# Introduction to Pandas

Instructor: Yiyang (Ian) Wang

# Pandas

- ‘Pandas’ is derived from the term "***panel data***", an econometrics term for data sets that include observations over multiple time periods for the same individuals.
- Open Source Library
    - https://pandas.pydata.org/
- Pandas builds on Numpy arrays (we will discuss Numpy next week)

# What’s Pandas for

- Pandas will extract the data from that CSV into a DataFrame — a table, basically — then let you do things like:
    - Calculate statistics and answer questions about the data
    - Clean the data by doing things like removing missing values and filtering rows or columns by some criteria
    - Visualize the data with help from Matplotlib. Plot bars, lines, histograms, bubbles, and more
    - Store the cleaned, transformed data back into a CSV, other file or database

# How does Pandas fit into the data science toolkit

- Not only is the pandas library a central component of the data science toolkit but it is used in conjunction with other libraries in that collection.
- Pandas is built on top of the ***NumPy*** package, meaning a lot of the structure of NumPy is used or replicated in Pandas. Data in pandas is often used to feed statistical analysis in ***SciPy***, plotting functions from ***Matplotlib***, and machine learning algorithms in ***Scikit-learn***.

# Pandas vs Numpy

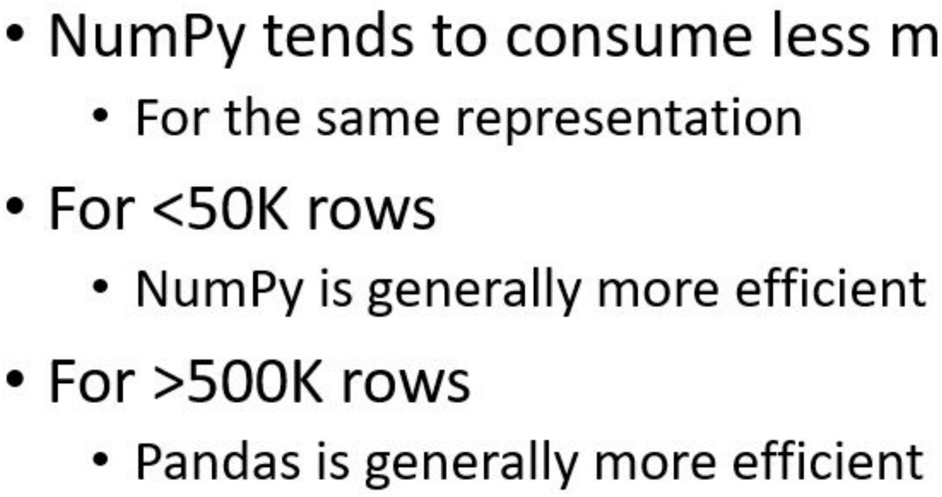
**Numpy**

- Any dimension
- Indexing by position (e.g., row or column)
- Usually a single type (e.g., int, float)

**Pandas**

- Limited to 1 (Series) or 2 (DataFrame) dimensions
- Indexing primarily by column names
- Each column has a its own type

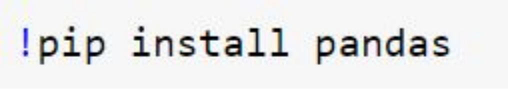
# More Pandas Vs. NumPy



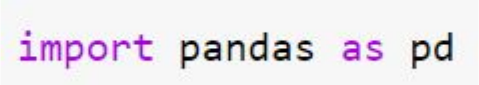
- NumPy tends to consume less memory than Pandas
  - For the same representation
- For <50K rows
  - NumPy is generally more efficient
- For >500K rows
  - Pandas is generally more efficient

