



Data + Donuts 12/10/19

Tiffany Chu and Ian Rose

| Citywide Data Science and Predictive Analytics

Partner with various City departments to do multi-departmental data analytics projects

- Transportation
- Office of Finance
- Street Services
- Sanitation
- Housing + Community Investment Development
- City Planning
- Mayor's initiatives

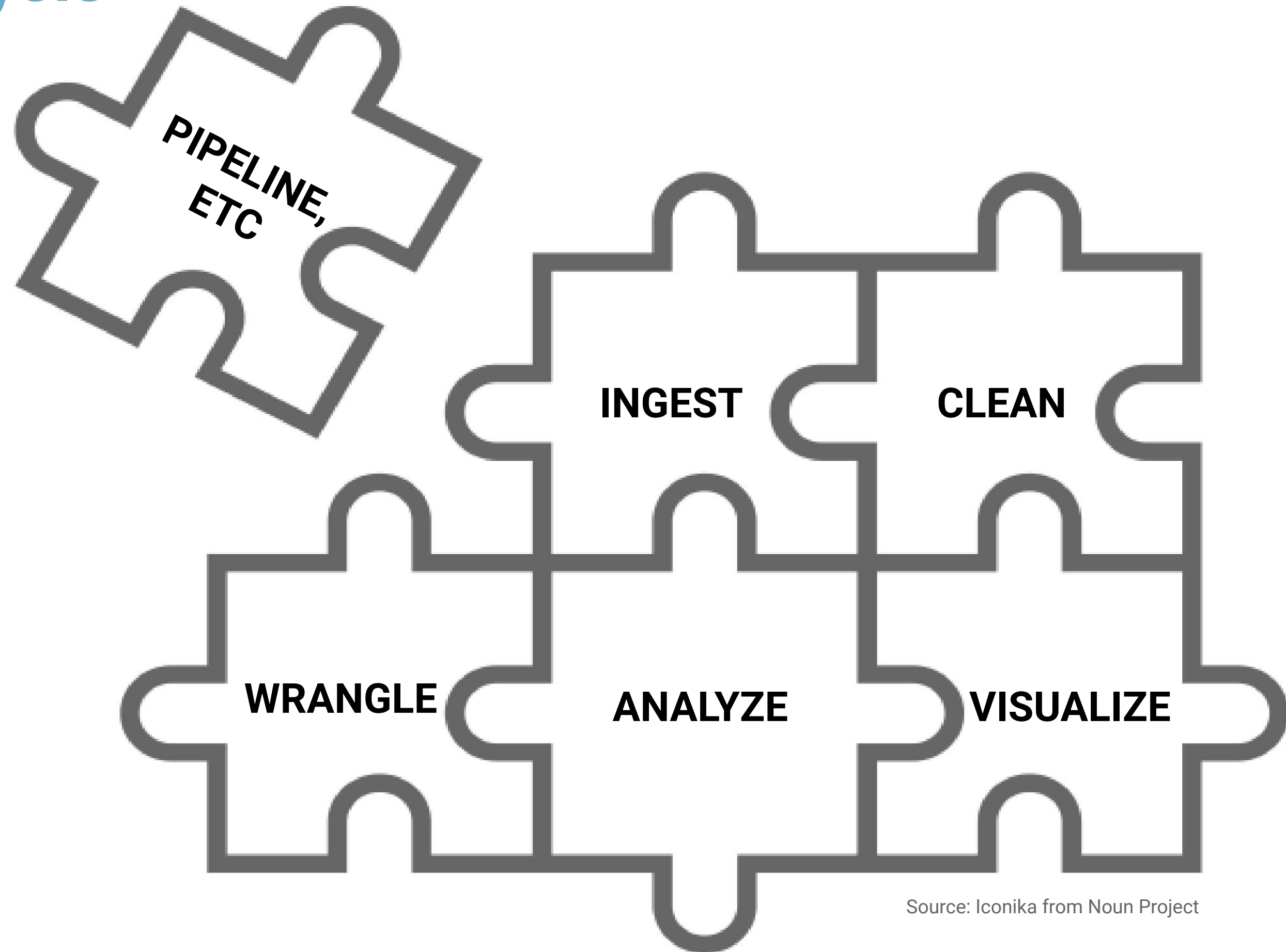
| Best Practices

Our hard-earned lessons and standard we're striving for:

