Final Project

Swarnima Sircar

11 April 2019

Geocoding

Geocoding is simply the process of providing geographical coordinates corresponding to a location. Let us say that we want to overlay **some visualisations of x on Barcelona on a map of the city**(this content depends on our exercises). We have used country codes to identify countries before, but since no standard equivalent exists for cities, we need its latitude and longitude.

There are two ways to geocode an address, the first, by using its latitude and longitude, and the second: by using a character string. The latter uses the Google Maps API to geocode the string to a longitude and latitude (akin to what happens when type a location in Google Maps' search bar). Let us try to do this. Coincidentally, the `get_map` function in the `ggmap` package can help us get base maps for places

```
library(ggmap)
get_map("Barcelona")
```

However, we get the following error:

```
Error: Google now requires an API key.
       See ?register_google for details.
```

Getting your Google Maps API key

As the error message explaines, before we can start geocoding, we need to obtain an API key from Google. An API key is the name given to the secret token Google issues to you so that you can use its web service (the Google Maps API).

Note that we are interacting with propietary and paid content for the first time in this class. The Google Maps API is not a free or open-source service. There is a free allowance of 40,000 calls to the geocoding API per month, however, once you cross the 40,000 limit, each query to the API costs $0.005 USD. You will be required to provide your billing information even if you plan only to use the free service. For the purposes of this class, *please be careful* so that you do not incur execessive costs.

Now, let us actually get the the API key so that we can start geocoding.

Back to Geocoding

We have already created chloropleth maps of the world. `ggplot2` itself has inbuilt maps of the continental United States. Why did we bother with the tedious process of installing and making sure that we are able to use `ggmap`?

Using R's or R packages' built-in datasets is certainly extremely convenient. However, these data are generally limited to United States. One of the main strengths of `ggmap` lies in its ability to get granular spatial data for *other* countries as well. The `geocode` function from the `ggmap` package extracts longitude and latitude data from Google Maps API, based on a location query. The Google Maps database is one of the most (if not the most) comprehensive repository of spatial data in the world. Apart from **geocoding**, `ggmap` also includes functions for **distance calculations** and **calculating routes**. It is also useful if we want to use a base map that has somewhat "unusual" layers, such as "satellite", and "roads".

Let us first load `ggmap` and our API key into this session. The `register_google` function stores the Maps API key (which is stored in the text file called "google-api.txt").

```
library(ggmap)
api <- readLines("google-api.txt") # Text file with the API key
register_google(key = api)
# getOption("ggmap")
```

With `ggmap` loaded and our API key in hand, let us begin.

```
geocode("Barcelona")
```

It works! We have just successfully made a query to the Google Maps API. This gives us the longitude and latitude data for the center of Barecelona. We now have a single latitude and longitude data point. We could put this point on a map.

```
Do that here.
```

We can extend this. Let us plot all the places first-year students at Yale-NUS visited in 2018 during Week 7 of their first semester.

```
# library(tidyverse)
# library(ggmap)
# api <- readLines("google-api.txt") # Text file with the API key
# register_google(key = api)
# getOption("ggmap")
locations <- c("Johannesburg", "Oxford", "Denpasar Bali",
               "Kraków", "Bochnia Salt Mine", "Phnom Penh",
               "Hong Kong", "Dehradun", "Singapore", "Penang", "Bali") %>%
    geocode()

world <- map_data("world")
ggplot() +
    geom_polygon(data = world,  aes(long, lat, group = group), fill = "grey") +
    geom_point(data = locations, aes(lon, lat), colour = "red", size = 5) +
    coord_map("ortho", orientation = c(30, 80, 0)) +
    theme_void()
```

But, as we stated in the beginning of this section, we want a *map* of Barcelona, not just its location on a map. The `ggmap` package has a convenient `get_map` function that we can use to get base maps for locations. There are two ways to geocode an address, the first, by using its latitude and longitude, and the second: by using a character string. The latter uses the Google Maps API to geocode the string to a longitude and latitude (akin to what happens when type a location in Google Maps' search bar).

```
get_map("Barcelona", zoom = 10)
```

It works! We also note that one can use the `zoom` parameter to set the degree to which we want the map magnified. The value may be set to a double between 3 and 20, where lower values mean that the map is more zoomed out, and higher values mean a higher degree of magnification.