No Table, No Problem: Harvest your data with Rvest

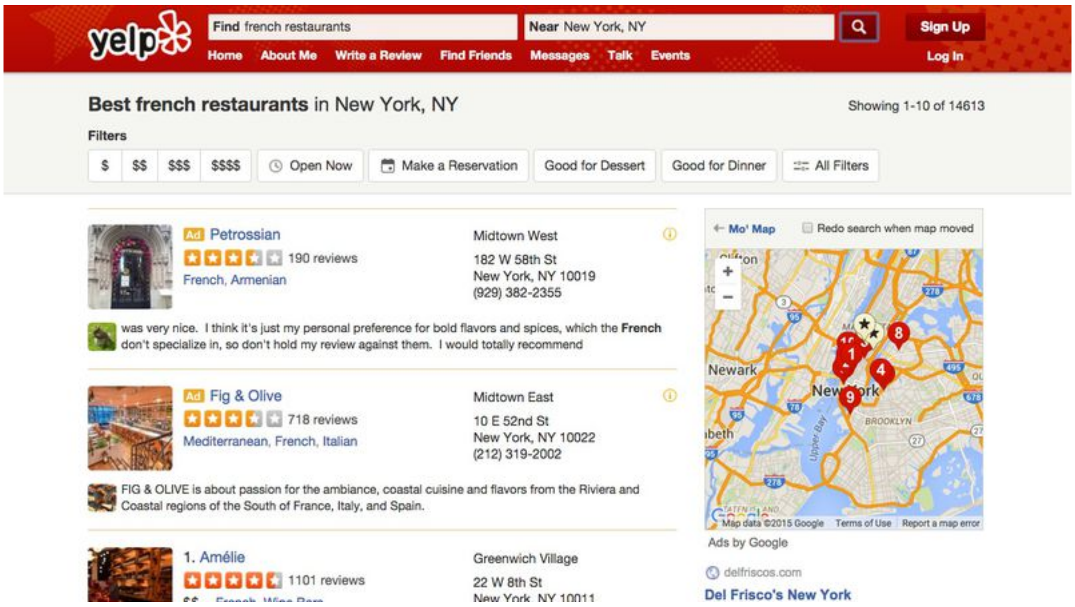
Ben Pirie

Introduction

The internet is filled with useful data - if you have a question, there’s likely an answer online. But if you want to perform any sort of analysis, the data you need is likely not packaged in a convenient downloadable file for you. Out of this need has spawn a technique known as web-scraping. This allows us to search through the HTML code of web pages to retrieve data from the root. While this sounds great, you might be wondering where to begin.

For both beginners, and more advance data scientists, there is not a much easier method for web-scraping than rvest (a package by Hadley Wickham). When combined with the web tool SelectorGadget, even your grandma could learn to web-scrape!

To see the beauty of rvest I will be looking at the Yelp website to help gather data on the top 100 restaurants in San Francisco, CA. Yelp has lots of useful data, but none of it is in an easily downloadable form. For example, here is a Yelp search for the best French restaurants in New York City



Best french restaurants in New York, NY

Showing 1-10 of 14613

Ad Petrossian
190 reviews
French, Armenian
Midtown West
182 W 58th St
New York, NY 10019
(929) 382-2355

was very nice. I think it's just my personal preference for bold flavors and spices, which the **French** don't specialize in, so don't hold my review against them. I would totally recommend

Ad Fig & Olive
718 reviews
Mediterranean, French, Italian
Midtown East
10 E 52nd St
New York, NY 10022
(212) 319-2002

FIG & OLIVE is about passion for the ambiance, coastal cuisine and flavors from the Riviera and Coastal regions of the South of France, Italy, and Spain.

