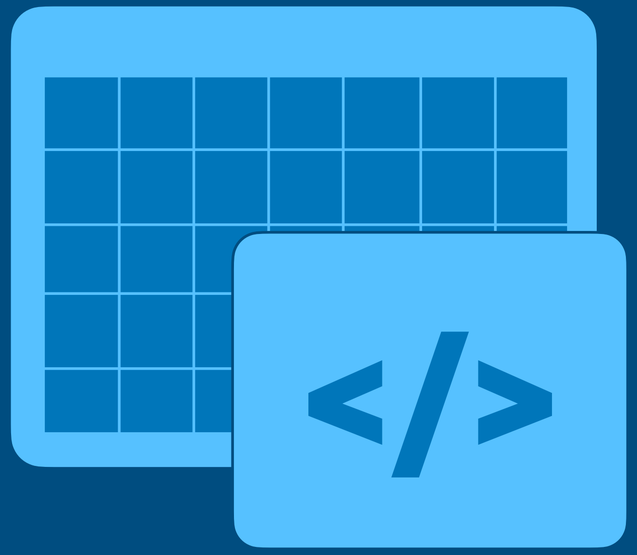


Analyzing Structured Data in Python

WEEK 1: PANDAS

18 March 2022

CDCS Python Course Series

Instructor: Lucy Havens

ASSIGNMENT

Watch the videos in sections *6. Introduction to Pandas* and *7. Baby Names with Pandas* in the course *Python: Data Analysis (2015)*
https://www.linkedin.com/learning/python-data-analysis-2015/dataframes-in-pandas?u=50251009
Follow along in your own Jupyter Notebook!

Find your own CSV file to load and turn into a DataFrame (or create your own). What questions can you ask about it using the methods and functions in Pandas?

HOW DID IT GO?

FURTHER RESOURCES

- Noteable User Guide: https://noteable.edina.ac.uk/user_guide/#hide_ge_7
- Jupyter Notebooks, Noteable: https://github.com/edina/Exemplars2020/blob/master/TeachingDocs/Tutorials/UsingNoteableBeginner.ipynb
- Jupyter Notebooks: https://glam-workbench.github.io/getting-started/
- Python: https://programminghistorian.org/en/lessons/introduction-and-installation

PANDAS: A QUICK RECAP

Pandas is built on NumPy

Pandas-specific data structures:

1. Series
2. DataFrame

Reference: https://pandas.pydata.org/pandas-docs/stable/user_guide/10min.html

PANDAS

DataFrames are like tables, storing data in rows and columns

- Text
- Numbers
- Lists, arrays, dictionaries

Each column can have a unique data type, or **dtype**

PANDAS

DataFrames can be created from data in numerous formats, including lists and arrays, dictionaries, and CSV files

```
In [142]: pd.read_csv('foo.csv')
Out[142]:
     Unnamed: 0           A           B           C           D
0    2000-01-01    0.350262    0.843315    1.798556    0.782234
1    2000-01-02   -0.586873    0.034907    1.923792   -0.562651
2    2000-01-03   -1.245477   -0.963406    2.269575   -1.612566
3    2000-01-04   -0.252830   -0.498066    3.176886   -1.275581
4    2000-01-05   -1.044057    0.118042    2.768571    0.386039
..          ...         ...         ...         ...         ...
995  2002-09-22  -48.017654   31.474551   69.146374  -47.541670
996  2002-09-23  -47.207912   32.627390   68.505254  -48.828331
997  2002-09-24  -48.907133   31.990402   67.310924  -49.391051
998  2002-09-25  -50.146062   33.716770   67.717434  -49.037577
999  2002-09-26  -49.724318   33.479952   68.108014  -48.822030

[1000 rows x 5 columns]
```

```
In [9]: df2 = pd.DataFrame({'A': 1.,
   ...:                     'B': pd.Timestamp('201
   ...:                     'C': pd.Series(1, inde
   ...:                     'D': np.array([3] * 4,
   ...:                     'E': pd.Categorical(["
   ...:                     'F': 'foo'})
   ...:

In [10]: df2
Out[10]:
     A          B    C  D      E    F
0  1.0 2013-01-02  1.0  3   test  foo
1  1.0 2013-01-02  1.0  3  train  foo
2  1.0 2013-01-02  1.0  3   test  foo
3  1.0 2013-01-02  1.0  3  train  foo
```

```
In [5]: dates = pd.date_range('20130101', periods=6)

In [6]: dates
Out[6]:
DatetimeIndex(['2013-01-01', '2013-01-02', '2013-01-03', '2013-01-04',
               '2013-01-05', '2013-01-06'],
              dtype='datetime64[ns]', freq='D')

In [7]: df = pd.DataFrame(np.random.randn(6, 4), index=dates, columns=list('ABCD'))

In [8]: df
Out[8]:
                   A         B         C         D
2013-01-01  0.469112 -0.282863 -1.509059 -1.135632
2013-01-02  1.212112 -0.173215  0.119209 -1.044236
2013-01-03 -0.861849 -2.104569 -0.494929  1.071804
2013-01-04  0.721555 -0.706771 -1.039575  0.271860
2013-01-05 -0.424972  0.567020  0.276232 -1.087401
```

