Teaching Data Science Using Jupyter Notebooks and Binder

Brian Kim
University of Maryland

Graham Henke
New York University

Motivation

Teach **data science tools** such as **Python** and **SQL** to beginners with no previous coding experience without requiring them to install anything on their own computers.

Ideally, it should also be **easy for instructors** to setup and deliver the material.

Vision:

“Our goal is to change the empirical foundation of social science, statistical and public agencies in the United States and transform understanding of how our society works. We are a fast growing university-based startup that has already created dozens of pilot projects, worked with over 100 agencies—federal, state and local—and trained over 250 agency staff.”

Coleridge Initiative

Data is more abundant than ever, but the capability of **government agencies** for analyzing them has lagged behind. Challenges include:

- Lack of training among government agency staff and social scientists
- Hesitation in sharing data with others
- Insufficient data infrastructure

Policy Research in the Modern World

The **Administrative Data Research Facility (ADRF)** is a FedRAMP certified, secure cloud-based environment designed to allow agencies to share data.

The **Applied Data Analytics Program** trains agency staff in data science techniques to take advantage of the data that is available to them.

Applied Data Analytics Program

Data analysis training using administrative data

- Six or ten day programs
- Aimed at government agency staff
- Provides training in modern data analysis techniques

Themes include:

- Criminal Justice
- Welfare Programs
- High Need Populations
- And more!

Applied Data Analytics Program

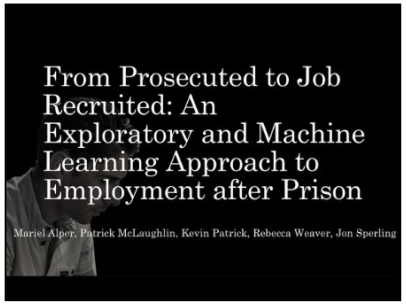
Project-based training program that covers topics such as:

- Text Analysis
- Network Analysis
- Machine Learning
- Inference
- Privacy and Confidentiality

Projects

Addressing Recidivism: Technical Violations

Mommy Don't Go: Recidivism of Mothers



From Prosecuted to Job Recruited: Employment after Prison

Source: https://coleridgeinitiative.org

Applied Data Analytics Program

One key hurdle: **Programming skills**

The program is taught using **Python and SQL,** but many participants have no prior programming experience.

Introduction to Python and SQL

Solution: **Introduction to Python and SQL** is provided in the four weeks leading up to the course as an asynchronous online course.

- Introduction to Python and Pandas
- Functions, Loops, and Visualizations
- Introduction to SQL
- Joins; Using SQL with Python

Teaching Introductory Material

Challenges:

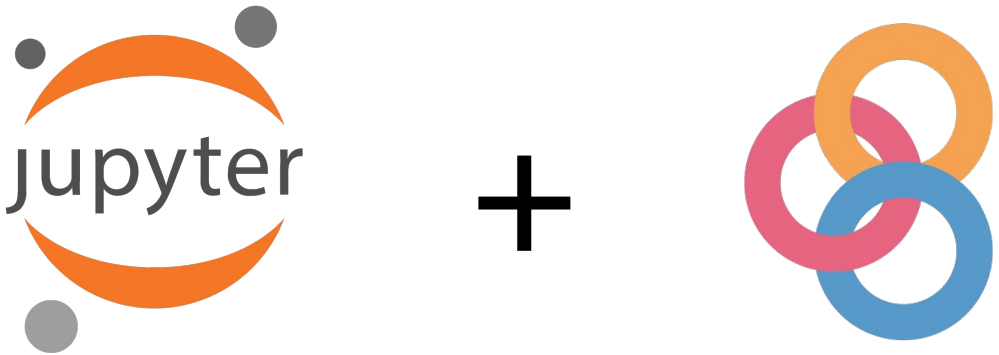
- **Remote.** We can't troubleshoot issues on each individual computer.
- **Government computers.** Installing new software is NOT trivial.
- **Coding beginners.** Content must be approachable from the very beginning

How do we address these?

Jupyter Notebooks and Binder

The material needs to be **easy to follow** and **easy to access.**

To do this, we use **Jupyter notebooks** and **Binder.**



Jupyter Notebooks

Jupyter notebooks provide a user-friendly, approachable experience that allows us to combine descriptive text with coding examples.

