02.522: Urban Data & Methods I: Computational Urban Analysis

Ate Poorthuis

Contents

Chapter 1

Syllabus

1.1 Class Schedule

January Term 2020
Tue 12.30-2pm Think Thank 7 (1.409)
Thu 11.00-12.30pm Think Thank 17 (2.202)
Office Hours: By appointment

1.2 Course Description

This course focuses on quantitative and computational approaches to urban analytics and data science. It exposes students to new ways of collecting large datasets ("big data") and innovative methods of analysing such datasets. It draws on both more conventional methods such as (spatial) statistics, as well as how to appropriately use methods from data science and machine learning within an urban context.

1.3 Format

The course is structured around two 1.5 hour classes per week that integrate lecture, discussion and in-class activities and exercises in an interactive manner. The class is further structured around 6 blocks (each lasting two weeks).

1.4 Expectations

Students are expected to be present and actively participate in each class, as well as on the class online forum (Slack). Before coming to class, you will have read the assigned readings and you are coming to class prepared to participate in discussion and exercises.

You are also expected to produce your own work, whether individually or in groups. Do not copy work from the internet or other published sources without proper citations. This is plagiarism and if a student is found to be doing so, he or she will be subject to disciplinary measures including potentially failing the course.

1.5 Assessment

There will be a variety of assessments throughout the semester. Emphasis is on your performance overall, with relatively low weight placed on individual items. Continued participation throughout the semester will enable you to do well in this course.

Assessment Items	Percentage	Period
Class Participation	15%	Throughout term
Assignments (six)	45%	Throughout term
Final Project	40%	Week 14

1.5.1 Assignments

Each block consists of a series of exercises that culminate in an assigment/report that will be due before the start of the next block (Monday 23.59).

1.5.2 Final Project

For the final project, you will develop conduct a computational analysis of a topic and dataset of your own choice. This is an opportunity to explore (an aspect of) your final term project in more detail – or take a deep-dive in a topic or dataset that you're interested in. You can choose to use already existing datasets, or collect your own but you must use one or more computational methods to help answer your reseach question. In Week 6, you will hand in a research proposal of a maximum of 1000 words. It should discuss your motivation for the project; its objectives and research question; data requirements (does the data already exist? where do you get it from? does it need a lot of

post-processing); and the methods you plan on utilizing. The remainder of the term is spent on collecting and analyzing data. A prototype of your analysis is due at the end of Week 13 and the final version at the end of Week 14. You are required to hand in both the code, a written paper and (where appropriate) visualizations as part of your Github repository. You can do this in a single form factor through one or more RMarkdown documents.

1.6 Deadlines

Deadlines are as noted in the course syllabus or on the specific assignment. If something is due on a specific date, you have until midnight on that day to submit the assignment.

1.7 Software

We will use R, RStudio and a series of packages (most of them compatible with the Tidyverse). All software used in the course is open-source and freely available.

1.8 Detailed Outline

1.8.1 Block 1: Analysis of HDB Resale Prices I

1. Reproducible Science & Project Management (+ re-view of tidyverse basics)

- Reading:
  - Excuse me, do you have a moment to talk about version control?
  - Happy Git with R (Chapter 1-15)
  - Reproducible research and quantitative geography
  - Pro Git (optional)

2. Exploratory Data Analysis (Univariate Statistics & Visualization)

- Reading:
  - Happy Git with R (Chapter 20-23)
  - Data Visualization Chapter 3
  - Burt et al., *Elementary Statistics for Geographers*, (Chapter 3)

1.8.2 Block 2: Analysis of HDB Resale Prices II

3. Sampling, Bootstrapping & Confidence Intervals

- Reading:
  - Burt et al., *Elementary Statistics for Geographers*, (Chapter 6)
  - Modern Dive Chapter 8
  - Modern Dive Chapter 9

4. Correlation & Linear Regression

- Reading:
  - Burt et al., *Elementary Statistics for Geographers*, (Chapter 4)
  - Modern Dive Chapter 6

1.8.3 Block 3: Geodemographics of SG Neighborhoods I

5. Dimension Reduction (MDS/PCA)

- Reading:
  - Gatrell, Anthony C. "Multidimensional Scaling." In *Quantitative Geography*. Routledge & Kegan Paul, 1981.
  - Principal Component Analysis, Explained Visualy
  - Wyly's notes on PCA

6. Dimension Reduction (T-SNE)

- Reading:
  - Van der Maaten's Google Tech Talk

7. Break

1.8.4 Block 4: Geodemographic of SG Neighborhoods II

8. Clustering (non-spatial)

- Reading:
  - Fortunato, F. Community Detection in Graphs. Specifically section 4a-c.
  - Jain, A. Data clustering: 50 years beyond K-means
  - Ester et al. A Density-Based Algorithm for Discovering Clusters