PANDAS

If you have a lot of data, you can select a subset of rows to view

```
In [13]: df.head()
Out[13]:
                   A         B         C         D
2013-01-01  0.469112 -0.282863 -1.509059 -1.135632
2013-01-02  1.212112 -0.173215  0.119209 -1.044236
2013-01-03 -0.861849 -2.104569 -0.494929  1.071804
2013-01-04  0.721555 -0.706771 -1.039575  0.271860
2013-01-05 -0.424972  0.567020  0.276232 -1.087401

In [14]: df.tail(3)
Out[14]:
                   A         B         C         D
2013-01-04  0.721555 -0.706771 -1.039575  0.271860
2013-01-05 -0.424972  0.567020  0.276232 -1.087401
2013-01-06 -0.673690  0.113648 -1.478427  0.524988
```

PANDAS

There built in methods to reorganize, group, and reshape your data, and calculate summary statistics

```
In [22]: df.sort_values(by='B')
Out[22]:
                   A         B         C         D
2013-01-03 -0.861849 -2.104569 -0.494929  1.071804
2013-01-04  0.721555 -0.706771 -1.039575  0.271860
2013-01-01  0.469112 -0.282863 -1.509059 -1.135632
2013-01-02  1.212112 -0.173215  0.119209 -1.044236
2013-01-06 -0.673690  0.113648 -1.478427  0.524988
2013-01-05 -0.424972  0.567020  0.276
```

```
In [89]: df.groupby('A').sum()
Out[89]:
            C         D
A
bar  1.732707  1.073134
foo  2.824590 -0.574779
```

```
In [19]: df.describe()
Out[19]:
              A         B         C         D
count  6.000000  6.000000  6.000000  6.000000
mean   0.073711 -0.431125 -0.687758 -0.233103
std    0.843157  0.922818  0.779887  0.973118
min   -0.861849 -2.104569 -1.509059 -1.135632
25%   -0.611510 -0.600794 -1.368714 -1.076610
50%    0.022070 -0.228039 -0.767252 -0.386188
75%    0.658444  0.041933 -0.034326  0.461706
          2112  0.567020  0.276232  1.071804
```

LET'S CODE!

CONSIDERATIONS WHEN WORKING WITH DATA

Data are summaries

Data reflect power distributions in society

Who gets to have a voice?

Data about people represent only select characteristics about people, they are not complete representations of people

