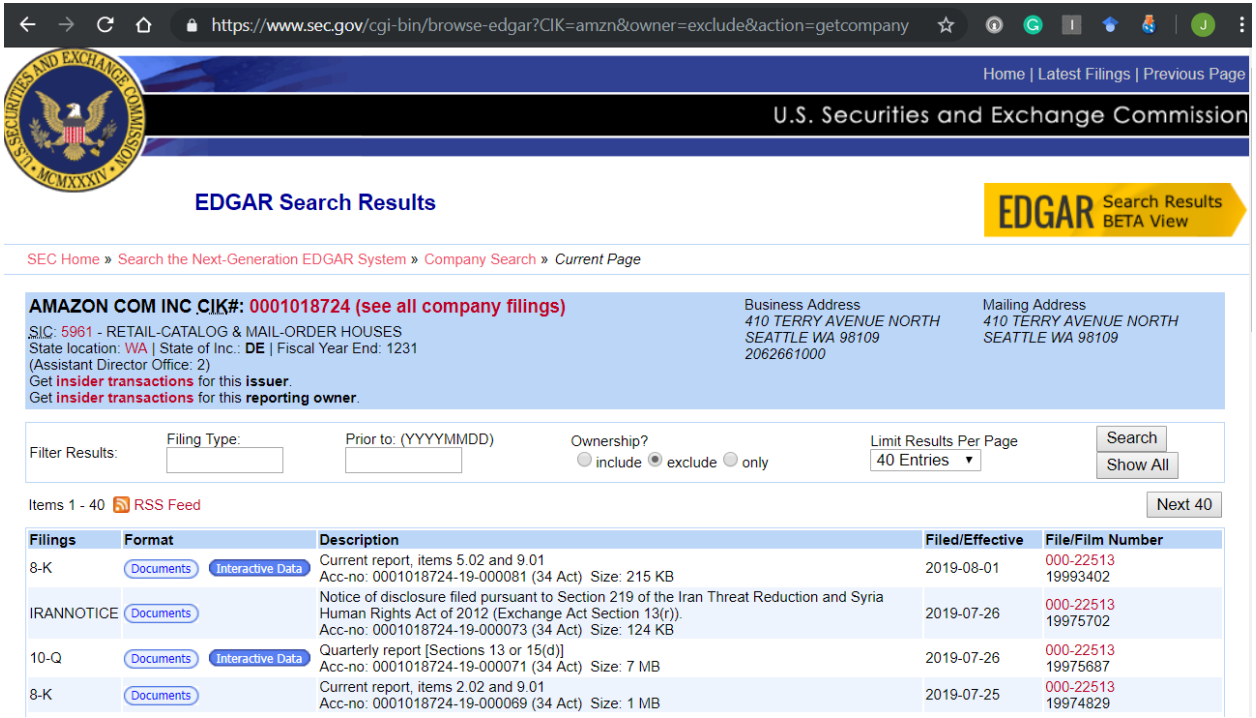
- Web scraping  from public websites
  - How to collect data from websites
  - 10K reports from SEC.gov
  - EU Open Data
  - Open data Swiss
- Collecting data using API
  - Tweeter
  - Facebook
- Database in IMD library
  - Bloomberg
  - Thomson One
  - Datastream
  - Factiva: news media
- IoT (Internet of Things)
  - Smart building
  - Wearable devices
  - Web traffic

# Data collecting approaches

- Level 1. Manual collection
- Level 2. Using a code to download multiple files
- Level 3. Using API (Application Programming Interface)
- Level 4. Using a code for Web scraping

# Level 1: manual collection
## ex. Company filings from SEC.gov

- https://www.sec.gov/edgar/searchedgar/companysearch.html
- Mouse right-click and download individual files




- Easy to do; can check what we get immediately
- Boring and time-consuming
- We used to hire interns to do so

# Level 2: Using a code to download multiple files
# ex. Company filings from SEC.gov

```
import requests
def download (url):
    path = url.split("/")[-1]
    r = requests.get(url)
    open(path, 'wb').write(r.content)

urls =
['https://www.sec.gov/Archives/edgar/data/320193/0000
320193-19-000026.txt',
'https://www.sec.gov/Archives/edgar/data/320193/00011
93125-19-004664.txt']

for url in urls:
    download (url)
```

Define a function for downloading

Use the same file name obtained from url

Get the file indicated by the url

Write the content of the file indicated by the url as the name defined in path

Set url addresses (ex. from SEC.gov)