1. Amélie
1101 reviews
Greenwich Village
22 W 8th St

Ads by Google

delfriscos.com

Del Frisco's New York

As you can see, each restaurant in the list has information on ratings, type of food, address, neighborhood, etc. However, it is not in a format that is easily convertable into a table - if you wanted to do data analysis, then you have run into a barrier before you have even started. But fear not! Web-scraping will come to your rescue. Though the name *web-scraping* may seem intimidating, with rvest and SelectorGadget it couldn’t be easier.
Before we begin you’ll need to do the following:

- install the rvest package using the command `install.package(rvest)`
- add the SelectorGadget bookmark to your browser’s bookmark bar

> Several Notes:
> I am running R version 3.4.1
> I will also do a bit of data processing and visualization on the data that we extract to inspire what you meght want to do next. If you want to replicate this part you will also need to install the ggvis and dplyr packages using `install.packages(ggvis)` and `install.packages(dplyr)`
> I am running ggvis version 0.4.3, dplyr version 0.7.4, and rvest verison 0.3.2

Some Background on rvest

Rvest is a package in R, built by Hadley Wickham. The pakage makes it easy to scrape information from web pages. As more and more information gets published online, having this skill is crucial - many of the publishers of this information do not make it easily downloadable.

The basic principal behind rvest, along with other web-scraping tools, is searching for key terms. Websites generally store information in HTML format, which is a very structured language. Because of this, if you know what you are looking for, it’s quite easy to find. This may not make sense yet, but keep reading - the example should clarify how rvest works.

Web Scraping with rvest

```r
# loading the necessary packages
library(rvest)
```

```
## Warning: package 'rvest' was built under R version 3.4.2
## Loading required package: xml2
```

```
## Warning: package 'xml2' was built under R version 3.4.2
```

```r
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
##
## 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
```

```r
library(ggvis)
```

```
## Warning: package 'ggvis' was built under R version 3.4.2
```

Step 1: Decide what data you want

The first question to ask is “what data do I want to gather?”. Well, as a senior in college who will be moving to San Francisco after I graduate, I am currently thinking about which part of the city I might want to live in. One thing I want to consider (since I am a self-proclaimed foodie) is where the best restaurant in the city are located. So, I will look to gather data on the top 100 restaurants in San Francisco. If I can put this data together in a nice data frame, then I can do some interesting analysis!

On Yelp, I can simply search for restaurants in San Francisco and order the results by “best”, to get this webpage.

Step 2: Download the HTML code

In order to gather the data we want, we’ll need to download the HTML code in which it stored.

> Note: You may have noticed the results from my above search on Yelp only return 10 retaurants per page. But, I said I was interested in finding out about the top 100. To keep it simple at first, we will just look at the first page with the top 10 results for now.

We use the rvest function `read_html()` to read the contents of a specified URL’s html code.

```r
URL1 <- "https://www.yelp.com/search?find_loc=San+Francisco,+CA&start=0&sortby=rating&cflt=restaurants"
yelp1 <- read_html(URL1)
```

Now we have an object `yelp1` which contains all the html code behind the first page of our Yelp search results. If we take a look at this object, which is stored in R as an `xml_document`, we can see that is simply a document which conatins XML (specifically HTML) code.

```r
yelp1
```

```
## {xml_document}
## <html xmlns:fb="http://www.facebook.com/2008/fbml" class="no-js" lang="en">
## [1] <head>\n<script>            window.yPageStart = new Date().getTime() ...
## [2] <body id="yelp_main_body" class="jquery country-us logged-out">\n\n  ...
```

Step 3: Nodes

In order to make sense of this `xml_document` I have to introduce the concept of a node. Our webpage is written in HTML, which is simply a specific version of XML, or Extensive Markup Language. Everything in an XML document is a node. The document itself, the elements of the document, the text in the elements, the attributes and even the comments are all nodes.

Lets take a look at some simple HTML code:

```
<html>

<head>
    <title>My First Web Page</title>
</head>

<body>
    <h1>My First Web Page</h1>
    <p><b>Hello World Wide Web!</b></p>
    <p><i>Hello World Wide Web!</i></p>
    <p><u>Hello World Wide Web!</u></p>
    <p>This is my first web page.</p>
    <p>HTML tags can give <b><i>various</i></b>
    <u>looks and format</u> to the content of this web page.</p>
</body>

</html>
```

We can clearly see the node structure. The whole document is a node, surrounded by `<html></html>`. There is also a heading "My First Web Page", which is surrounded by `<h1></h1>`. There are paragraph nodes, each surrounded by `<p></p>`. And so on…
To find the data we need from a webpage, all we need to do is find out where the node containing that data is. Though, with a long and complex HTML document, this can be a daunting task.

