



Transitioning from Analytics to Data Science

ABIGAIL HADDAD



The Three-Minute Version

In data science, we're coding for transparency and replicability in addition to accuracy

Be able to talk about projects you did in technical and non-technical ways

Teaching yourself and debugging are two key parts of the job





Who I Am

Public Policy Ph.D.

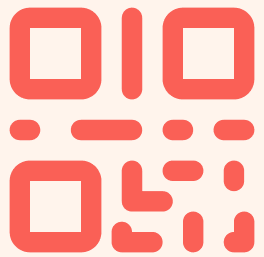
Research/stats/data science for the Department of Defense

Lead Data Scientist

Code for DC tomorrow if you want to talk about DC government data and how to make it easier to find

You can find this here: <https://github.com/abigailhaddad/slides/>

slido



Join at slido.com
#3805224

① Start presenting to display the joining instructions on this slide.

slido



What's your primary tool for analyzing data?

ⓘ Start presenting to display the poll results on this slide.



Agenda

What to learn and how to learn it

Better coding practices

Resumes and interviewing





What to Learn

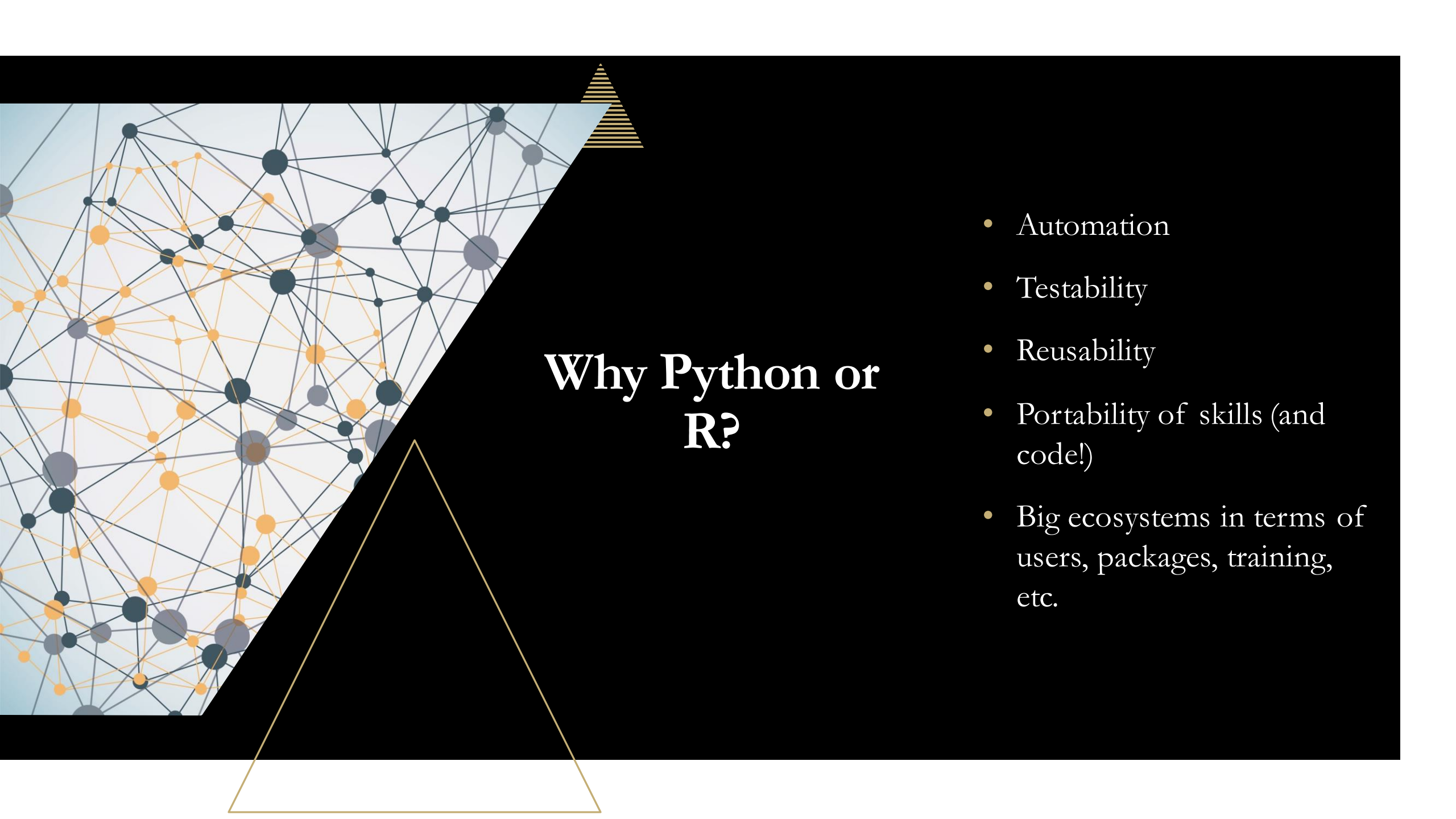
Python or R or both

SQL

Git

Statistics/probability/data analysis

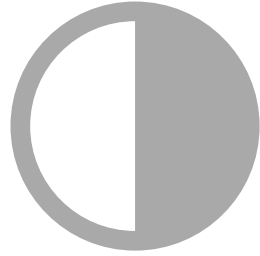




Why Python or R?

- Automation
- Testability
- Reusability
- Portability of skills (and code!)
- Big ecosystems in terms of users, packages, training, etc.

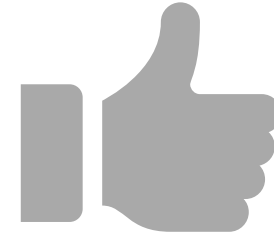
Python and R



Why this matters less now

Increased interoperability

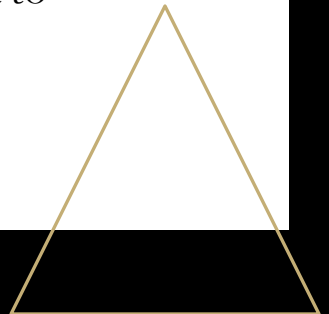
Increased functionality



How to choose?

Which do you like more?

What does the company/industry you want to work in use?





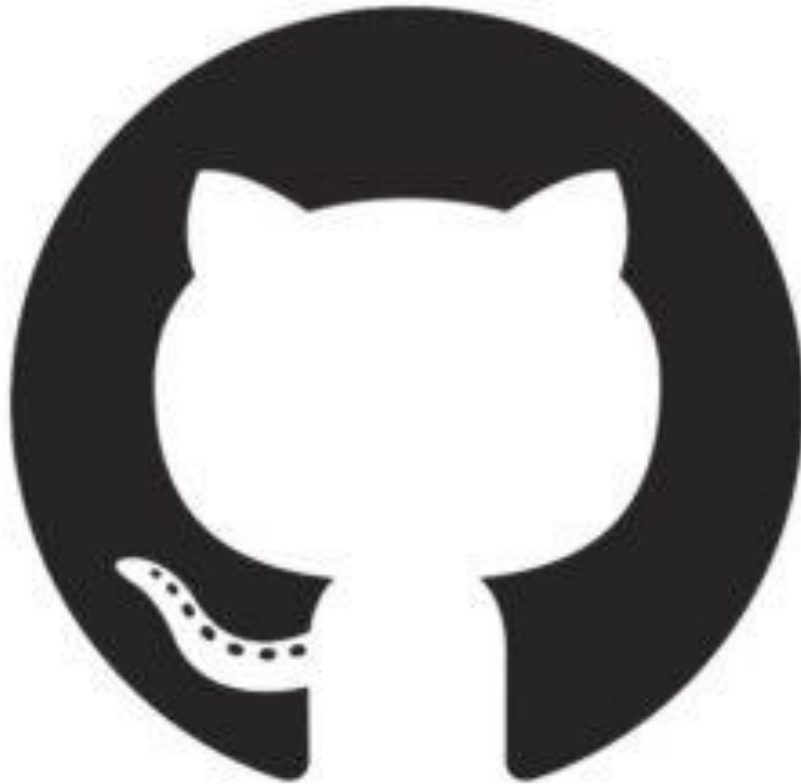
Relevant Dimensions for Picking How to Learn