- Goal: Reproducibility
- GitHub
- Data Pipelines
- Data Management
- Shared platform for analysis (JupyterHub)
- Tutorials

<https://cityoflosangeles.github.io/best-practices/>

| Data Analysis



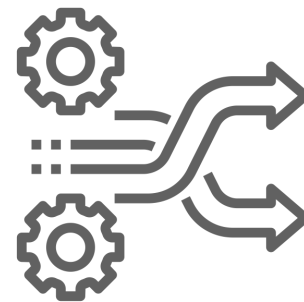
Source: Iconika from Noun Project

| Motivation

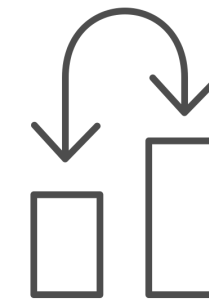
- What's the research question?
 - Policy interventions, factors / mechanisms, outcomes
- So many datasets, mix & match
- Merge data to compare and see what's going on



Observe trends



Current resource allocation
vs “need”-driven allocation



Normalize by population
or area

| Ingest and Clean

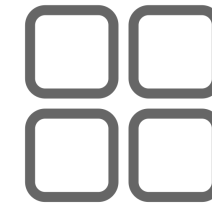
- Multiple data sources
- Identify the unit of analysis



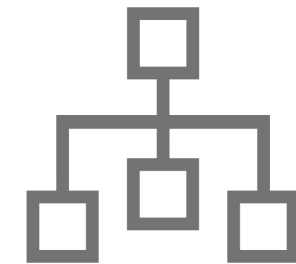
Geography
tract, council district



Time
year, quarter



Other Category
treatment, intervention



Mixed
tract-year,
tract-treatment-year

- Clean each data source to the common unit (least common factor)
 - Aggregate
 - Spatial join + dissolve

Merge

- **one-to-one (1:1)**
merge on **GEOID** and **Year**
GEOID and **Year** appear once in both dfs
- **many-to-one (m:1)**
merge on **GEOID**
GEOID appears multiple times in left df;
once in right df
- **one-to-many (1:m)**
merge on **Year**
Year appears once in left df; multiple times
in right df

collisions			traffic volume		
GEOID	Year	# collisions	GEOID	Year	traffic vol
A	2018	10	A	2018	2,000
A	2019	8	A	2019	1,500
B	2018	15	B	2018	2,500

collisions			GEOID characteristics	
GEOID	Year	# collisions	GEOID	area (sqmi)
A	2018	10	A	5
A	2019	8	B	4
B	2018	15		

collisions by year		traffic volume by quarter		
Year	# collisions	Year	Qtr	traffic vol
2018	30	2018	1	500
2019	40	2018	2	700
		2019	1	450

| Crosswalks

- Correspondence tables that link elements together

tracts to council districts crosswalk
lists the CD associated with each tract

GEOID	CD
A	1
B	5
C	14
D	5

streets to tracts crosswalk
lists the tract associated with each street segment

Segment	Tract
Wilshire1	1000
Main4	2000
Hoover2	3000
Pico5	4000

- Spatial join + clipping to create crosswalk
- Facilitates the merging and aggregating across multiple dfs

| Putting it Together

```
df1 = pd.merge(collisions, traffic_volume, on = ['GEOID', 'Year'],  
               how = 'inner', validate = '1:1')
```

collisions

GEOID	Year	# collisions
A	2018	10
A	2019	8
B	2018	15
C	2017	20

traffic volume

GEOID	Year	traffic vol
A	2018	2,000
A	2019	1,500
B	2018	2,500
C	2017	2,500

df1

GEOID	Year	# collisions	traffic vol
A	2018	10	2,000
A	2019	8	1,500
B	2018	15	2,500
C	2017	20	2,500