Python Workbook 1: What are the characteristics in the number of jobs by census block? ¶

In these workbooks, we will start with a motivating question, then walk through the process we need to go through in order to answer the motivating question. Along the way, we will walk through various Python commands and develop skills as we work towards answering the question.

As you work, there will be headers that are in **GREEN**. These indicate locations where there is an accompanying video. This video may walk through the steps or expand on the topics discussed in that section. Though it isn't absolutely necessary to watch the video while working through this notebook, we highly recommend watching them at least once on your first time through.

If you have not yet watched the "Introduction to Jupyter Notebooks" video, watch it before you proceed!

You will also run into some headers that are in **RED**. These headers indicate a checkpoint to practice writing the code yourself. You should stop at these checkpoints and try doing the exercises and answering the questions posed in these sections.

NOTE: When you open a notebook, make sure you run each cell containing code from the beginning. Since the code we're writing builds on everything written before, if you don't make sure to run everything from the beginning, some things may not work.

In each of the workbooks, we will start out with a motivating question. Here, we'll introduce the data that we'll work with, which will lead into our motivating question for this workbook.

Longitudinal Employer-Household Dynamics (LEHD) Data

In these workbooks, we will be using LEHD data. These are public-use data sets containing information about employers and employees. Information about the LEHD Data can be found at https://lehd.ces.census.gov/.

We will be using the LEHD Origin-Destination Employment Statistics (LODES) datasets in our applications in this workbook. Each state has three main types of files: Origin-Destination data, in which job totals are associated with a home and work Census block pair, Residence Area Characteristic data, in which job totals are by home Census block, and Workplace Area Characteristic data, in which job totals are by workplace Census block. In addition to these three, there is a "geographic crosswalk" file with descriptions of the Census Blocks as they appear the in the LODES datasets.

Cells

Loading Libraries

We first load a few libraries. These libraries are essentially bundles of useful tools that can help us do specific tasks. In this case, we're going to be bringing in NumPy and Pandas, which are specifically useful for computing and data analysis. Don't worry too much about the specifics of libraries for now, just that we need to include the first few lines of code below if we want to use many of the tools described in this workbook.

If you'd like to read more about the libraries that we're loading here, see the following links: for NumPy and for Pandas

In [ ]:
```
# First, we import packages
# Note that everything after a "#" in Python will be ignored, so you can use it to write comments
import numpy as np # NumPy (Numerical Python) for scientific computing
import pandas as pd # Pandas, for data analysis tools
```

Notice that we loaded the libraries with aliases (e.g. `as np`). This is just to make it easier to use them later on. Whenever you see `np`, read it as `numpy`. The same goes for `pd` and `pandas`.

Jupyter Notebooks

Jupyter notebooks allow us to **describe the process** and give students a way to **run code themselves in the same document.**

But just using Jupyter notebooks isn't enough -- we want to make sure that there's as little initial setup cost as possible, including with **installation**.

Binder

Binder takes a **Git repository** and **builds the environment** that you need to run all the code in the Jupyter notebooks within that repository.

Setting this up might seem daunting … but it’s NOT.

Steps for Using Binder

1. Create **Jupyter notebooks** with the instructional material.
2. Put Jupyter notebooks (and any associated datasets) on a **GitHub repository**.
3. Create a **dependency file** (e.g. requirements.yml)
4. Go to **Binder** (https://mybinder.org) and fill in details.
5. Hit **launch**!

Step 1: Make Jupyter Notebooks

Download and install at https://jupyter.org/install.

Create notebooks with instructional material.

Loading Libraries

We first load a few libraries. These libraries are essentially bundles of useful tools that can help us do specific tasks. In this case, we're going to be bringing in NumPy and Pandas, which are specifically useful for computing and data analysis. Don't worry too much about the specifics of libraries for now, just that we need to include the first few lines of code below if we want to use many of the tools described in this workbook.