Cost

Level of structure provided

Time commitment

Credential



Do Projects

-Anything you're interested in (**or can do at work**), but also:

- Data cleaning
- Data analysis
- Modelling
- Visualization

-Better coding practices

-GitHub



Everything is Broken Until It's Not

- A lot of coding is debugging
- Resources: Stack Overflow, documentation, GitHub, ChatGPT

Better Coding Practices: Not Just Nice But Part of the Job



ACCURATE



TRANSPARENT



REPEATABLE

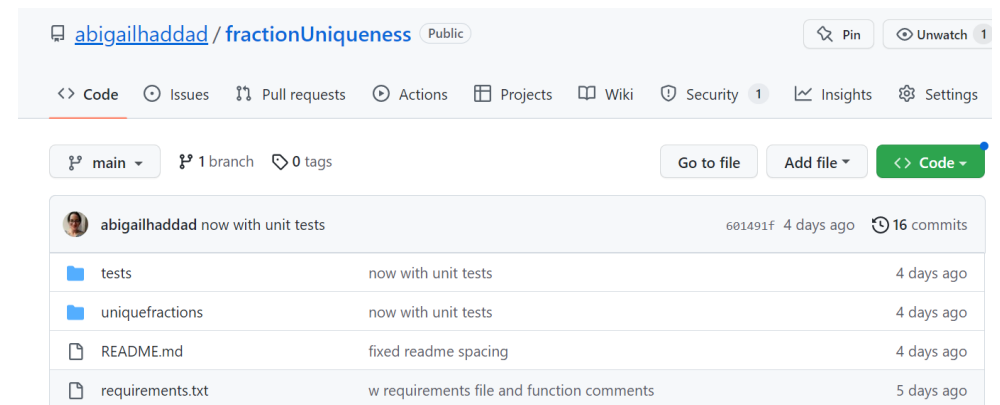
Version Control

WHY

- Avoid multiple copies floating around
- Let multiple people work on the same thing at once
- Easily revert back when you break something
- Compare versions

This is a benefit of working with code as opposed to Graphical User Interfaces

WHAT IT LOOKS LIKE



Documentation

WHAT DID YOU DO AND WHY?

- Readme files
- Inputs/outputs
- Function-level code

EXAMPLE

```
def genDFOfNumeratorsAndDenominators(max_denominator, min_denominator=0):  
    """this generates all of the possible numerator/denominator fractions that are between  
    zero and one, given the min and max denominators  
  
    Args:  
        max_denominator: maximum possible denominator  
        min_denominator (optional - default is 0): minimum possible denominator  
  
    Returns:  
        df: pandas df with all combinations of numerators and denominators  
  
    """  
    df = pd.DataFrame([[x, y] for x in range(0, max_denominator + 1)  
                        for y in range(min_denominator, max_denominator + 1) if y >= x])  
    df.columns=["Numerator", "Denominator"]  
    return(df)
```

