ISA 401: Business Intelligence & Data Visualization

07: Connecting to APIs in R

Fadel M. Megahed, PhD

Associate Professor
Department of Information Systems and Analytics
Farmer School of Business
Miami University

Twitter: FadelMegahed
GitHub: fmegahed
Email: fmegahed@miamioh.edu
Office Hours: Automated Scheduler for Virtual Office Hours

Spring 2022

Quick Refresher from Last Week

- ✅ Understand when can we scrape data (i.e., `robots.txt`)
- ✅ Scrape a webpage Using R
- ✅ Utilize loops or `purr::map` to download data from multiple webpages.

Learning Objectives for Today's Class

- Describe what is an API
- Download data using APIs

What is an API?

What is an API? [1]

- An **API** is an acronym for application programming interface.
- It is a **popular** approach to interact with an application/service or data since it:
  - Defines a set of functionalities independent of implementation (i.e., it only exposes information that a programmer might find useful and keep those parts consistent even if the implementation changes later)
  - Provides some level of privacy/control over one's internal data and the rate at which it can be accessed.

What is an API? [2]

What is an API? [3]

Scenario: Alone, you went into a warehouse and are trying to retrieve 3 screwdrivers, a toolbox, and 15 phillips screws. But you do not know, where those things are in the warehouse.

The API is the set of instructions provided by the warehouse manager on where/how to retrieve this information without touching/accessing other things in there.

Source: Matt Z. (2018). What is an API? (explanation with cartoon picture so a 5 year old could understand it)

Accessing APIs in R

The 3 Step Process

Before you dive into the API documentation, you **should first check if there is a R (or Python if you are familiar)** package/library that serves as a wrapper for that API.

- **Find** the **API's documentation** and find information about the following:
  A. Does the API require an **authentication key**?
  B. What are the API's **base URL** and **query parameters**?
  C. How does the request URL look like?

2.Craft your **request**. My recommendations are to:
A. First, start with a simple request.
B. Test that request in your browser and see what results you get.

- Construct that request in R by **either**: A. If the generated content seems to be a `JSON` file/webpage, you can capitalize on the reading the content from `jsonlite::fromJSON()`; **OR**

Demo 1: `tidycensus` vs Census API

Motivation | Learning Objectives | tidycensus Results | Direct Results

In socio-economic analysis, we are often interested in examining explanatory population-level variables. For the U.S., the decennial (once every 10 ten years) Census, and the 1-year and 5-year American Community Surveys are often the gold-standard for such data.

Luckily, the tidycensus is an R package that allows users to interface with a select number of the US Census Bureau’s data APIs and return tidyverse-ready data frames.

Demo 1: `tidycensus` vs Census API

Motivation | Learning Objectives | tidycensus Results | Direct Results

> In this demo, we will:
>
> - Set up an API key for the Census API
> - Use the `tidycensus` package to obtain the total population for Butler and Warren Counties in Ohio from https://api.census.gov/data/2020/dec/pl/variables.html
> - Extract the same data by capitalizing on the API itself (i.e., without the tidycensus package)

Demo 1: `tidycensus` vs Census API

Motivation   Learning Objectives   tidycensus Results   Direct Results

> In class, we will live code and capitalize on the `tidycensus` package to get the total population for Butler and Warren Counties in Ohio from https://api.census.gov/data/2020/dec/pl/variables.html

```
## # A tibble: 2 x 4
##   GEOID NAME                    variable  value
##   <chr> <chr>                   <chr>     <dbl>
## 1 39017 Butler County, Ohio P1_001N  390357
## 2 39165 Warren County, Ohio P1_001N  242337
```

Demo 1: `tidycensus` vs Census API

Motivation Learning Objectives tidycensus Results **Direct Results**

> In class, we will live code and capitalize on the `tidycensus` package to get the total population for Butler and Warren Counties in Ohio from https://api.census.gov/data/2020/dec/pl/variables.html

```
## # A tibble: 2 x 3
##   P1_001N state county
##   <chr>   <chr> <chr> 
## 1 390357  39    017   
## 2 242337  39    165   
```

Demo 2: Accuweather API

Demo Description | Code and Results

> - Go to https://developer.accuweather.com/ and create an account.
> - Add your first app (from the `MY APPS` tab) and copy the generated API key.
> - Then using the API Reference Tab → Locations API → City Search → find the location key for `Oxford, Ohio`
> - Use this information in the `Forecast API` to obtain the `5 Day Forecasts` for `Oxford Ohio`.

Demo 2: Accuweather API

Demo Description   Code and Results

```
## List of 2
##  $ Headline      :List of 9
##   ..$ EffectiveDate     : chr "2022-02-16T19:00:00-05:00"
##   ..$ EffectiveEpochDate: int 1645056000
##   ..$ Severity          : int 3
##   ..$ Text              : chr "Rain from Wednesday evening until Thursday evening, when it will cha
##   ..$ Category          : chr "snow"
##   ..$ EndDate           : chr "2022-02-18T01:00:00-05:00"
##   ..$ EndEpochDate      : int 1645164000
##   ..$ MobileLink        : chr "http://www.accuweather.com/en/us/oxford-oh/45056/daily-weather-forec
##   ..$ Link              : chr "http://www.accuweather.com/en/us/oxford-oh/45056/daily-weather-forec
##  $ DailyForecasts:'data.frame':    5 obs. of  8 variables:
##   ..$ Date       : chr [1:5] "2022-02-13T07:00:00-05:00" "2022-02-14T07:00:00-05:00" "2022-02-15T07
##   ..$ EpochDate  : int [1:5] 1644753600 1644840000 1644926400 1645012800 1645099200
##   ..$ Temperature:'data.frame':    5 obs. of  2 variables:
##   .. ..$ Minimum:'data.frame':    5 obs. of  3 variables:
##   .. ..$ Maximum:'data.frame':    5 obs. of  3 variables:
##   ..$ Day        :'data.frame':    5 obs. of  5 variables:
##   .. ..$ Icon                 : int [1:5] 19 3 3 6 18
##   .. ..$ IconPhrase           : chr [1:5] "Flurries" "Partly sunny" "Partly sunny" "Mostly cloudy"
```

Demo 3: The CryptoCompare API

Demo Description Code and Results

- Create a Personal (Free) account at CryptoCompare.com
- Click on create your free key to create your API key and copy the key.
- Go to the documentation, and test their sample call by executing the call after you have pasted your API key in the call.
  - The executed call returns the price of BTC (Bitcoin) in USD, JPY and EUR.
- Now click on the Historical Data Tab on the left
  - Click on Daily Pair OHLCV and Execute the Sample Call for BTC
  - This returns 10 days worth of OHLCV for BTC in USD.
- Let us obtain the price for $SHIB over the past 100 days.

Demo 3: The CryptoCompare API

Demo Description

Code and Results

```
##            time      high       low      open     close
## 96  2022-02-08 3.500e-05 2.941e-05 3.258e-05 3.100e-05
## 97  2022-02-09 3.409e-05 3.044e-05 3.100e-05 3.298e-05
## 98  2022-02-10 3.373e-05 3.053e-05 3.298e-05 3.085e-05
## 99  2022-02-11 3.239e-05 2.789e-05 3.085e-05 2.871e-05
## 100 2022-02-12 2.985e-05 2.747e-05 2.871e-05 2.915e-05
## 101 2022-02-13 3.217e-05 2.839e-05 2.915e-05 3.048e-05
```

Recap

Summary of Main Points

By now, you should be able to do the following:

- Describe what is an API
- Download data using APIs

Supplementary Reading on Accessing APIs

- Getting Started with httr
- Managing secrets