Step 4: SelectorGadget

SelectorGadget is an incredibly useful tool which makes finding our nodes easy. These nodes can be found using patterns, or in the jargon, CSS. SelectorGadget makes it increadibly easy to find the CSS that identifies our desired data. To find out more how it works, check out their webpage!

Step 5: Extracting Data

Using SelectorGadget, I found the CSS for the data that I wanted to retrieve. Specifically:

- The restaurant names are encoded with `#super-container .js-analytics-click span`
- The prices, given by a number of dollar signs, are given by `.price-range`
- The type of restaurant is given by `.category-str-list`
- The neighborhood that the restaurant is in is given by `.neighborhood-str-list`
- The rating out of 5 is encoded with `.rating-large`

Let's go through extracting each of these bits of information one-by-one.

```
# scraping the names of the restaurants
restaurant_nodes <- html_nodes(yelp1, "#super-container .js-analytics-click span")
restaurant_nodes
```

```
## {xml_nodeset (11)}
##  [1] <span>Extreme Pizza</span>
##  [2] <span>Piccolo Petes Cafe</span>
##  [3] <span>Cuisine of Nepal</span>
##  [4] <span>Ichido</span>
##  [5] <span>Gary Danko</span>
##  [6] <span>Khai</span>
##  [7] <span>Liholiho Yacht Club</span>
##  [8] <span>DIP</span>
##  [9] <span>Shizen Vegan Sushi Bar &amp; Izakaya</span>
## [10] <span>L<U+0092>ardoise Bistro</span>
## [11] <span>Kokkari Estiatorio</span>
```

The `html_nodes()` function takes in am `xml_document` or set of nodes, along with the CSS you are looking for. The function returns a set of XML nodes (the `xml_nodeset` object class). We can convert these nodes into a character vector using the `html_text()` function.
(Note: you can also retrieve your data using xpath instead of CSS, but I will focus on CSS here as it is much simmpler when you have access to a tool like SelectorGadget).

```
# getting restaurant name text
restaurants1 <- html_text(restaurant_nodes)
restaurants1
```

```
##  [1] "Extreme Pizza"                     "Piccolo Petes Cafe"
##  [3] "Cuisine of Nepal"                  "Ichido"
##  [5] "Gary Danko"                        "Khai"
##  [7] "Liholiho Yacht Club"               "DIP"
##  [9] "Shizen Vegan Sushi Bar & Izakaya" "L<U+0092>ardoise Bistro"
## [11] "Kokkari Estiatorio"
```

We're now left with a nice character vector containing the names of the top restaurants in San Francisco.

Before preceding onto the other bits, it's a good time to not that rvest is compatible with the pipe operator `%>%`. This useful feature can make our code much cleaner.

```
# scraping the prices of the restaurants
prices1 <- yelp1 %>%
  html_nodes(".price-range") %>%
  html_text()
prices1
```

```
##  [1] "$$"   "$"    "$$"   "$$$$" "$$$$" "$$$$" "$$$"  "$$"   "$$"   "$$$"
## [11] "$$$"
```

```
#scraping the types of the restaurants
types1 <- yelp1 %>%
  html_nodes(".category-str-list a:nth-child(1)") %>%
  html_text()
types1
```

```
##  [1] "Pizza"              "Coffee & Tea"       "Himalayan/Nepalese"
##  [4] "Seafood"            "American (New)"     "Vietnamese"
##  [7] "Bars"               "Sandwiches"         "Sushi Bars"
## [10] "French"             "Greek"
```

```
#scraping the neighborhoods of the restaurants
raw_neighborhoods1 <- yelp1 %>%
  html_nodes(".neighborhood-str-list") %>%
  html_text()
raw_neighborhoods1
```

```
##  [1] "\n            Visitacion Valley        "
##  [2] "\n            Bernal Heights, Mission        "
##  [3] "\n            SoMa        "
##  [4] "\n            Russian Hill, Fisherman's Wharf        "
##  [5] "\n            Mission Bay        "
##  [6] "\n            Lower Nob Hill        "
##  [7] "\n            North Beach/Telegraph Hill        "
##  [8] "\n            Mission        "
##  [9] "\n            Duboce Triangle        "
## [10] "\n            Financial District        "
```

