Data Manipulation and Visualization in Python

Slides by:

Joseph E. Gonzalez

jegonzal@cs.berkeley.edu

Last week Professor Hellerstein said:

"Computers like rectangular data"

& People

16.99	1.01	2.0
10.34	1.66	3.0
20.01	3.50	3.0
23.68	3.31	2.0

Matrix

sex	fare	alive
male	7.25	no
female	71.2833	yes
female	7.925	yes
female	53.1	yes

Table

Matrix Data

16.99	1.01	2.0
10.34	1.66	3.0
20.01	3.50	3.0
23.68	3.31	2.0

- Single type
  - *typically* numbers
    - Others?
  - *not always same* **meaning …**
- Indexed/Slice by **row** and **column** location
  - may take transpose or permute rows and columns
  - Need to be careful!
- Algebraic operations (+, *) are key to analytics routines
- Software/hardware heavily optimized for matrices
  - BLAS: Basic Linear Algebra Subprograms
  - SIMD SSE & AVX hardware vector processors

`import numpy as np`

The most widely used library for
multi-dimensional and linear algebra in python.

… review notebook …

Matrices are Awesome

Why would I want anything else?

Limitations of Matrices (Ndarrays)

- Real data often has **multiple types**: Text, Numbers, Dates …
  - Could store in separate arrays …
- Aligning separate arrays is **error prone** and **cumbersome**
  - ```
    zip(name[age > 3 & state[state_index] == CA],
        email[age > 3 & state[state_index] == CA])
    ```
- Location based row indexing is **error prone**
  - `zip(sort(pay), name[argsort(pay)])`
- Many computations don't naturally map to Linear Algebra
  - e.g., grouping
    ```

Tabular Data

- Most widely used data format
- **Named columns** of **different** types
  - Each column has a single type
  - Columns are indexed **by name**
- **Unordered rows** correspond to records
  - Indexed by keys, (e.g., last-name + first-name)
  - **Filtered** by predicates (e.g., fare > 70)
- Relationships may span multiple tables
  - Joins connect data across tables

sex	fare	alive
male	71.25	no
female	71.2833	yes
female	7.925	yes
female	53.1	yes

How do we compute with/on tables?

- SQL Language (Relational Algebra) *[Next Week]*
  - Most widely used language for manipulating data
  - **Declarative** specification of what we want
    - *"Make a table with these columns containing records which satisfy these properties constructed from these other tables"*
- DataFrames APIs *[Today]*
  - Hybrid between tables and matrices
  - Integrates well with **imperative languages** (e.g., Python)
    - Do this and then do that
  - Often layered over **matrix** and **relational frameworks**
- Data Scientists use both!

The Data Frame Table Abstraction

- Introduced along with the S (S-Plus & R) statistical programming languages
  - John Chambers while at Bell labs in the early 90s
- Provides an efficient & flexible table abstraction for
  - Data manipulation
  - Statistical analysis
- Widely adopted in many other analytics tools
  - Python, Julia, Spark
  - increasingly backed by relational (SQL) data systems

Pandas: Python Data Frames

- Developed by Wes McKinney while at AQR Capital Management in 2008
  - Initially designed for fast time-series and data analysis
  - Does a lot more → steep learning curve (too many features!)
- Features
  - Relies on NumPy and native optimizations → **relatively efficient**
  - Row and column indexes → **reduces transformation errors**
  - Specialized functions for handling: *missing values*, *dates*, *strings*, and *plotting*
- **Integrates with common python data science tools**
  - Scikit-Learn (Machine Learning), Matplotlib & Seaborn (Plotting), …

`import pandas as pd`

Python DataFrames

… review notebook …

Visualization in Python

- **Matplotlib**: visualization library based on *MATLAB*
  - most widely used python visualization library
  - Bad defaults, cumbersome/dated API
- **Seaborn**: runs on-top of *matplotlib*
  - Improves defaults and appearance
  - Provides additional functionality for common visualizations
- **Bokeh**: grammar of graphics based web visualizations
  - Not designed for print and limited statistical support
- Others: Plot.ly, GGPlot

`import seaborn as sns`

Improved visualization

… review notebook …

Summary

- Explored matrices and tables in python
- Advantages of computing on rectangular data
- Limitations of working with arrays
- Tables as a way to get around many of the limitations
  - Covered a lot of useful syntax (read the python notebook)
- More advanced tools for visualization
  - Simplify stratified analysis
  - Combine statistical inference and visualization