1:1 merge on GEOID, Year

| Putting it Together

```
df2 = pd.merge(df1, tract_characteristics, on = 'GEOID',  
               how = 'inner', validate = 'm:1')
```

df1

GEOID	Year	# collisions	traffic vol
A	2018	10	2,000
A	2019	8	1,500
B	2018	15	2,500
C	2017	20	2,500

tract characteristics

GEOID	area
A	5
B	4
C	6

df2

GEOID	Year	# collisions	traffic vol	area
A	2018	10	2,000	5
A	2019	8	1,500	5
B	2018	15	2,500	4
C	2017	20	2,500	6

m:1 on GEOID

| Putting it Together

```
df3 = pd.merge(df2, crosswalk, on = 'GEOID',  
               how = 'inner', validate = 'm:1')
```

df2

GEOID	Year	# collisions	traffic vol	area
A	2018	10	2,000	5
A	2019	8	1,500	5
B	2018	15	2,500	4
C	2017	20	2,500	6

crosswalk

GEOID	CD
A	1
B	5
C	14

df3

GEOID	Year	# collisions	traffic vol	area	CD
A	2018	10	2,000	5	1
A	2019	8	1,500	5	1
B	2018	15	2,500	4	5
C	2017	20	2,500	6	14

m:1 on GEOID

df3 is the merged combination of collisions, traffic_volume, tract_characteristics, and crosswalk

| Additional Resources

- Merging dfs

- <https://guides.nyu.edu/quant/merge>
(Tableau, SPSS, JMP, Stata, SAS, R, Matlab, Python)
- <https://www.shanelynn.ie/merge-join-dataframes-python-pandas-index-1/>
- <https://towardsdatascience.com/why-and-how-to-use-merge-with-pandas-in-python-548600f7e738>

- Clip congressional_districts to City of LA boundary and create new geometry column

- `df = gpd.sjoin(congressional_districts, city_boundary, how = 'inner', op = 'intersects')`
- `boundary = city_boundary.geometry.iloc[0]`
- `df['new_geom'] = df[df.intersects(boundary)].intersection(boundary)`

| Infrastructure for Civic Data Teams

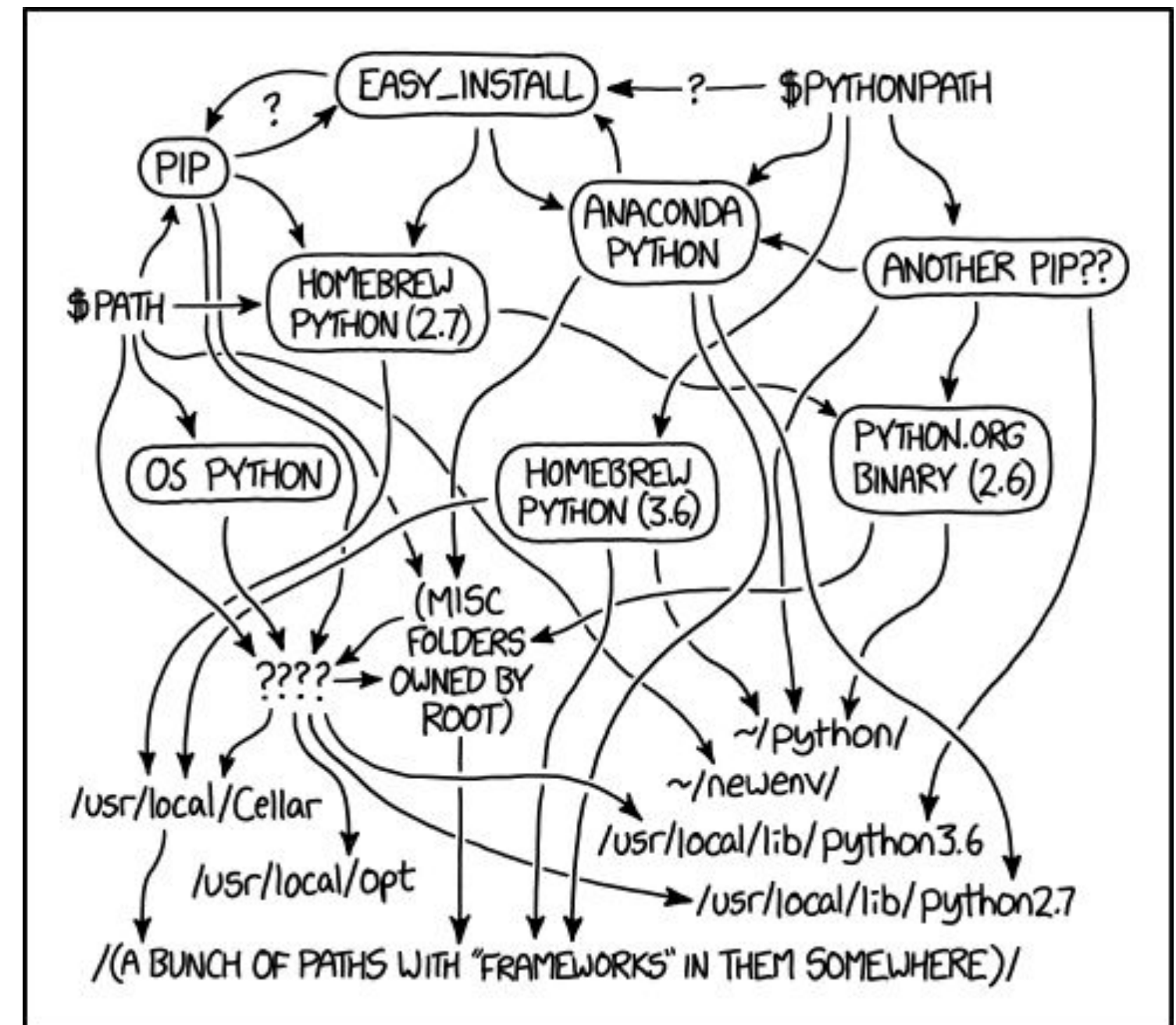
Data analysis is hard: how do we set up the infrastructure for our analysts to spend more time doing that, and less time fighting their hardware/software?