With a little string manipulation we can get a nice character vector of the neighborhoods.

```
# getting the length of each string in the vector of neighborhoods
str_length <- unlist(lapply(raw_neighborhoods1, nchar))

# removing all extra spaces and characters
neighborhoods1 <- substr(raw_neighborhoods1, start = 14, stop = str_length -8)
neighborhoods1
```

```
##  [1] "Visitacion Valley"               "Bernal Heights, Mission"
##  [3] "SoMa"                            "Russian Hill, Fisherman's Wharf"
##  [5] "Mission Bay"                     "Lower Nob Hill"
##  [7] "North Beach/Telegraph Hill"      "Mission"
##  [9] "Duboce Triangle"                 "Financial District"
```

On Yelp, the ratings are given by 5 stars which are filled in to represent the average rating by customers. However, for our nice clean data frame we would like to have this data represented as a number. Fortunately, this data is encoded behind the image as a “title” attribute.

```
# scraping the ratings of the restaurants
raw_ratings1 <- yelp1 %>%
  html_nodes(".rating-large") %>%
  html_attr("title")
raw_ratings1
```

```
##  [1] "3.0 star rating" "5.0 star rating" "4.5 star rating"
##  [4] "5.0 star rating" "4.5 star rating" "4.5 star rating"
##  [7] "4.5 star rating" "4.5 star rating" "4.5 star rating"
## [10] "4.5 star rating" "4.5 star rating"
```

Along with the `html_nodes()` function, rvest contains the `html_attr()` function. When we pass the CSS “.rating-large” into `html_nodes()`, we get a set of nodes which contain a lot of information on the stars. By passing this set of nodes into `html_attr()` with the CSS “title” to extract the information on the number of stars for each restaurant.

With some simple string manipulation we can turn this into a clean numeric vector.

```
ratings1 <- substr(raw_ratings1, 1, 3) %>%
  as.numeric()
ratings1
```

```
##  [1] 3.0 5.0 4.5 5.0 4.5 4.5 4.5 4.5 4.5 4.5 4.5
```

A careful reader may have noticed that each of the vectors of data we scraped (besides neighborhoods1) includes 11 elements rather than 10. This is because each Yelp page has an advertisement at the top, which includes a restauurant’s name, type, price, and rating. Thus we must remove the first element of each of those corresponding vectors before we build our data frame.

```
restaurants1 <- restaurants1[2:length(restaurants1)]
types1 <- types1[2:length(types1)]
prices1 <- prices1[2:length(prices1)]
ratings1 <- ratings1[2:length(ratings1)]

top_ten <- data.frame(restaurants1, types1, neighborhoods1, prices1, ratings1)
top_ten
```

```
##                        restaurants1            types1
## 1                 Piccolo Petes Cafe       Coffee & Tea
## 2                   Cuisine of Nepal Himalayan/Nepalese
## 3                             Ichido            Seafood
## 4                         Gary Danko     American (New)
## 5                               Khai         Vietnamese
## 6                Liholiho Yacht Club               Bars
## 7                                DIP         Sandwiches
## 8  Shizen Vegan Sushi Bar & Izakaya          Sushi Bars
## 9                   L<U+0092>ardoise Bistro              French
## 10                 Kokkari Estiatorio              Greek
##                      neighborhoods1 prices1 ratings1
## 1                 Visitacion Valley       $      5.0
## 2           Bernal Heights, Mission      $$      4.5
## 3                              SoMa    $$$$      5.0
## 4  Russian Hill, Fisherman's Wharf    $$$$      4.5
## 5                       Mission Bay    $$$$      4.5
## 6                    Lower Nob Hill     $$$      4.5
## 7       North Beach/Telegraph Hill      $$      4.5
## 8                           Mission      $$      4.5
## 9                   Duboce Triangle     $$$      4.5
## 10              Financial District     $$$      4.5
```

Extending the Example

Now that I’ve proven that this web-scraping process works for one page, let’s try and scrape the first 10 pages of our Yelp results to gather data on all top 100 restaurants in San Francisco.