9. Clustering (spatial, including refresher on spatial data structures)

- Reading:
  - Guo, D. Regionalization with dynamically constrained agglomerative clustering and partitioning (REDCAP)
  - Poorthuis, A. How to draw a neighborhood

1.8.5 Block 5: Spatial Statistics

10. Spatial Statistics I (Spatial Autocorrelation)

- Reading:
  - TBA

11. Spatial Statistics II (Spatial Regression Modeling)

- Reading:
  - TBA

1.8.6 Block 6: Classification

12. Classification (logistic regression & SVM)

- Reading:
  - TBA

13. Classification (overflow, possible excursion to deep learning)

- Reading:
  - TBA

14. Final Project Studio

Chapter 2

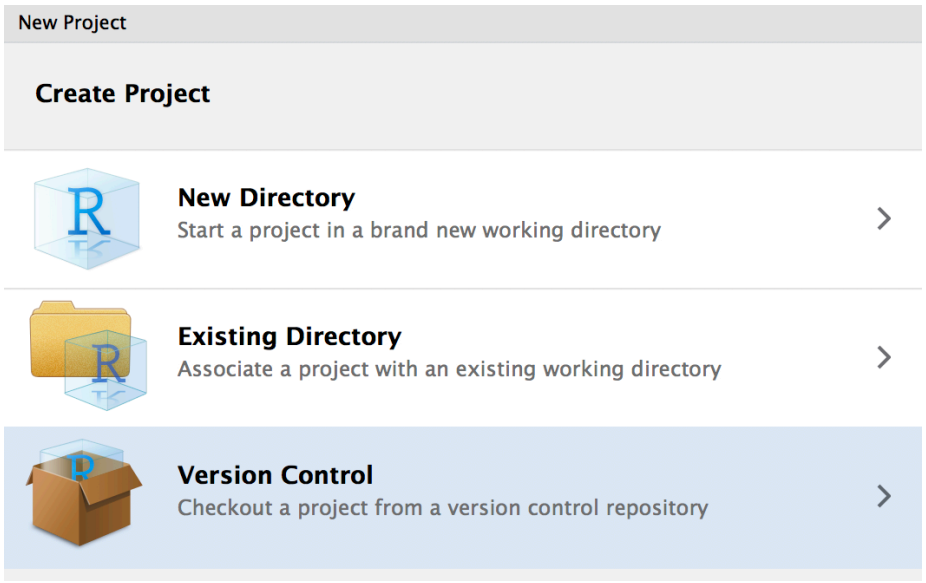
Week 1

2.1 Taking control of your repository

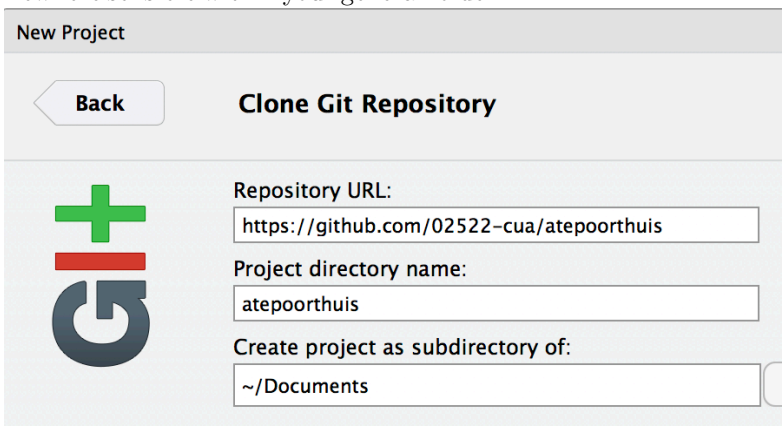
Our key task for this first week is to make ourselves comfortable with using git as a version control system for projects that involve data analysis. Git is a very powerful system – and many folks even use it to manage other types of projects (e.g. writing). To keep things simple, we'll focus specifically on how git relates to our workflow with R/Rstudio for now.

We will start by taking possesion of the repository you will use to do your work in the context of this course. It has been set up for you already on github.com and should follow the structure `02522-cua/yourname`. For example, my repository would reside at https://github.com/02522-cua/atepoorthuis. Find your repository (here's a list of all repos in our course organization) and take note (i.e. copy it to the clipboard) of its url.

Start a new project in RStudio and choose to start one from version control.



Choose to start a project from Git and paste your github repository url in. Do make sure you store the repository somewhere sensible within your general folder organization so you can find it again.



Once you click OK, RStudio will create a new directory and will download (this is called 'cloning' in git parlance) the contents of your repository into it.

