Geocoding Wineries

YZ Analytics

Why Geocode?

Geocoding locations is generally a good idea because it allows for spatial analysis and spatial visualizations. In our case, geocoding the wine reviews will allow us to create interactive maps that visualize the wineries and allows users to explore our dataset.

Using ggmap

The most convenient approach to perform geocoding in R is to use the `ggmap` package. However, the rather recent change in Google's API requires setting up a project and generating a key in the Google Cloud Project. While we are billed for every observation that we geocode, we have credits available each month. To find out more about the API usage and billing refer to the official website. To learn more about the `ggmap` package check out the project's Github repository.

Reading in the Data

```
library(ggmap)
library(dplyr)
library(readr)
library(purrr)
```

```
data_dir <- "../data"
file_name <- "winemag-data-130k-v2.csv"

path <- file.path(data_dir, file_name)
```

```
Wine <- read_csv(path) %>%
    rename(id = X1) %>%
    mutate(id = id + 1)
```

The `mutate_geocoded` function from `ggmap` would be great if it actually had some error handling. However, it is not robust enough for our needs. Instead, we will create our own function to handle both network and API errors, while ensuring completion of our code.

We will operate in a small sample of observations. This will allow us to test our function and then apply it to the whole dataset.

```
set.seed(2019)

subset <- Wine %>%
    count(winery, country) %>%
    mutate(address = paste0(winery, " ", country)) %>%
    sample_n(20)
```

Setting Up Helper Function

We mentioned that the function we use to geocode our observations has to be robust. We take advantage of the `purrr` adverbs to handle internal messages that slow down the operation of the geocoding function in `ggmap`. Additionally, we prevent network failures from being an issue by allowing failed requests to retry at most once after a short delay. Finally, in order to be able to ensure the completion of our code without errors, we wrap the geocoding functionality with an alternative for when the function fails.

You can see all these components working together in the function below.

```
geocode_robustly <-
    possibly(
        insistently(
            quietly(geocode),
            rate = rate_delay(0.1, max_times = 2)),
        otherwise = list(result = tibble(lon = NA_real_, lat = NA_real_))
    )
```

It is important to make sure that the `otherwise` argument matches the type of output given by the function that is wrapped by the `possibly` adverb.

Finally, we apply our function to the addresses of the observations.

```
locations <- subset %>%
    pull(address) %>%
    map_dfr(~ geocode_robustly(.x)$result)

subset %>% bind_cols(locations)
## # A tibble: 20 x 6
##    winery              country     n address                     lon   lat
##    <chr>               <chr>   <int> <chr>                     <dbl> <dbl>
##  1 Sassy Bitch         Chile       3 Sassy Bitch Chile        -83.8   32.5
##  2 Fratelli Casetta    Italy       9 Fratelli Casetta Italy     8.01  44.8
##  3 Château de l'Hyver~ France      1 Château de l'Hyvernièr~   -1.35  47.2
##  4 Chanin              US         22 Chanin US                  NA     NA
##  5 Antonopoulos        Greece      1 Antonopoulos Greece       23.7   38.0
##  6 Dr. Leimbrock       Germany     5 Dr. Leimbrock Germany      7.01  49.9
##  7 Estrella Creek      US          1 Estrella Creek US       -121.    35.7
##  8 Macchia             US          6 Macchia US                 NA     NA
##  9 Shannon             US          4 Shannon US               -86.5   36.4
## 10 Ponte               US          3 Ponte US                 -74.0   40.6
## 11 Point Concepción    US          1 Point Concepción US     -120.    34.4
## 12 Inception           US         15 Inception US             -95.7   37.1
## 13 Château Rahoul      France     12 Château Rahoul France     -0.426 44.7
## 14 Quinta dos Murças   Portug~    19 Quinta dos Murças Port~   -7.69  41.2
## 15 Cremaschi Furlotti  Chile       8 Cremaschi Furlotti Chi~ -71.7  -35.6
## 16 Michel Moritz       France      1 Michel Moritz France       NA     NA
## 17 Hirsch              US         21 Hirsch US                -79.6   41.3
## 18 Arnaud de Villeneu~ France      3 Arnaud de Villeneuve F~    3.85  43.6
## 19 Domaine de Lischet~ France      1 Domaine de Lischetto F~   10.8   43.4
## 20 Goretti             Italy       7 Goretti Italy             12.7   41.5
```

Note that there will be some `NA` values in our location variables because we are largely relying on the integrity of the dataset and the Google Maps search engine.

Confirming that our function is operating as desired, we can now we geocode all the units in our dataset.

Putting it all Together

In the code below we perform a further optimization by only geocoding the set of wineries. This allows us to avoid performing slow network requests on observations that have already been geocoded. The following represents every step taken to geocode our dataset with more than 100K observations. (The code will take some time to run.)

```r
# 1. Add address column to geocode
Wine <- Wine %>%
    mutate(address = paste0(winery, " ", country))

# 2. Get unique addresses
Addresses <- Wine %>%
    count(address) %>%
    select(-n)

# 3. Geocode unique addresses
Locations <- Addresses %>%
    pull(address) %>%
    map_dfr(~ geocode_robustly(.x)$result)

# 4. Bind location info to addresses
Addresses <- Addresses %>%
    bind_cols(Locations)

# 5. Join into original dataset
Geocoded_Wine <- Wine %>%
    left_join(Addresses, by = "address")

# 4. Save geocoded dataset
file_name <- "geocoded.csv.gz"
Geocoded_Wine %>% write_csv(path = file.path(data_dir, file_name))
```

Now we have a geocoded version of the dataset that we can further refine for our analysis.

Verify

To prove that our geocoding worked we can import our new dataset.

```r
Wine <- read_csv("../data/geocoded.csv.gz") %>%
    glimpse()
## Observations: 129,971
## Variables: 17
## $ id                    <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 1...
## $ country               <chr> "Italy", "Portugal", "US", "US", "US", "...
## $ description           <chr> "Aromas include tropical fruit, broom, b...
## $ designation           <chr> "Vulkà Bianco", "Avidagos", NA, "Reserve...
## $ points                <dbl> 87, 87, 87, 87, 87, 87, 87, 87, 87, 87, ...
## $ price                 <dbl> NA, 15, 14, 13, 65, 15, 16, 24, 12, 27, ...
## $ province              <chr> "Sicily & Sardinia", "Douro", "Oregon", ...
## $ region_1              <chr> "Etna", NA, "Willamette Valley", "Lake M...
## $ region_2              <chr> NA, NA, "Willamette Valley", NA, "Willam...
## $ taster_name           <chr> "Kerin O'Keefe", "Roger Voss", "Paul Gre...
## $ taster_twitter_handle <chr> "@kerinokeefe", "@vossroger", "@paulgwin...
## $ title                 <chr> "Nicosia 2013 Vulkà Bianco  (Etna)", "Qu...
## $ variety               <chr> "White Blend", "Portuguese Red", "Pinot ...
## $ winery                <chr> "Nicosia", "Quinta dos Avidagos", "Rains...
## $ address               <chr> "Nicosia Italy", "Quinta dos Avidagos Po...
## $ lon                   <dbl> 14.395278, -7.276971, -95.712891, -85.89...
## $ lat                   <dbl> 37.74692, 41.38793, 37.09024, 42.21225, ...
```