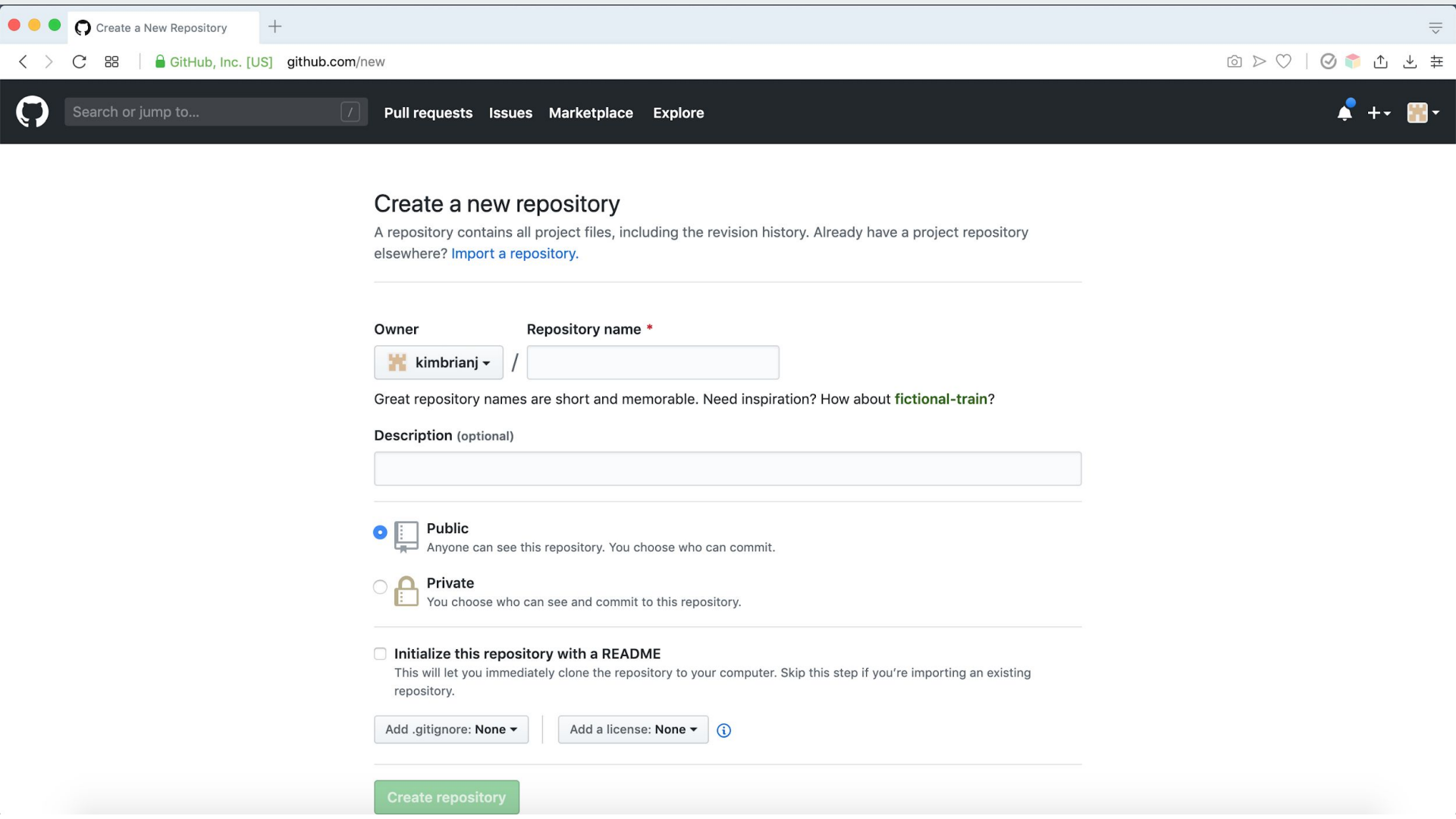
If you'd like to read more about the libraries that we're loading here, see the following links: for NumPy and for Pandas

In [ ]:
```
# First, we import packages
# Note that everything after a "#" in Python will be ignored, so you can use it to write comments
import numpy as np # NumPy (Numerical Python) for scientific computing
import pandas as pd # Pandas, for data analysis tools
```

Notice that we loaded the libraries with aliases (e.g. `as np`). This is just to make it easier to use them later on. Whenever you see `np`, read it as `numpy`. The same goes for `pd` and `pandas`.