If you click through the results pages you will notice several things. Firstly, each page is formatted the same and has the same underlying HTML structure. Additionally, the URL is the same for each page, except the last digits which indicate the rank of the restaurant the page starts with. Because of these two imprtant factors, we can implement a for-loop to make scraping all 10 pages easy!

```r
#our final complete data frame
yelp_dat <- data.frame()

# vector of URL ending digits
page_starts <- seq(0, 90, by = 10)


for (i in page_starts) {
  URL <- paste("https://www.yelp.com/search?find_loc=San+Francisco,+CA&start=",
                    i, "&sortby=rating&cflt=restaurants", sep = "")

  page <- read_html(URL)

  #scraping restaurant names
  restaurants <- page %>%
    html_nodes("#super-container .js-analytics-click span") %>%
    html_text()
  # removing advertisement
  if(length(restaurants) == 11) {
    restaurants <- restaurants[2:length(restaurants)]
  }

  # scraping ratings data
  ratings <- page %>%
    html_nodes(".rating-large") %>%
    html_attr("title") %>%
    substr(1, 3) %>%
    as.numeric()
  # removing advertisement
  if(length(ratings) == 11) {
    ratings <- ratings[2:length(ratings)]
  }

  # scraping prices data
  prices <- page %>%
    html_nodes(".price-range") %>%
    html_text()
  # removing advertisement
  if(length(prices) == 11) {
    prices <- prices[2:length(prices)]
  }

  # scraping restauract category data
  types <- page %>%
    html_nodes(".category-str-list a:nth-child(1)") %>%
    html_text()
  # removing advertisement
  if(length(types) == 11) {
    types <- types[2:length(types)]
  }

  # scraping location data
  neighborhoods <- page %>%
    html_nodes(".neighborhood-str-list") %>%
    html_text()
  # cleaning neighborhood strings
  neighborhoods <- substr(neighborhoods, start = 14, stop = unlist(lapply(neighborhoods, nchar)) -8)
  #removing advertisement
  if(length(neighborhoods) == 11) {
    neighborhoods <- neighborhoods[2:length(neighborhoods)]
  }

  page_dat <- data.frame(Name = restaurants, Rating = ratings, Price = prices, Category = types, Neighborhood = ne
ighborhoods)

  # adding page data to overall data frame
  yelp_dat <- rbind(yelp_dat, page_dat)
}
yelp_dat
```