**It really depends on the specific operations you're performing**

# Install and Import



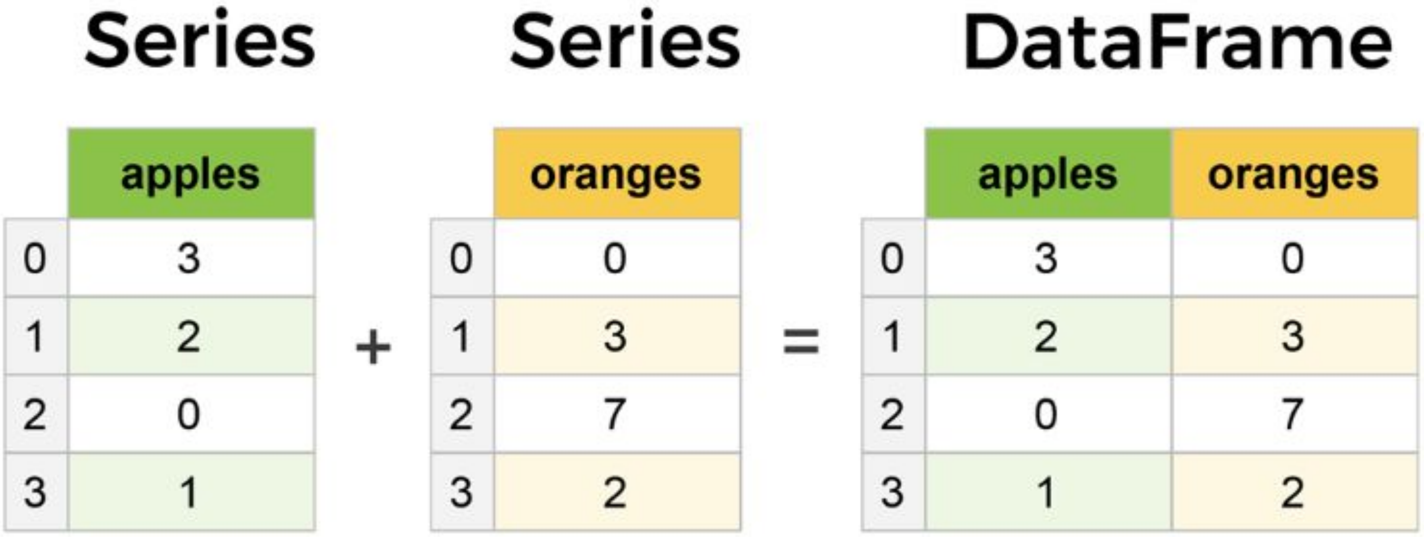
```
!pip install pandas
```



```
import pandas as pd
```

# Core components of pandas: Series and DataFrames

- The primary two components of pandas are the **Series** and **DataFrame**.
- A **Series** is essentially a column, and a **DataFrame** is a multi-dimensional table made up of a collection of Series.



Series

| | apples |
|---|---|
| 0 | 3 |
| 1 | 2 |
| 2 | 0 |
| 3 | 1 |

+

Series

| | oranges |
|---|---|
| 0 | 0 |
| 1 | 3 |
| 2 | 7 |
| 3 | 2 |

=

DataFrame

| | apples | oranges |
|---|---|---|
| 0 | 3 | 0 |
| 1 | 2 | 3 |
| 2 | 0 | 7 |
| 3 | 1 | 2 |

# A Series in Pandas

- A one-dimensional array-like object that can hold data of any type (integers, strings, floating-point numbers, etc.).
- Each element in a Series is associated with an index, which is a label that can be used to access the data.

# Data Objects

- Data sets are made up of data points
- A data object represents and entity
- Examples:
    - Sales database: object → customers, store items
    - Medical database: object → patients, treatments
    - University database: object → students, professors, courses
- Also called ***samples***, ***examples***, ***instances***, ***data points***, ***objects***, ***tuples***, ***vectors***
- Data objects are described by **attributes**
- In general, **rows** → **data objects**; **columns** → **attributes**

# Attributes

- Attributes (dimensions / features / variables): a data field representing a characteristic or property of a data object
  - E.g., customer_ID, name, address, income, GPA,...
- Types:
  - Nominal (Categorical)
  - Ordinal
  - Numeric: quantitative
    - Interval-scaled
    - Ratio-scaled

# In-Class Demo