Step 2: GitHub

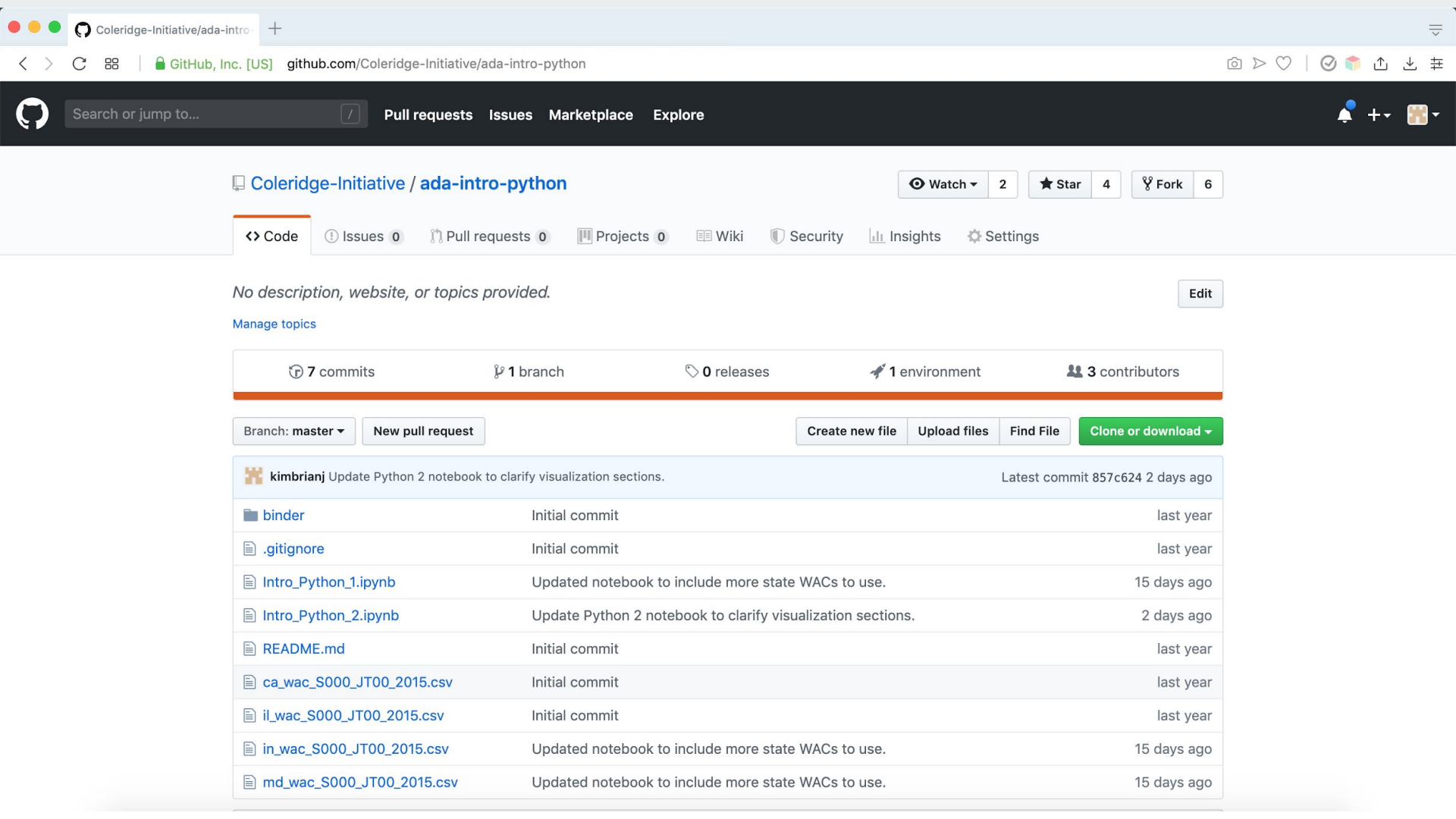
Great if you use **git** from the beginning, but even if not, you can just make a repository from scratch and just upload all the files manually.


Create a New Repository
GitHub, Inc. [US]
github.com/new
Search or jump to...
Pull requests
Issues
Marketplace
Explore
Create a new repository
A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.
Owner
Repository name *
kimbrianj
/
Great repository names are short and memorable. Need inspiration? How about fictional-train?
Description (optional)
Public
Anyone can see this repository. You choose who can commit.
Private
You choose who can see and commit to this repository.
Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.
Add .gitignore: None
Add a license: None
Create repository

GitHub

Binder uses the material on **GitHub**, but you don't need to know **Git** to use **Binder**.

For the purposes of Binder, you can treat it simply as a file storage system (though it is definitely useful to use the git features).