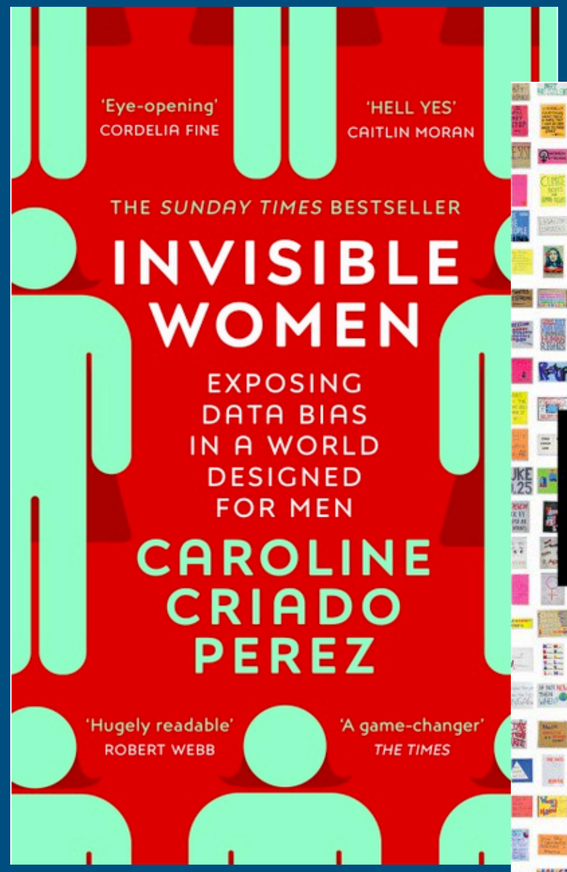
CONSIDERATIONS WHEN WORKING WITH DATA

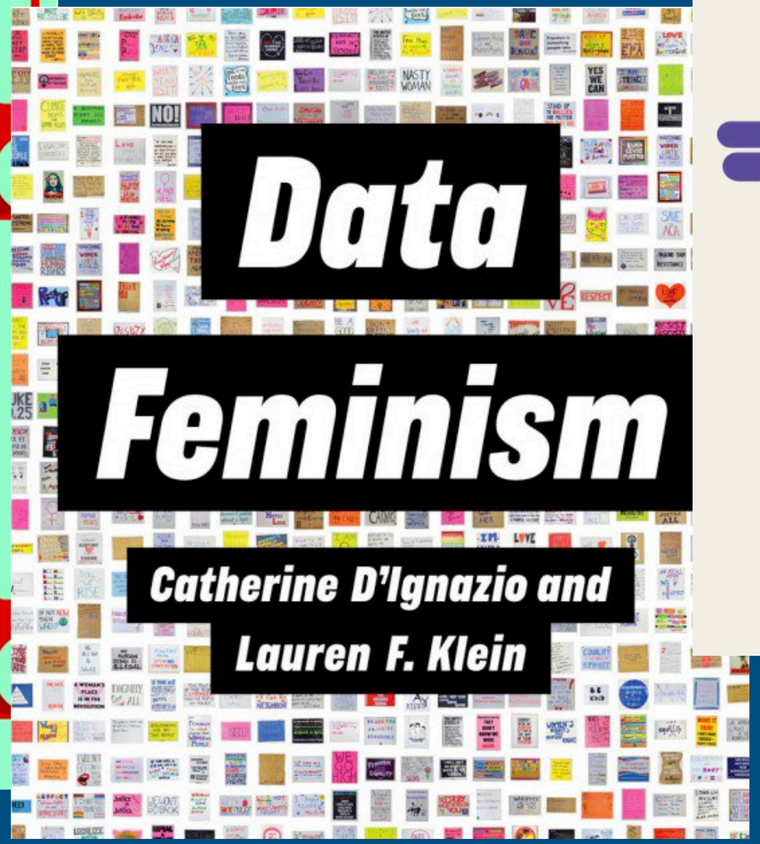
Make a **Data Biography** - *by Heather Krause*

