Lab 03 - Visualizing spatial data

2018-02-01

Due: 2018-02-08 at noon

Introduction

Have you ever taken a road trip in the US and thought to yourself "I wonder what La Quinta means". Well, the late comedian Mitch Hedberg thinks it's Spanish for *next to Denny's*.

If you're not familiar with these two establishments, Denny's is a casual diner chain that is open 24 hours and La Quinta Inn and Suites is a hotel chain.

These two establishments tend to be clustered together, or at least this observation is a joke made famous by Mitch Hedberg. In this lab we explore the validity of this joke and along the way learn some more data wrangling and tips for visualizing spatial data.

The inspiration for this lab comes from a blog post by John Reiser on his *new jersey geographer* blog. You can read that analysis here. Reiser's blog post focuses on scraping data from Denny's and La Quinta Inn and Suites websites using Python. In this lab we focus on visualization and analysis of these data. However note that the data scraping was also done in R, and we we will discuss web scraping using R later in the course. But for now we focus on the data that has already been scraped and tidied for you.

Getting started

- Go to the course organization on GitHub: `https://github.com/Sta199-S18`.
- Find the repo starting with `lab-03` and that has your team name at the end (this should be the only `lab-03` repo available to you).
- In the repo, click on the green **Clone or download** button, select **Use HTTPS** (this might already be selected by default, and if it is, you'll see the text

Clone with HTTPS as in the image below). Click on the clipboard icon to copy the repo URL.

- Go to RStudio Cloud and into the course workspace. Create a **New Project from Git Repo**. You will need to click on the down arrow next to the **New Project** button to see this option.
- Copy and paste the URL of your assignment repo into the dialog box:
- Hit OK, and you're good to go!

Packages

In this lab we will work with the `tidyverse` package. So we need to install and load it:

```
install.packages("tidyverse")
library(tidyverse)
```

Note that this package is also loaded in your R Markdown document.

Housekeeping

Git configuration

Your email address is the address tied to your GitHub account and your name should be first and last name.

- Go to the *Terminal* pane
- Type the following two lines of code, replacing the information in the quotation marks with your info.

```
git config --global user.email "your email"
git config --global user.name "your name"
```

To confirm that the changes have been implemented, run the following:

```
git config --global user.email
git config --global user.name
```

Project name:

Currently your project is called *Untitled Project*. Update the name of your project to be "Lab 03 - Visualizing spatial data".

Warm up

Pick one team member to complete the steps in this section while the others contribute to the discussion but do not actually touch the files on their computer.

Before we introduce the data, let's warm up with some simple exercises.

YAML:

Open the R Markdown (Rmd) file in your project, change the author name to your **team** name, and knit the document.

Commiting and pushing changes:

- Go to the **Git** pane in your RStudio.
- View the **Diff** and confirm that you are happy with the changes.
- Add a commit message like "Update team name" in the **Commit message** box and hit **Commit**.
- Click on **Push**. This will prompt a dialogue box where you first need to enter your user name, and then your password.

Pulling changes:

Now, the remaining team members who have not been concurrently making these changes on their projects should click on the **Pull** button in their Git pane and observe that the changes are now reflected on their projects as well.

The data

The data consist of two csv (comma separated values) files: one for Denny's locations and the other for La Quinta.

```
dn <- read_csv("data/dennys.csv")
lq <- read_csv("data/laquinta.csv")
```

Note that these data were scraped from here and here, respectively.
To help with our analysis we will also use a dataset on US states:

```
states <- read_csv("data/states.csv")
```

Each observation in this dataset represents a state, including DC. Along with the name of the state we have the two-letter abbreviation and we have the geographic area of the state (in square miles).

Exercises

Exercise 1: What are the dimensions of the Denny's dataset? (Hint: Use inline R code and functions like `nrow` and `ncol` to compose your answer.) What does each row in the dataset represent? What are the variables?

Exercise 2: What are the dimensions of the La Quinta's dataset? What does each row in the dataset represent? What are the variables?

We would like to limit our analysis to Denny's and La Quinta locations in the United States.

Exercise 3: Take a look at the websites that the data come from (linked above). Are there any La Quinta's locations outside of the US? If so, which countries? What about Denny's?

Exercise 4: Now take a look at the data. What would be some ways of determining whether or not either establishment has any locations outside the US using just the data (and not the websites). Don't worry about whether you know how to implement this, just brainstorm some ideas. Write down at least one as your answer, but you're welcomed to write down a few options too.