Coleridge-Initiative/ada-intro
GitHub, Inc. [US]
github.com/Coleridge-Initiative/ada-intro-python
Search or jump to...
Pull requests
Issues
Marketplace
Explore
Coleridge-Initiative / ada-intro-python
Watch 2
Star 4
Fork 6
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Security
Insights
Settings
No description, website, or topics provided.
Edit
Manage topics
7 commits
1 branch
0 releases
1 environment
3 contributors
Branch: master
New pull request
Create new file
Upload files
Find File
Clone or download
kimbrianj Update Python 2 notebook to clarify visualization sections.
Latest commit 857c624 2 days ago
binder Initial commit last year
.gitignore Initial commit last year
Intro_Python_1.ipynb Updated notebook to include more state WACs to use. 15 days ago
Intro_Python_2.ipynb Update Python 2 notebook to clarify visualization sections. 2 days ago
README.md Initial commit last year
ca_wac_S000_JT00_2015.csv Initial commit last year
il_wac_S000_JT00_2015.csv Initial commit last year
in_wac_S000_JT00_2015.csv Updated notebook to include more state WACs to use. 15 days ago
md_wac_S000_JT00_2015.csv Updated notebook to include more state WACs to use. 15 days ago

Step 3: Dependency file

environment.yml

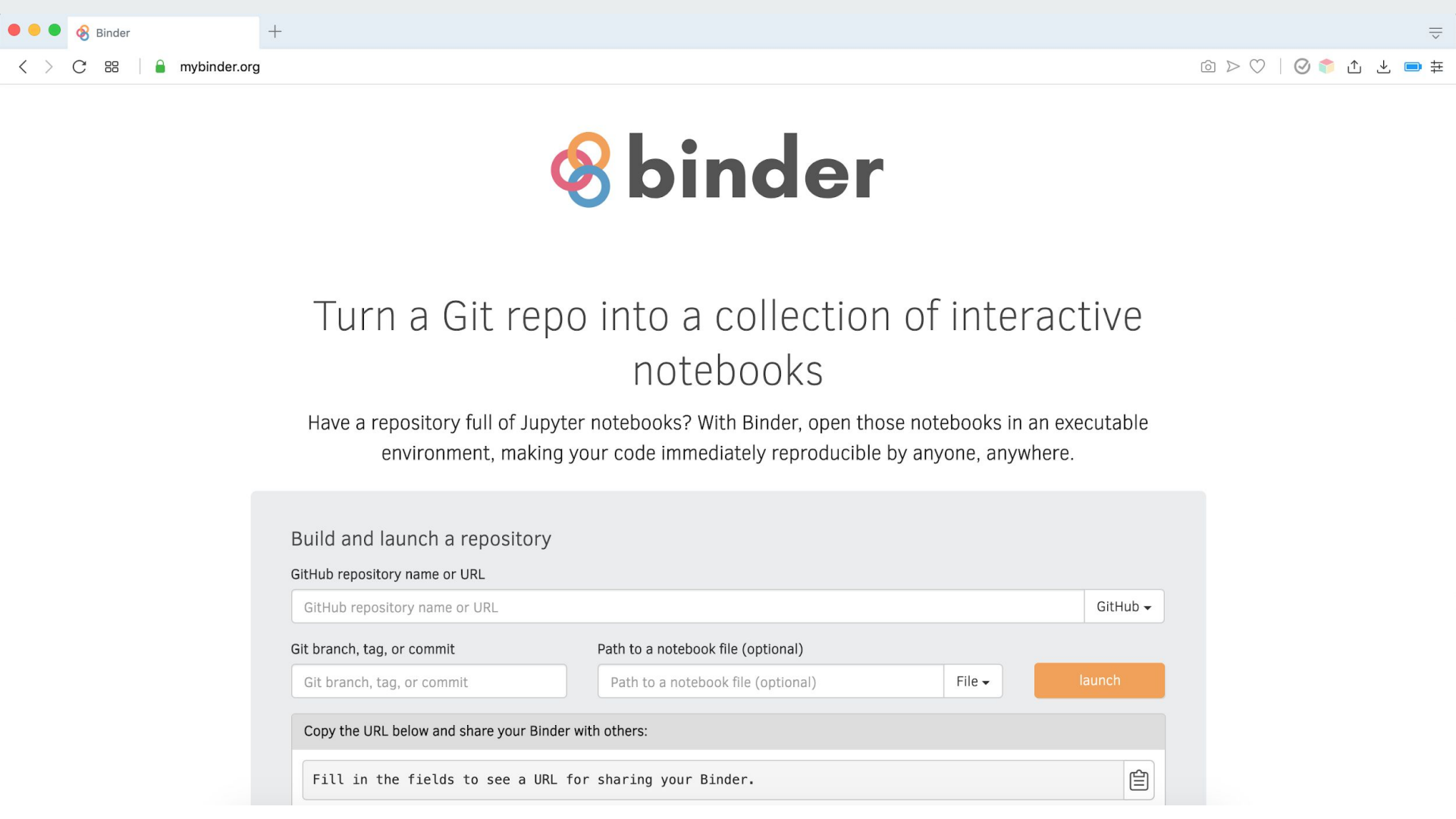
```yaml
name: ada-intro-python
channels:
  - conda-forge
dependencies:
  - matplotlib
  - ipywidgets
  - beakerx