Call the download function repeatedly

# Level 2: Using a code to download multiple files
# ex. Company filings from SEC.gov

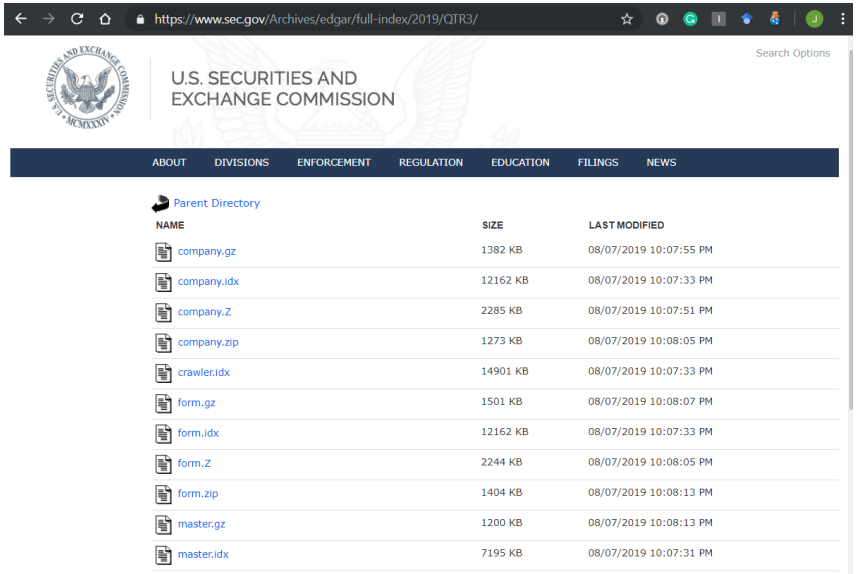
SEC.gov explains how to access files from their server:
https://www.sec.gov/edgar/searchedgar/accessing-edgar-data.htm

For example, we can use the file locations written in the master index provided by SEC.gov:
https://www.sec.gov/Archives/edgar/full-index/2019/QTR1/master.idx

master.idx

Description: Master Index of EDGAR Dissemination FeedLast Data Received: March 31, 2019Comments: webmaster@sec.govAnonymous FTP: ftp://ftp.sec.gov/edgar/Cloud HTTP: https://www.sec.gov/Archives/ CIK|Company Name|Form Type|Date Filed|Filename----------
---

1000045|NICHOLAS FINANCIAL INC|10-Q|2019-02-14|edgar/data/1000045/0001193125-19-039489.txt
1000045|NICHOLAS FINANCIAL INC|4|2019-01-15|edgar/data/1000045/0001357521-19-000001.txt
1000045|NICHOLAS FINANCIAL INC|4|2019-02-19|edgar/data/1000045/0001357521-19-000002.txt
1000045|NICHOLAS FINANCIAL INC|4|2019-03-15|edgar/data/1000045/0001357521-19-000003.txt
1000045|NICHOLAS FINANCIAL INC|8-K|2019-02-01|edgar/data/1000045/0001193125-19-024617.txt
1000045|NICHOLAS FINANCIAL INC|SC 13G/A|2019-02-04|edgar/data/1000045/0001104659-19-005360.txt
1000045|NICHOLAS FINANCIAL INC|SC 13G/A|2019-02-08|edgar/data/1000045/0001258897-19-001312.txt
1000045|NICHOLAS FINANCIAL INC|SC 13G/A|2019-02-11|edgar/data/1000045/0001019056-19-000082.txt



# Level 3. Using API (Application Programming Interface)

- API is a standardized way for external users to collect data from a website's database
- Some websites like wikipedia provide API to encourages developing a third-party software providing additional services based on the company's product
- Also, this can avoid the abuse of data access using a risky method to the server

- reference: https://www.mediawiki.org/wiki/API:Main_page

# Level 3. Using API (Application Programming Interface)
# ex. R code for collecting table information from Wikipedia

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

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

  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
}

get_infobox(title = "International Institute for Management Development")
```

Call some packages

Define a function get_infobox

We will use wikipedia api

Send query parameters and receive data

Rearrange the data into a table form

Call the function for “IMD”

# Level 3. Using API (Application Programming Interface)
# ex. R code for collecting table information from Wikipedia

## Output

| | key | value |
|---|---|---|
| 1 | Full name | IMD |
| 2 | Logo of IMD | Logo of IMD |
| 3 | Type | Private Business School |
| 4 | Established | 1990 |
| 5 | President | Jean-François Manzoni |
| 6 | Academic staff | 50 |
| 7 | Administrative staff | 300 |
| 8 | Students | 8,900 |
| 9 | Other students | 90 |
| 10 | Location | Lausanne, Vaud, Switzerland46°31'N 6°37'E<U+FEFF> / <U+... |
| 11 | Campus | Urban |
| 12 | Website | http://www.imd.org/ |
| 13 | Alumni: 100,000 | Alumni: 100,000 |

# Level 4. Using a code for Web scraping

Even though there is no API provided, it is possible to extract texts from any websites using R and Python.

[to be updated]

# References and IMD Champions

**10K and proxy statements from SEC.gov and context analysis**
- Abraham Lu at IMD Global Board Center (abraham.lu@imd.org)

**Webscribing codes**
https://www.datacamp.com/courses/working-with-web-data-in-r

# Public data sources (examples to be updated)

- Data from public websites
  - SEC.gov
  - EU Open Data
  - Open data Swiss
  - London Data Store
  - Wikipedia

- Data from commercial websites
  - Tweeter
  - Amazon