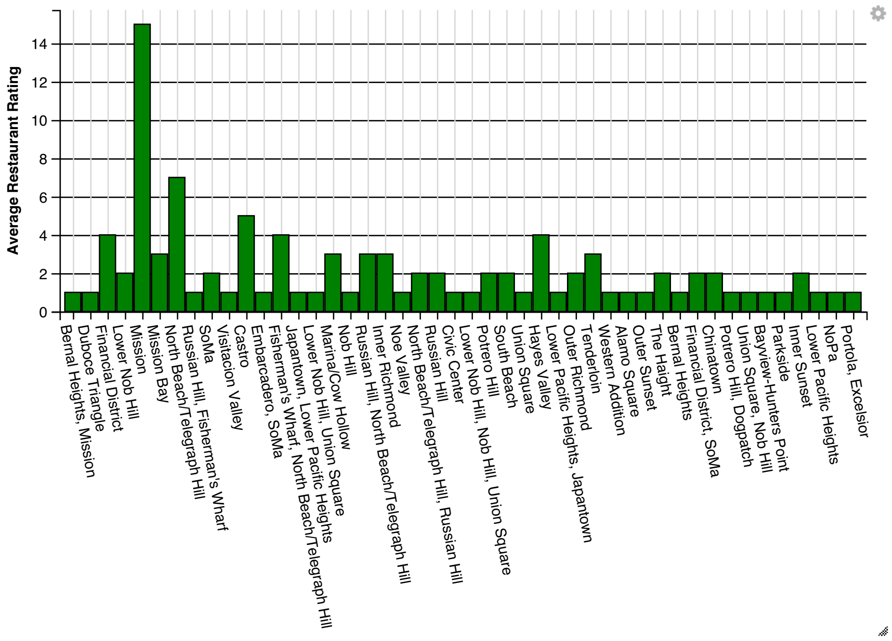
```
##                                  Name Rating Price              Category
## 1                  Piccolo Petes Cafe    5.0     $           Coffee & Tea
## 2                    Cuisine of Nepal    4.5    $$     Himalayan/Nepalese
## 3                              Ichido    5.0  $$$$                Seafood
## 4                          Gary Danko    4.5  $$$$         American (New)
## 5                                Khai    4.5  $$$$             Vietnamese
## 6                  Liholiho Yacht Club    4.5   $$$                   Bars
## 7                                 DIP    4.5    $$             Sandwiches
## 8     Shizen Vegan Sushi Bar & Izakaya    4.5    $$             Sushi Bars
## 9                    L<U+0092>ardoise Bistro    4.5   $$$                 French
## 10                 Kokkari Estiatorio    4.5   $$$                  Greek
## 11            Italian Homemade Company    4.5    $$                Grocery
## 12                            Tacorea    4.5     $                Mexican
## 13                             OzaOza    4.5  $$$$               Japanese
## 14                  Anchor Oyster Bar    4.5   $$$                Seafood
## 15                        Seven Hills    4.5   $$$                Italian
## 16       The Codmother Fish and Chips    4.5     $                British
## 17                     Rooster & Rice    4.5    $$           Chicken Shop
## 18                             Bodega    4.5    $$              Wine Bars
## 19                          The House    4.5   $$$           Asian Fusion
## 20               Hog Island Oyster Co    4.5    $$                Seafood
## 21                         Pink Onion    4.5    $$                  Pizza
## 22                         The Morris    4.5   $$$         American (New)
## 23                            Surisan    4.5    $$                 Korean
## 24        Betty Lou<U+0092>s Seafood & Grill    4.5    $$                Seafood
## 25                          Lazy Bear    4.5  $$$$         American (New)
## 26                           Frascati    4.5   $$$         American (New)
## 27                          La Fusión    4.5    $$         Latin American
## 28                          La Ciccia    4.5   $$$                Seafood
## 29                   Kitchen Istanbul    4.5    $$                  Greek
## 30                            Frances    4.5   $$$         American (New)
## 31                         Sotto Mare    4.5    $$                Seafood
## 32                       Stones Throw    4.5   $$$         American (New)
## 33                   Golden Boy Pizza    4.5     $                  Pizza
## 34                       Rove Kitchen    4.5    $$                Burgers
## 35  Tuba Authentic Turkish Restaurant    4.5    $$          Mediterranean
## 36                   Fermentation Lab    4.5    $$             Gastropubs
## 37                             Garaje    4.5    $$                Mexican
## 38                            Hogwash    4.5    $$             Gastropubs
## 39                         Chez Maman    4.5    $$                 French
## 40              Hopwater Distribution    4.5    $$ American (Traditional)
## 41                      Don Pisto<U+0092>s    4.5    $$                Mexican
## 42                       Pizzetta 211    4.5    $$                  Pizza
## 43                          Cholo Soy    4.5     $               Peruvian
## 44                            Tselogs    4.5     $               Filipino
## 45                              Birba    4.5    $$              Wine Bars
## 46                     Marufuku Ramen    4.5    $$                  Ramen
## 47                        Box Kitchen    4.5     $                Burgers
## 48                                HRD    4.0    $$                 Korean
## 49              State Bird Provisions    4.0   $$$     Tapas/Small Plates
## 50                        El Farolito    4.0     $                Mexican
## 51          Brenda<U+0092>s French Soul Food    4.0    $$     Breakfast & Brunch
## 52                            Lolinda    4.0   $$$            Steakhouses
## 53                             RM 212    4.5    $$             Tapas Bars
## 54                Baonecci Ristorante    4.0    $$                  Pizza
## 55                          Hook Fish    4.5    $$                Seafood
## 56                               Nopa    4.0   $$$         American (New)
## 57                         DragonEats    4.5     $             Vietnamese
## 58                   Jiangnan Cuisine    4.5    $$           Shanghainese
## 59                        Alma Cocina    4.5    $$         Latin American
## 60            Tony<U+0092>s Pizza Napoletana    4.0    $$                  Pizza
## 61                          Takoba SF    4.5    $$               Japanese
## 62                          Parada 22    4.0    $$         Latin American
## 63                    Burma Superstar    4.0    $$                Burmese
## 64                         Il Pollaio    4.5    $$                Italian
## 65                    Bella Trattoria    4.0    $$                Italian
## 66                     Hai Ky Noodles    4.0     $                Chinese
## 67                        Saucy Asian    4.5    $$                 Korean
## 68                               Loló    4.0    $$     Tapas/Small Plates
## 69                   Myriad Gastropub    4.5    $$             Gastropubs
## 70                   Red Hill Station    4.5    $$                Seafood
## 71                           The Bird    4.0     $           Chicken Shop
## 72               Good Mong Kok Bakery    4.0     $               Bakeries
## 73                        Suppenküche    4.0    $$                 German
## 74                           Castagna    4.5    $$                Italian
## 75                          Causwells    4.0    $$         American (New)
## 76              Fog Harbor Fish House    4.0    $$                Seafood
## 77                   Sons & Daughters    4.0  $$$$         American (New)
## 78                        FOB Kitchen    4.5    $$               Filipino
## 79                               aina    4.0    $$     Breakfast & Brunch
## 80                           Tanguito    4.0     $              Argentine
## 81     Farmhouse Kitchen Thai Cuisine    4.0    $$                   Thai
## 82                             Souvla    4.0    $$                  Greek
## 83                               Duna    4.5    $$        Modern European
## 84                          Mac Daddy    4.0    $$ American (Traditional)
## 