COMP 150: Data in Python

Week 09, Lecture 14

Agenda

1. Data and Python
    a. Why should you care?
    b. Early lessons
    c. Best practices
2. Pandas
    a. What is it?
    b. Why should you use it?
    c. General practices
    d. Ex: read a CSV
3. Plotly
    a. What is it?
    b. Why should you use it?
    c. General practices
    d. Ex: scatter plot

Data and Python: Why do we Care?

- The good:
  - It's everywhere
  - Looks great on a resume
  - Fast
  - Extendable
  - More professional results
- The hard:
  - Requires learning
  - "I can just use Excel"

Data and Python: Why do we Care?

"I'll just use Excel"

- Yes!
- You absolutely can!
- It's fine to do this!

...But it's better if you don't.

Data and Python: Why do we Care?

Excel Example:

- Python is faster
- The results are better (and can be interactive)
- The results are *easily* shared
- Extension is easy

Data and Python: Early Lessons

Three steps:

1. Get the data in
2. Do something with it
3. Show something about it

Only step (1) is non-optional.

Data and Python: Early Lessons

Three steps:

1. Get the data in
2. Do something with it
3. Show something about it

Data and Python: Early Lessons

Three steps:

1. **Get the data in**
2. Do something with it
3. Show something about it

Data and Python: Early Lessons

Three steps:

1. **Get the data in**
2. Do something with it
3. Show something about it

Solution for (1): Pandas Data Frames

Data and Python: Early Lessons

Three steps:

1. Get the data in
2. Do something with it
3. Show something about it

Data and Python: Early Lessons

Three steps:

1. Get the data in
2. Do something with it
3. **Show something about it**

Data and Python: Early Lessons

Three steps:

1. Get the data in
2. Do something with it
3. **Show something about it**

Solution for (3): Plotly or Seaborn

Data and Python: Early Lessons

Three steps:

1. Get the data in
2. Do something with it
3. Show something about it

Data and Python: Early Lessons

Three steps:

1. Get the data in
2. **Do something with it**
3. Show something about it

Data and Python: Early Lessons

Three steps:

1. Get the data in
2. **Do something with it**
3. Show something about it

Solution for (2): Anything!

Best Practices

- Data Security
  - Colab *is not secure*. Do *not* give it access to your Google Drive.
  - Never, ever give an AI agent access to your file system

Best Practices

- Data Security
  - **Colab *is not secure***. Do *not* give it access to your Google Drive.
  - Never, ever give an AI agent access to your file system

Best Practices

- Data Security
  - **Colab *is not secure***. Do *not* give it access to your Google Drive.
  - Never, ever give an AI agent access to your file system

Best Practices

- Data Security
  - **Colab *is not secure***. Do *not* give it access to your Google Drive.
  - Never, ever give an AI agent access to your file system

Never,

never ever,

never ever ever *ever*

give an AI access to your file system.

Ever.

Best Practices

- Data Security
  - Colab *is not secure*. Do *not* give it access to your Google Drive.
  - Never, ever give an AI agent access to your file system
- Name your data
  - “df” --- no
  - “temp_data” -- better
  - “temperature = abc # This contains temperature data of the form blah units blah” -- best
- Leave yourself notes
  - # Changing from YYYYMMDD to DD-MM-YY because the boss said so
  - # Note - these are in units of Kelvin, not Farenheit
  - # Found some missing values around the month of april, just a note

Pandas

- Library to manipulate data
- Stores things in big tables
- The good:
  - Is dead-simple to use
  - Can be arbitrarily complex if needed
  - Absolutely everywhere
- The bad:
  - Since it *can* be complex, some examples are intimidating
  - Python is **not** the most performant data language in existence
  - BUT
  - It *is* currently the most-common and most-accepted, and it's for a reason