```

Step 4: Binder

Go to https://mybinder.org.

Fill in the information.


Binder
mybinder.org
binder
Turn a Git repo into a collection of interactive notebooks
Have a repository full of Jupyter notebooks? With Binder, open those notebooks in an executable environment, making your code immediately reproducible by anyone, anywhere.
Build and launch a repository
GitHub repository name or URL
GitHub repository name or URL
GitHub
Git branch, tag, or commit
Git branch, tag, or commit
Path to a notebook file (optional)
Path to a notebook file (optional)
File
launch
Copy the URL below and share your Binder with others:
Fill in the fields to see a URL for sharing your Binder.

Introduction to Python

GitHub link: https://github.com/Coleridge-Initiative/ada-intro-python

You can also link to a **specific notebook**:
https://mybinder.org/v2/gh/Coleridge-Initiative/ada-intro-python/master?filepath=Intro_Python_1.ipynb

Introduction to SQL

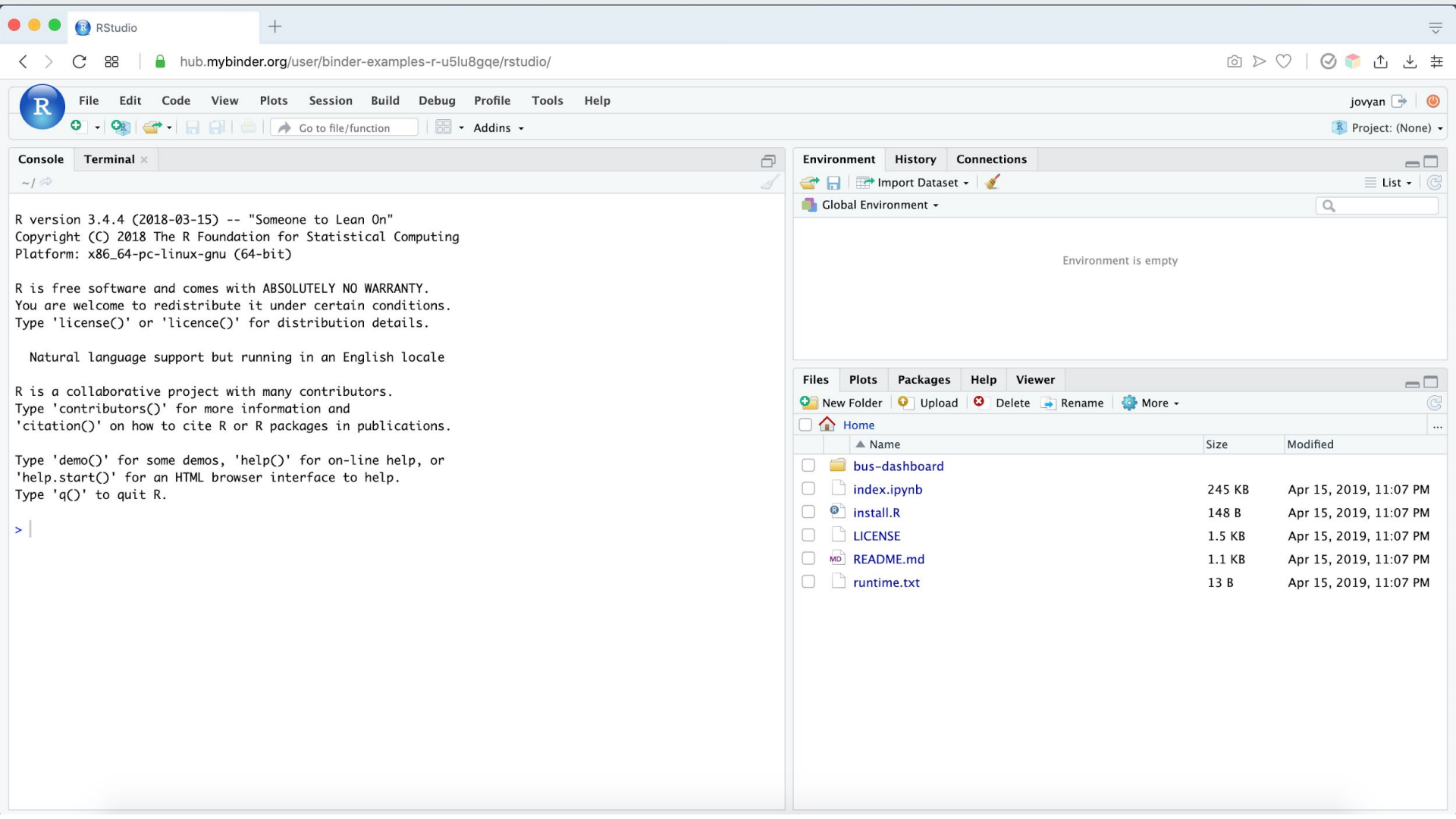
SQL is a little bit more tricky, because it requires a database, but it’s doable as well.