Good-Enough Code

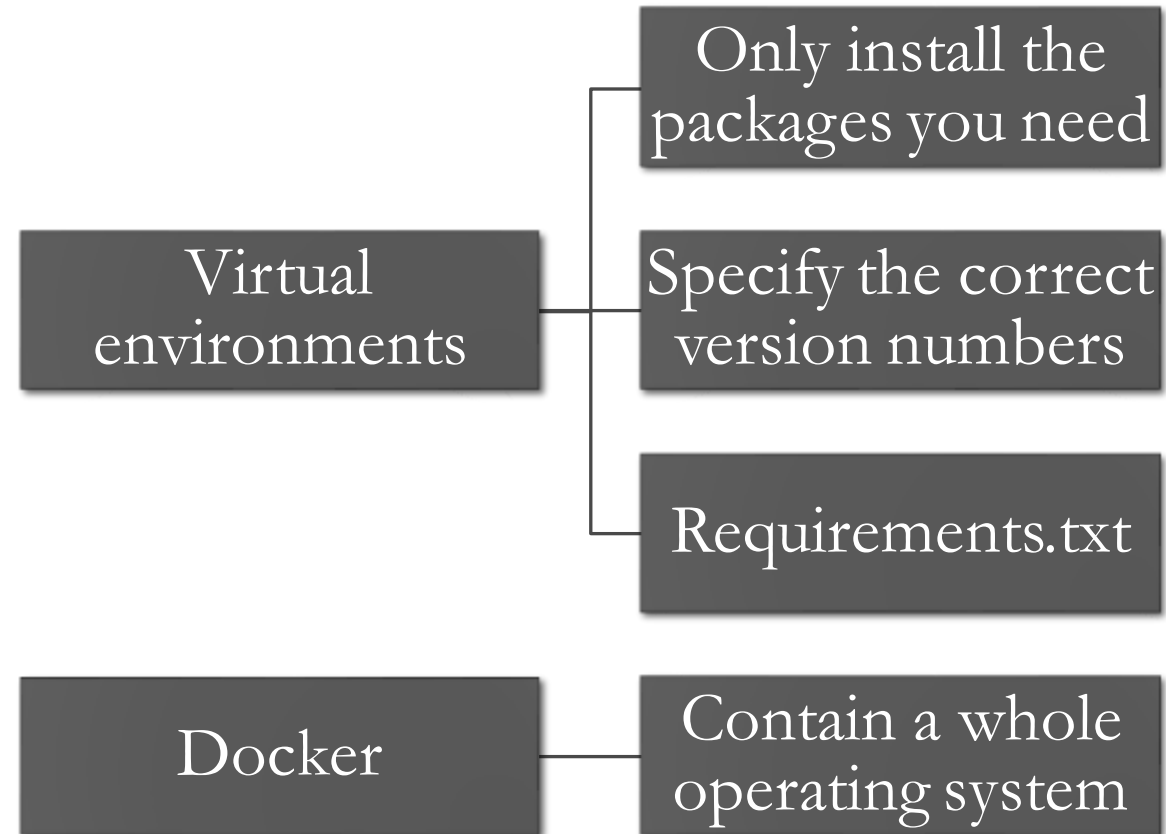
PRACTICES

- Code is in functions
- Each function does one thing
- Functions and variables named in useful ways
- No complicated workflows
- Don't repeat yourself