85                      The Spice Jar    4.0    $$           Asian Fusion
## 86           Huli Huli Hawaiian Grill    4.5     $               Hawaiian
## 87                  House Of Pancakes    4.0     $                Chinese
## 88                        Oasis Grill    4.0     $                  Greek
## 89                            Bellota    4.0   $$$                Spanish
## 90                The Castro Republic    4.5    $$         Latin American
## 91                Trattoria Contadina    4.0    $$                Italian
## 92                           San Tung    4.0    $$                Chinese
## 93          Akira Japanese Restaurant    4.5    $$               Japanese
## 94                          Cockscomb    4.0   $$$                   Bars
## 95                             a Mano    4.0    $$                Italian
## 96           Breakfast at Tiffany<U+0092>s    4.0    $$                 Diners
## 97                             Souvla    4.5    $$          Mediterranean
## 98                      Kitchen Story    4.0    $$     Breakfast & Brunch
## 99                        Katsu House    4.5    $$             Sushi Bars
## 100                    Yummy Dumpling    4.5     $    Do-It-Yourself Food
##                                      Neighborhood
## 1                               Visitacion Valley
## 2                         Bernal Heights, Mission
## 3                                            SoMa
## 4                 Russian Hill, Fisherman's Wharf
## 5                                     Mission Bay
## 6                                  Lower Nob Hill
## 7                      North Beach/Telegraph Hill
## 8                                         Mission
## 9                                 Duboce Triangle
## 10                             Financial District
## 11       Russian Hill, North Beach/Telegraph Hill
## 12                   Lower Nob Hill, Union Square
## 13               Japantown, Lower Pacific Heights
## 14                                         Castro
## 15                                       Nob Hill
## 16  Fisherman's Wharf, North Beach/Telegraph Hill
## 17                              Marina/Cow Hollow
## 18       Russian Hill, North Beach/Telegraph Hill
## 19                     North Beach/Telegraph Hill
## 20                              Embarcadero, SoMa
## 21                                        Mission
## 22                                        Mission
## 23  Fisherman's Wharf, North Beach/Telegraph Hill
## 24       North Beach/Telegraph Hill, Russian Hill
## 25                                        Mission
## 26                                   Russian Hill
## 27                             Financial District
## 28                                     Noe Valley
## 29                                 Inner Richmond
## 30                                         Castro
## 31                     North Beach/Telegraph Hill
## 32                                   Russian Hill
## 33                     North Beach/Telegraph Hill
## 34                                 Lower Nob Hill
## 35                                        Mission
## 36                                   Civic Center
## 37                                    South Beach
## 38                                   Union Square
## 39                                   Potrero Hill
## 40         Lower Nob Hill, Nob Hill, Union Square
## 41                     North Beach/Telegraph Hill
## 42                                 Outer Richmond
## 43                                        Mission
## 44                                     Tenderloin
## 45                                   Hayes Valley
## 46               Lower Pacific Heights, Japantown
## 47                                           SoMa
## 48                                    South Beach
## 49                               Western Addition
## 50                                        Mission
## 51                                     Tenderloin
## 52                                        Mission
## 53                             Financial District
## 54                     North Beach/Telegraph Hill
## 55                                   Outer Sunset
## 56                                   Alamo Square
## 57                                     The Haight
## 58                                 Outer Richmond
## 59                                        Mission
## 60                     North Beach/Telegraph Hill
## 61                       Financial District, SoMa
## 62                                     The Haight
## 63                                 Inner Richmond
## 64       North Beach/Telegraph Hill, Russian Hill
## 65                                 Inner Richmond
## 66                                     Tenderloin
## 67                                         Castro
## 68                                        Mission
## 69                                        Mission
## 70                                 Bernal Heights
## 71                       Financial District, SoMa
## 72                                      Chinatown
## 73                                   Hayes Valley
## 74                              Marina/Cow Hollow
## 75                              Marina/Cow Hollow
## 76  Fisherman's Wharf, North Beach/Telegraph Hill
## 77                         Union Square, Nob Hill
## 78                                        Mission
## 79                         Potrero Hill, Dogpatch
## 80  Fisherman's Wharf, North Beach/Telegraph Hill
## 81                                        Mission
## 82                                   Hayes Valley
## 83                                        Mission
## 83                                         Mission
## 84                                   Potrero Hill
## 85                                         Mission
## 86                       Bayview-Hunters Point
## 87                                       Parkside
## 88                          Financial District
## 89                                   Mission Bay
## 90                                         Castro
## 91        Russian Hill, North Beach/Telegraph Hill
## 92                                   Inner Sunset
## 93                       Lower Pacific Heights
## 94                                   Mission Bay
## 95                                  Hayes Valley
## 96                          Portola, Excelsior
## 97                                           NoPa
## 98                                         Castro
## 99                                      Chinatown
## 100                                  Inner Sunset
```