https://github.com/Coleridge-Initiative/ada-intro-sql

Other Possibilities

Recall: Jupyter notebooks were designed to be compatible with many coding languages, as long as you have the appropriate kernel.

You can even do this with **RStudio**.


RStudio
hub.mybinder.org/user/binder-examples-r-u5lu8gqe/rstudio/
File Edit Code View Plots Session Build Debug Profile Tools Help
jovyan
Go to file/function
Addins
Project: (None)
Console Terminal
~/
R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
>
Environment History Connections
Import Dataset
List
Global Environment
Environment is empty
Files Plots Packages Help Viewer
New Folder Upload Delete Rename More
Home
Name Size Modified
bus-dashboard
index.ipynb 245 KB Apr 15, 2019, 11:07 PM
install.R 148 B Apr 15, 2019, 11:07 PM
LICENSE 1.5 KB Apr 15, 2019, 11:07 PM
README.md 1.1 KB Apr 15, 2019, 11:07 PM
runtime.txt 13 B Apr 15, 2019, 11:07 PM

Remarks

Updating course material is very quick and easy.

Easy setup for both instructors and students

No saving work

Does not scale up well

Alternatives

JupyterHub with cloud server

- Much more scalable and stable
- Higher setup cost for organizers and participants

Docker Container

- Requires setup by participants
- Might have OS problems

Alternatives (cont.)

Virtual Box

- Heavyweight
- Requires setup by participants

Others

- Let me know if you have any ideas!

When to use Binder

Use Binder when …

- You have a relatively small group of students/participants.
- Saving work isn't crucial (e.g. no project work).
- Audience is not technically advanced.
- Installation may be a barrier.
- You want to include additional interactive material.

Interact

Expressions

Programming languages are much simpler than human languages. Nonetheless, there are some rules of grammar to learn in any language, and that is where we will begin. In this text, we will use the Python programming language. Learning the grammar rules is essential, and the same rules used in the most basic programs are also central to more sophisticated programs.

Programs are made up of *expressions*, which describe to the computer how to combine pieces of data. For example, a multiplication expression consists of a `*` symbol between two numerical expressions. Expressions, such as `3 * 4`, are *evaluated* by the computer. The value (the result of *evaluation*) of the last expression in each cell, `12` in this case, is displayed below the cell.

```
3 * 4
```

```
12
```

The grammar rules of a programming language are rigid. In Python, the `*` symbol cannot appear twice in a row. The computer will not try to interpret an expression that differs from its prescribed expression structures. Instead, it will show a `SyntaxError` error. The *Syntax* of a language is its set of grammar rules, and a `SyntaxError` indicates that an expression structure doesn't match any of the rules of the language.

Acknowledgements

Thanks to everyone at the Coleridge Initiative, especially Clayton Hunter who makes sure everything runs as it should.

Thank you all for listening.