| Best Practices: Reproducible Environments

Reproducibility is an unsolved problem in science!

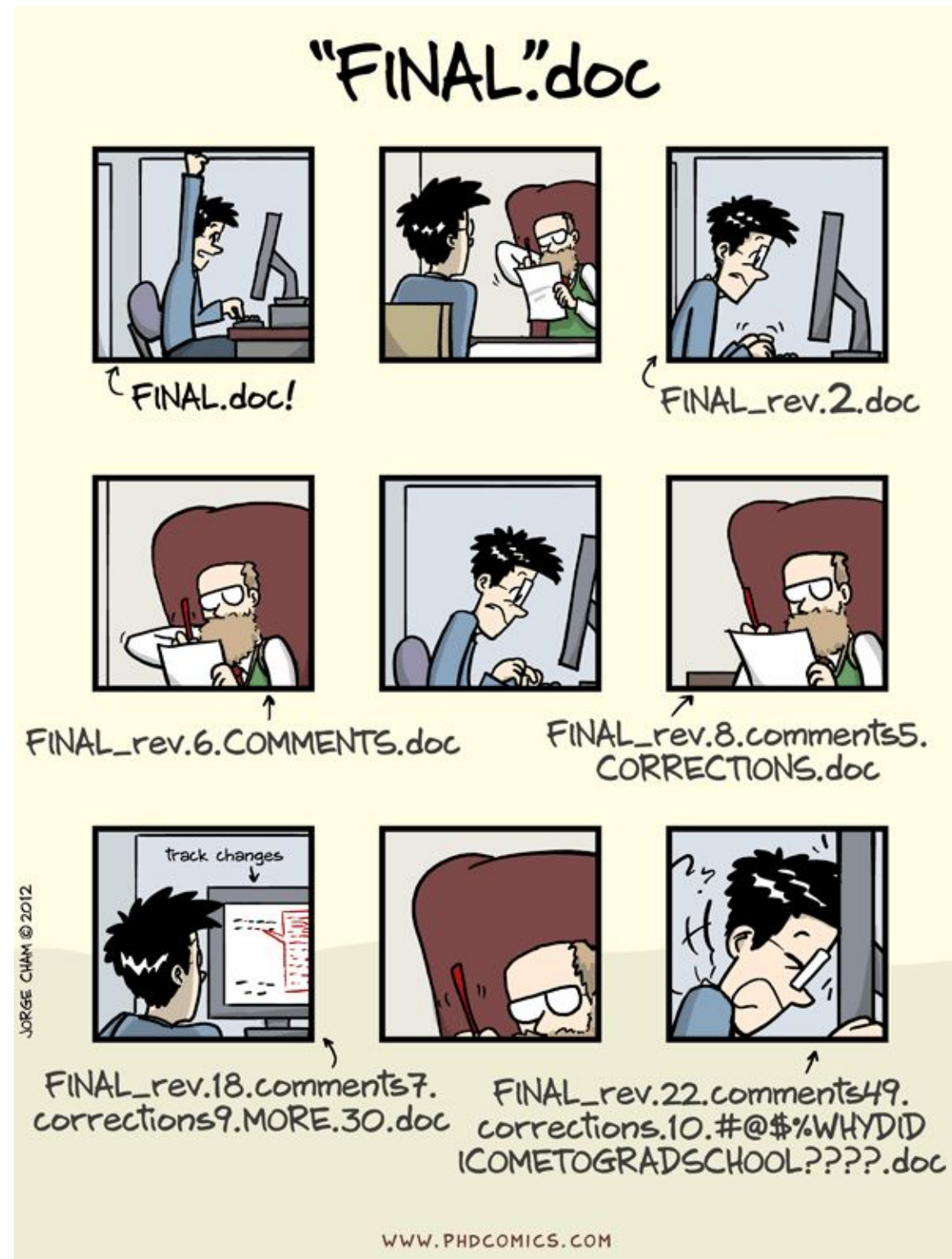
Tools that can help:

- Documentation, documentation, documentation
- Dockerfiles / requirements.txt / environment.yml
- Shared cloud compute
- CI/CD



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED
THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

| Best Practices: Version Control



| Best Practices: Continuous Integration

AKA: “catch errors before they are a problem”



| Cloud-Native Infrastructure

AKA “someone else’s computer”

Pros:

- Scalable
- Reproducible
- Reliable

Cons:

- Can have a steep learning curve
- Networking can be difficult
- “Oops, my cluster is down, now I can’t work.”



| Cloud-Friendly Formats

Use file formats that are open, standardized, and easy to share on cloud resources:

- Parquet
 - GeoJSON
 - GeoTIFF
 - Zarr
 - CSV*
 - Shapefile*
- * Caveats apply

Try to avoid:

- Excel (cf. [The Excel Error that Changed History](#))
- PDF (please please please)
- ../../ian-rose/files/some-forgotten-file.csv

| Literate computing: Jupyter and RMarkdown

The screenshot displays a Jupyter Notebook environment with a file explorer on the left and a main workspace divided into a code editor and an output view.

Code Editor Content:

Neighborhood level scooter usage

Let's collect a snapshot of the scooter trips available for a given time window in a given neighborhood within Los Angeles.

The datasets are large (order tens of gigabytes) and stored in a PostGIS database. We will use ibis for assembling geospatial queries for the database, and GeoPandas for analysing the in-memory results of the query.

Make some relevant imports

```
[1]: import contextily as ctx
import ibis
import intake
import geopandas as gpd
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
import pandas as pd
from util import clean_geometry
import ipywidgets as widgets
```

cat = intake.open_catalog('catalog.yml')

Load the neighborhood and street data from the catalog

```
[2]: neighborhoods = cat.geohub.neighborhood_councils.read()

d = widgets.Dropdown(options=neighborhoods.Name.unique(),
                    value='Central Hollywood')
d
```

Central Hollywood

```
[*]: # select the neighborhood from above
neighborhood = neighborhoods[neighborhoods.Name == d.value]
neighborhood_geom = neighborhood.iloc[0].geometry
streets = cat.geohub.streets.read()
```

Define a few useful projections:

- `socal_meters` : NAD83 projection for Southern California, meters
- `web_mercator` : Web mercator projection, used for map tiles
- `wgs84` : WGS84 lat/lon projection

Output View Content:

The output view displays two maps of Central Hollywood:

- Dockless mobility trips:** A map showing a dense network of red lines representing scooter trips.
- Heatmap of dockless mobility usage:** A heatmap showing the intensity of scooter usage across the neighborhood, with a color scale ranging from 250 (dark purple) to 1750 (yellow).

Mode: Command | Ln 1, Col 1 | neighborhood-scoots.ipynb

| A Civic Data Science Tech Stack

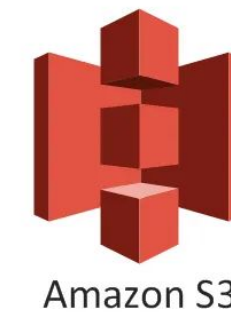
Analysis Environment



Data Analysis



Data Access



Infrastructure



| Thanks for listening!

Questions?