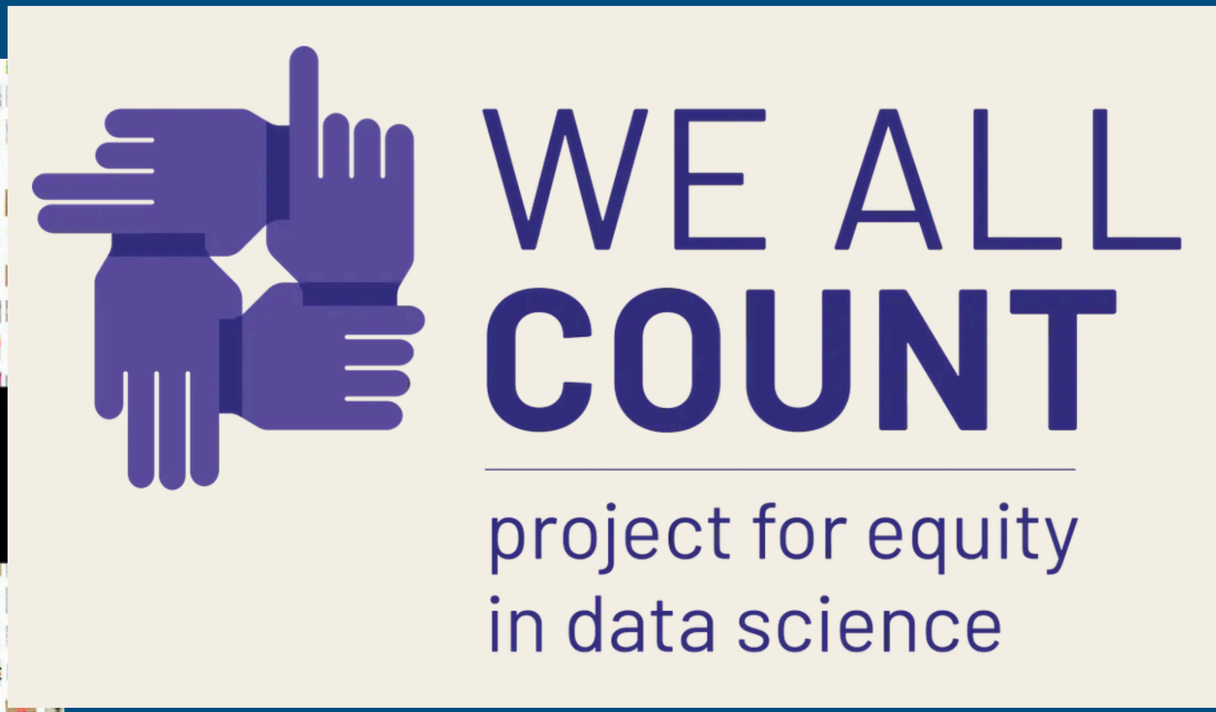
- Who is the data collector? The data owner?
- Why was the data collected?
- How was the data collected?
- What was the design and process for collection?
- Where was the data collected? Stored?
- When did the data collection happen?

Reference: https://weallcount.com/2019/01/21/an-introduction-to-the-data-biography/

MORE ON WORKING WITH DATA





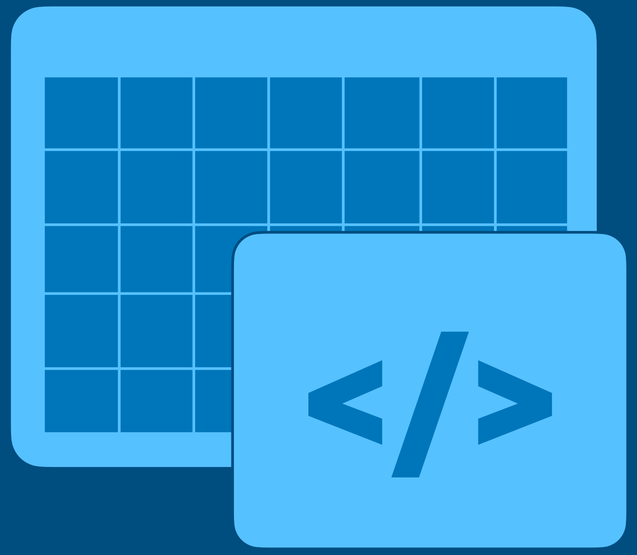


Refs: https://www.penguin.co.uk/books/111/1113605/invisible-women/9781784706289.html, https://data-feminism.mitpress.mit.edu/, https://weallcount.com/

UP NEXT

Analyzing **XML** (Extensible Markup Language) data

With the **ElementTree** library



THANKS EVERYONE!

Next course meeting: Monday, 10:00-11:00 AM BST

Office hours available on Wednesday (30 minutes)

To schedule, please message me on Teams!