lesson 04

Pandas Underneath the Hood: Working with NumPy

Python for Financial Analysis

Rajah Chacko

elvtr

Syllabus Review

1. Introduction to Python: Python in Finance
2. Python Basic Syntax: Importing Libraries
3. Working with Pandas
4. **Pandas Underneath the Hood: Working with NumPy**
5. Data Wrangling and Visualization
6. Extracting Financial Insights from Charts and Graphs
7. Financial Calculations with Python: Part 1
8. Financial Calculations with Python: Part 2
9. CAPM and Portfolio Management
10. Linear Regression
11. Time Series Analysis
12. Algorithmic Trading

Bonus Class: Cryptocurrency Beyond the Basics with a Fintech Guest Speaker

Class agenda

- Pythonic: applying functions with the '.' operator
- Pandas: Selecting data with loc and iloc
- Pandas: dropping rows, adding columns
- NumPy arrays and Pandas Series
- Array functions
- Speedy NumPy
- Tradeoffs between arrays and lists
- Pythonic: how to add to empty lists and dictionaries.

Pythonic: applying functions with the '.' operator

- The . gets us to an attribute within a class
- The . gets us to a column within a dataframe

Pandas: Selecting data with loc and iloc

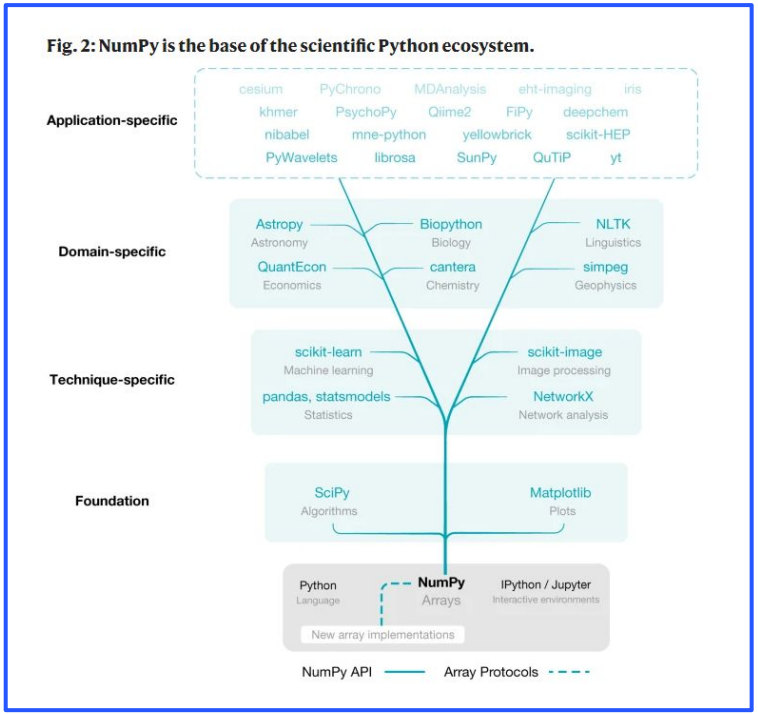
- Loc filters the dataframe by criteria
  A. Like a SELECT in a database
  B. Can also use loc for certain columns
- iloc filters the dataframe by index
  A. Pulls out individual rows
  B. (Not pythonic to step through a dataframe)

Pandas: dropping rows, adding columns

- Dropping
  A. We can drop missing values with dropna()
  B. We can drop columns or rows with drop()
- Adding columns
  A. Setting all to a constant
  B. Setting by function

NumPy arrays and Pandas Series

- Basics of NumPy
  - A. Grumpy NumPy?
  - B. Waiting in the wings
  - C. Think differently to make it NumPy friendly
- Array functions
  - A. np.array()
  - B. np.shape
  - C. np.arange
  - D. np.linspace



Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020). https://doi.org/10.1038/s41586-020-2649-2

Speedy NumPy

- The timeit function
  A. NumPy doesn't always win... when will it lose?
  B. Must think in terms of vectors, arrays, and matrices
- NumPy arrays vs. Python lists

	NumPy arrays	Python lists
Mixed types?	No	Yes
Speed	Faster (if really big)	Easier
How you code it	Think like a vector machine	Think like a programmer
Types to love	Ints, floats, booleans	Strings and objects, too

Adding columns with NumPy

- np.where for true / false
- np.select for more complex conditions

Starting from scratch

- Empty list: [ ]
- Empty dictionary: { }
- Empty set: set()
- Empty DataFrame

Assignment #4

Create a class that gets initialized with a ticker. Create a method that takes reads from the API based on the ticker (instead of creating the spreadsheets manually). Create methods that print the same stats. Instantiate objects with three different tickers and display the stats for those ticker objects. Calculate the stats using NumPy Arrays.

Take-home (optional): Use the timeit function to compare speeds in Pandas and NumPy for a month of daily data (about 22 points), a year of daily data (about 220 points), and a decade of daily data (about 2200 points).

Resources
(part 1)

- loc and iloc:

  https://www.geeksforgeeks.org/difference-between-loc-and-iloc-in-pandas-dataframe/

- Missing values:

  https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.isna.html

- Dropping missing values:

  https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.htm

- Dropping rows:

  https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop.html

Resources
(part 2)

- NumPy:

  White paper:
  https://www.nature.com/articles/s41586-020-2649-2

  Reference:
  https://numpy.org/doc/stable/reference/

  Quick start:
  https://numpy.org/doc/stable/user/quickstart.htm

- Timeit:

  Reference:
  https://docs.python.org/3/library/timeit.html

  Good example:
  https://www.geeksforgeeks.org/timeit-python-examples/

Q&A