2.2 Making your first commit

Now that your repository has found a place on your computer, let's practice changing some things around and see how these changes will be reflected online. There are a few specific steps involved that you need to understand (also check out this decent visual summary):

1. You make some actual changes. You can modify existing files and save those changes or create entirely new files.
2. After making changes, git does not actually now about them yet. We tell git about any changes by `adding` or `staging` changed files. This sounds tedious but it is actually very handy when you're experimenting and don't necessarily want your half-baked experiment to be sent to the online repository.
3. Once you are happy with your changes, you can make a `commit` of all the staged changes up until that moment. I like to think of this is a milestone. You say: '*these are my changes, please check them in*'. This is also reflected by the fact that every `commit` is accompanied by a commit message that summarizes those changes. We generally keep these messages short so they're easier to read later.
4. After commiting, the changes only still exist on our local computer. We can finally `push` our changes up to Github.com.

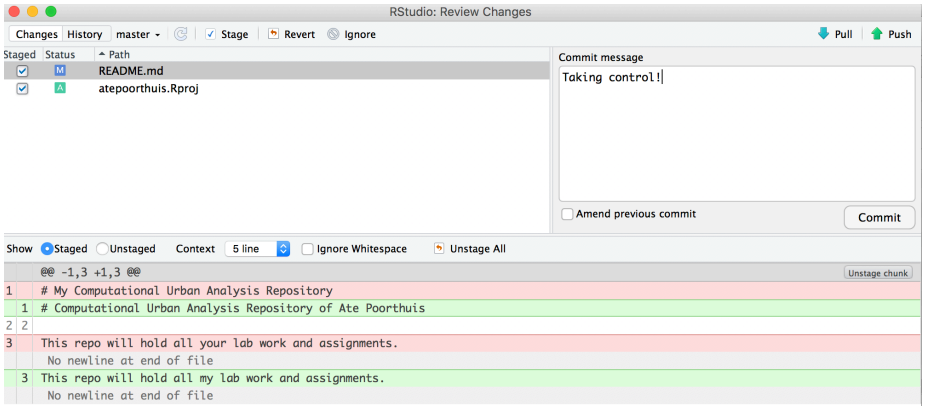
Time for practice. Open the README.md file in the 'root' of your project. The current description is a bit generic so let's personalize. I changed mine to:

```
# Computational Urban Analysis Repository of Ate Poorthuis

This repo will hold all my lab work and assignments.
```

Save your change and go to the `Git` pane (usually in the right-hand side of your RStudio window). You should see your README.md file show up there now. If it does, click the `Diff` or `Commit` button. This will open up another window, in which we can do all of our next steps.

- First we need to `stage` our changes. Hit the tick-box in front of README.md. (If the `.proj` file shows up, we can stage that too)
- Next we need to `commit` our staged changes. Write a short commit message describing your change and hit the Commit button.



Now we have commited our changes *locally* but this is not yet reflected on github.com. We can `push` the state of our repository up to github.com. You

will notice a message on the top-left of the git window indicating 'Your branch is 1 commit ahead of origin/master'. `Origin` is git parlance for the online version of our repository. To bring `origin` back into sync with our local version, just press 'Push' and go to your repository on github.com to see the changes. Congratulations you are now a budding git ninja!

2.3 Setting up our HDB data

Now that you have properly taken control of your repository, it is time to start setting up our data for subsequent analysis. The task at hand is to analyze HDB resale prices – we will start next week with univariate statistics (i.e. analyzing single variables in isolation), after which we will move to multivariate statistics (i.e. looking at the relation between multiple variables). For now, we are just going to set up shop: read and clean our dataset using a series of `tidyverse` operations.

Create a new R Notebook and save it with an appropriate name in your `lab-practice` folder. Use the R Notebook to follow along and experiment with the dataset. Do not just copy the instructions below but type them by hand instead. Also note that it is your responsibility to make sure any of the libraries used below (as recognized from them being loaded with `library`) are actually installed (with `install.packages`).

You can find a data of resale transactions going back to 1990 on the data.gov.sg website. For now, we will just consider the data from 2015-now so you can download this data set directly from https://www.dropbox.com/s/xorwcqe4h7d1u7q/resale-flat-prices-based-on-registration-date-from-jan-2015-onwards.csv?dl=1.

As we will use this data across several blocks, you should create a `data/` folder at the root of your project to hold this data.

A short note on the `here` package. Rmarkdown documents are always executed in the context of your their current directory. Move a document all of the sudden it might not be able to find the files it's using anymore! This is why we use `here`: it will always point to the root of our project folder.