Pandas DataFrame

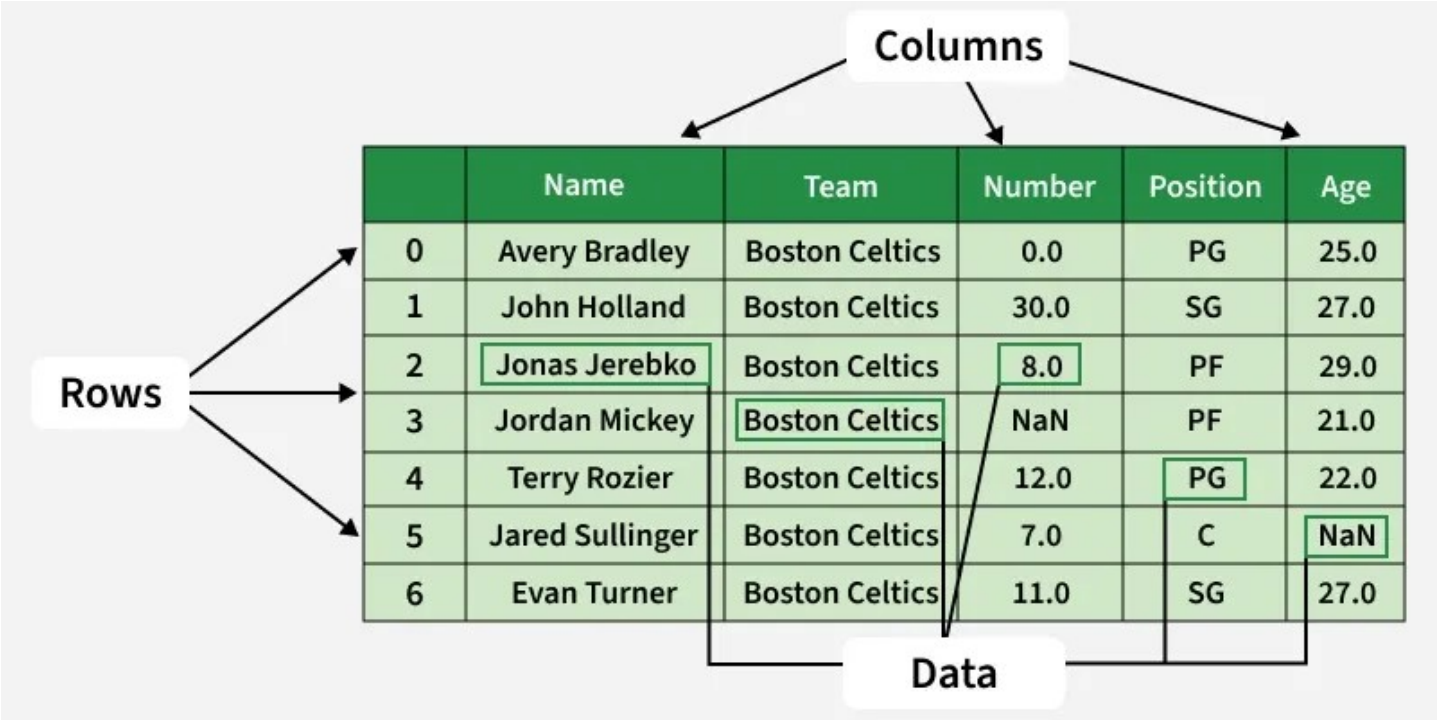

Image from GeeksForGeeks

Pandas Tutorial

- Content taken from Pandas's *10 Minutes to Pandas* tutorial
  - https://pandas.pydata.org/docs/user_guide/10min.html
- We will review:
  - Series vs DataFrame
  - Ingesting a CSV
  - head(), tail()
  - Index vs Column
  - describe(), value_counts()
  - mean, max, min, std
  - resample on numeric columns

Pandas: In General

- This is how you get your data into Python
- Avoids having to use clunky formats
- Simple to use, powerful if you need it
- *Everywhere*

Plotly: Visualization

- Massive graphing (plotting) library
- The good:
  - Polished
  - Easy to configure
  - Works directly with Pandas (plotly.express)
  - Widely-used, examples exist for anything
- The hard:
  - It can get ridiculously complex
  - There is an effort ceiling (diminishing returns)
  - Almost *too* many examples
  - Different modes of use complicates entry

Seaborn: Visualization

- High-power plotting library
- The good:
  - Intensely focused examples
  - Beautiful, consistent results
- The hard:
  - A *lot* more complicated
  - Difficult to configure to exact specifications

Visualizing Best-Practices

- For anything quick and descriptive:
  - Plotly-express
- For anything professional and similar to examples:
  - Seaborn
- For extremely complex, custom plotting:
  - Plotly graph objects (go) -- does *not* work directly with DataFrames
  - Matplotlib -- old, but extremely customizable
  - Give up and go to JavaScript -- D3 is just better