EXAMPLE

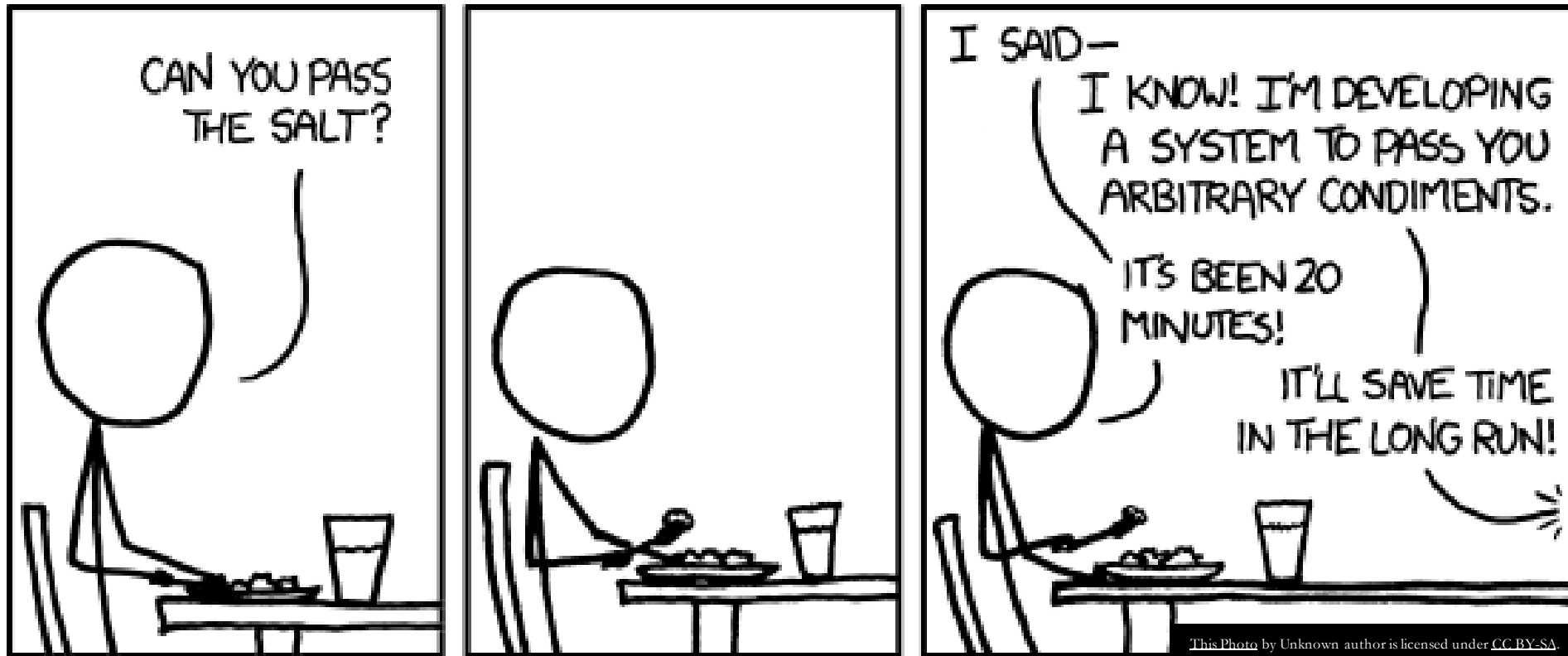
```
def genDFOfNumeratorsAndDenominators(max_denominator, min_denominator=0):  
    """this generates all of the possible numerator/denominator fractions that are between  
    zero and one, given the min and max denominators  
  
    Args:  
        max_denominator: maximum possible denominator  
        min_denominator (optional - default is 0): minimum possible denominator  
  
    Returns:  
        df: pandas df with all combinations of numerators and denominators  
  
    """  
    df = pd.DataFrame([[x, y] for x in range(0, max_denominator + 1)  
                        for y in range(min_denominator, max_denominator + 1) if y >= x])  
    df.columns=["Numerator", "Denominator"]  
    return(df)
```


Isolation/Containerization



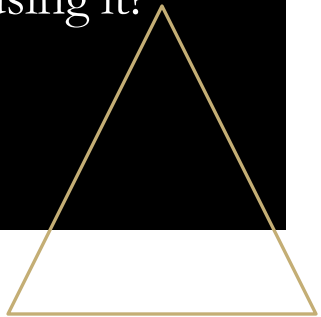
Am I Overengineering?

Don't Worry About This Yet





Data Work on Your Resume

- What language did you use? What packages?
 - Conceptually, what were you doing? Descriptive statistics? A linear regression? Markov modeling? A random forest?
 - What were the outputs, in plain language? A paper? A graph? A dashboard?
 - What problem were you solving?
 - What was this for? Did you get something published? Did it get used in the world?
 - Is there anything about your processes that indicates ‘good coding practices’? Like, did you use git? Did you write documentation? Test cases? Did your code go in a module? Are other people using it?
- 



Talking About What You Built



- What were you were trying to do?
- How did you do it?
- What does success mean (especially for models)
- Do you understand how the technical choices you made relate to the outcomes you're interested in?
- What was it used for?

slido



Audience Q&A Session

① Start presenting to display the audience questions on this slide.

Contact Info

<https://github.com/abigailhaddad>

abigail.Haddad@gmail.com