We will determine whether or not the establishment has a location outside the US using the `state` variable in the `dn` and `lq` datasets. We know exactly which states are in the US, and we have this information in the `states` dataframe we loaded.

Exercise 5: Find the Denny's locations that are outside the US, if any. To do so, filter the Denny's locations for observations where `state` is not in `states$abbreviation`. The code for this is given below. Note that the `%in%` operator matches the states listed in the `state` variable to those listed in `states$abbreviation`. The `!` operator means **not**. Are there any Denny's locations outside the US?

"Filter for `states` that are not in `states$abbreviation`."

```
dn %>%
  filter(!(state %in% states$abbreviation))
```

Exercise 6: Add a country variable to the Denny's dataset and set all observations equal to `"United States"`. Remember, you can use the `mutate` function for adding a variable.

We don't need to tell R how many times to repeat the character string "United States" to fill in the data for all observations, R takes care of that automatically.

```
dn %>%
  mutate(country = "United States")
```

Make sure to save the result of this as `dn` again so that the stored data frame contains the new variable going forward.

Exercise 7: Find the La Quinta locations that are outside the US, and figure out which country they are in. This might require some googling. Take notes, you will need to use this information in the next exercise.

Exercise 8: Add a country variable to the La Quinta dataset. Use the `case_when` function to populate this variable. You'll need to refer to your notes from Exercise 7 about which country the non-US locations are in. Here is some starter code to get you going:

```
lq %>%
  mutate(country = case_when(
    state %in% state.abb     ~ "United States",
    state %in% c("ON", "BC") ~ "Canada",
    state == "ANT"           ~ "Colombia",
    ...                      # fill in the rest
  ))
```

GOING FORWARD we will work with the data from the United States only. All Denny's locations are in the United States, so we don't need to worry about them. However we do need to filter the La Quinta dataset for locations in United States.

```
lq <- lq %>%
  filter(country == "United States")
```

Exercise 9: Which states have the most and fewest Denny's locations? What about La Quinta? Is this surprising? Why or why not?

Next, let's calculate which states have the most Denny's locations *per thousand square miles*. This requires joinining information from the frequency tables you created in Exercise 8 with information from the `states` data frame.

First, we count how many observations are in each state, which will give us a data frame with two variables: `state` and `n`. Then, we join this data frame with the `states` data frame. However note that the variables in the `states` data frame that has the two-letter abbreviations is called `abbreviation`. So when we're joining the two data frames we specify that the `state` variable from the Denny's data should be matched `by` the `abbreviation` variable from the `states` data:

```
dn %>%
  count(state) %>%
  inner_join(states, by = c("state" = "abbreviation"))
```

Before you move on the the next question, run the code above and take a look at the output. In the next exercise you will need to build on this pipe.

Exercise 10: Which states have the most Denny's locations per thousand square miles? What about La Quinta?

Next, we put the two datasets together into a single data frame. However before we do so, we need to add an identifier variable. We'll call this `establishment` and set the value to `"Denny's"` and `"La Quinta"` for the `dn` and `lq` data frames, respectively.

```
dn <- dn %>%
  mutate(establishment = "Denny's")
lq <- lq %>%
  mutate(establishment = "La Quinta")
```

Since the two data frames have the same columns, we can easily bind them with the `bind_rows` function:

```
dn_lq <- bind_rows(dn, lq)
```

We can plot the locations of the two establishments using a scatter plot, and color the points by the establishment type. Note that the latitude is plotted on the x-axis and the longitude on the y-axis.

```
ggplot(dn_lq, mapping = aes(x = longitude, y = latitude, color = establishment)) +
  geom_point()
```

See here for help with the syntax for customizing your plots. You can also choose different themes to change the overall look of your plots, see here for help with these.

The following two questions ask you to create visualizations. These should follow best practices you learned in class, such as informative titles, axis labels, etc.

Exercise 11: Filter the data for observations in North Carolina only, and recreate the plot. You should also adjust the transparency of the points, by setting the `alpha` level, so that it's easier to see the overplotted ones. Visually, does Mitch Hedberg's joke appear to hold here?

Exercise 12: Now filter the data for observations in Texas only, and recreate the plot, with an appropriate `alpha` level. Visually, does Mitch Hedberg's joke appear to hold here?

That's it for now! In the next lab we will take a more quantitative approach to answering these questions.