Conclusion

Using rvest to scrape your data from the web couldn't be easier. It is incredibly valuable to take the (short amount of) time to learn rvest and SelectorGadget - with these two tools in hand you are well on your way to gathering the data you need for your analysis projects!

Now that we have this data in a nice data frame format we can do all kinds of exploratory data analysis with it. For example, if we wanted to know how many of the top 100 restaurants are in each neighborhood, we could generate the following graph:

```
yelp_dat %>%
  mutate(Neighborhood = as.factor(Neighborhood)) %>%
  count(Neighborhood) %>%
  ggvis(~Neighborhood, ~n, fill := "green", fill.hover := "yellow") %>%
  layer_bars() %>%
  add_axis("y", title = "Average Restaurant Rating", properties = axis_props(
    grid = list(stroke = "black")
  )) %>%
  add_axis("x", title = "", properties = axis_props(
    labels = list(angle = 80, align = "left")
  ))
```



Or, suppose you had a long day of work and are suffering from decision fatigue. You can't decide where to go for dinner, so you randomly select one of the top 100 restaurants using the following code:

```
# including a seed for reproducibility
set.seed(84957)
index <- sample(1:length(yelp_dat$Name), 1)
yelp_dat[index,]
```

```
##           Name Rating Price Category Neighborhood
## 39 Chez Maman    4.5    $$   French Potrero Hill
```

The options of what to do with data are endless, the hard part is getting it - rvest makes that part easy too.

Happy web-scraping!

References

- Hadley Wickham rvest introduction
- Yelp
- Yelp search screenshot
- photo of HTML code
- rvest README
- XML node stucture information
- Youtube video rvest tutorial