Once the data is saved, let's read it in.

```
sales <- read_csv(here::here("data/hdb_resale_2015_onwards.csv"))
```

Use `glimpse()` and `View()` to inspect the resulting tibble. As you'll see the data is relatively clean.

```
sales %>% glimpse()
```

```
## Observations: 79,100
## Variables: 11
## $ month               <chr> "2015-01", "2015-01", "2015-01", "2015-01", "20...
```

```
## $ town                <chr> "ANG MO KIO", "ANG MO KIO", "ANG MO KIO", "ANG ...
## $ flat_type           <chr> "3 ROOM", "3 ROOM", "3 ROOM", "3 ROOM", "3 ROOM...
## $ block               <chr> "174", "541", "163", "446", "557", "603", "709"...
## $ street_name         <chr> "ANG MO KIO AVE 4", "ANG MO KIO AVE 10", "ANG M...
## $ storey_range        <chr> "07 TO 09", "01 TO 03", "01 TO 03", "01 TO 03",...
## $ floor_area_sqm      <dbl> 60, 68, 69, 68, 68, 67, 68, 68, 67, 68, 67, 68,...
## $ flat_model          <chr> "Improved", "New Generation", "New Generation",...
## $ lease_commence_date <dbl> 1986, 1981, 1980, 1979, 1980, 1980, 1980, 1981,...
## $ remaining_lease     <dbl> 70, 65, 64, 63, 64, 64, 64, 65, 62, 69, 60, 64,...
## $ resale_price        <dbl> 255000, 275000, 285000, 290000, 290000, 290000,...
```

There are just a few adjustments we need to make.

- `month` should probably be converted to a date
- `flat_type`, `storey_range` and `flat_model` are currently `chr` columns but can better be represented as `factor` variables as they are categorical variables.

```
library(lubridate)
library(forcats)

sales %>%
  mutate(month = ymd(month, truncated = 1),
         flat_type = as_factor(flat_type),
         storey_range = as_factor(storey_range),
         flat_model = as_factor(flat_model))
```

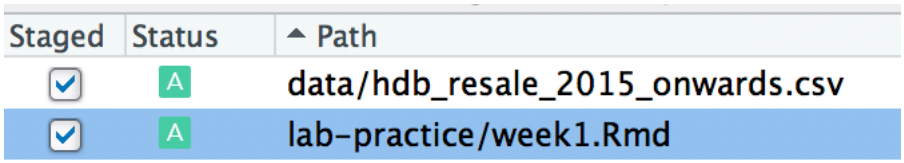
```
## # A tibble: 79,100 x 11
##    month      town  flat_type block street_name storey_range floor_area_sqm
##    <date>     <chr> <fct>     <chr> <chr>       <fct>                 <dbl>
##  1 2015-01-01 ANG ~ 3 ROOM    174   ANG MO KIO~ 07 TO 09                 60
##  2 2015-01-01 ANG ~ 3 ROOM    541   ANG MO KIO~ 01 TO 03                 68
##  3 2015-01-01 ANG ~ 3 ROOM    163   ANG MO KIO~ 01 TO 03                 69
##  4 2015-01-01 ANG ~ 3 ROOM    446   ANG MO KIO~ 01 TO 03                 68
##  5 2015-01-01 ANG ~ 3 ROOM    557   ANG MO KIO~ 07 TO 09                 68
##  6 2015-01-01 ANG ~ 3 ROOM    603   ANG MO KIO~ 07 TO 09                 67
##  7 2015-01-01 ANG ~ 3 ROOM    709   ANG MO KIO~ 01 TO 03                 68
##  8 2015-01-01 ANG ~ 3 ROOM    333   ANG MO KIO~ 01 TO 03                 68
##  9 2015-01-01 ANG ~ 3 ROOM    109   ANG MO KIO~ 01 TO 03                 67
## 10 2015-01-01 ANG ~ 3 ROOM    564   ANG MO KIO~ 13 TO 15                 68
## # ... with 79,090 more rows, and 4 more variables: flat_model <fct>,
## #   lease_commence_date <dbl>, remaining_lease <dbl>, resale_price <dbl>
```

Now it is time to get our cheatsheets out. Last term we gained experience with a suite of `tidyverse` staples to manipulate tables in different ways. The most important of these are probably `filter`, `arrange`, `group_by`, and `summarize`.

Try to put these to practice to answer the following questions:

- What is the earlier lease commencement date and where are these flats located?
- What are the largest HDB flats in Singapore? How much did they sell for?
- What is the most expensive flat in Punggol?
- Which town has, on average, the largest flats (by floor area)?
- Which town has, on average, the cheapest flats per square meter?

Answer each of these questions in a separate code chunk. When you are finished, save your RMarkdown document. You're now ready to make your first 'real' commit.



Walk through these steps – `stage` your changes, `commit` the staged changes (with an appropriate commit message), and finally `push` your local commits up to the repository on github.com.