Working with Data in Python

Chicago Federal Reserve Bank Workshop 2016

May 24, 2016

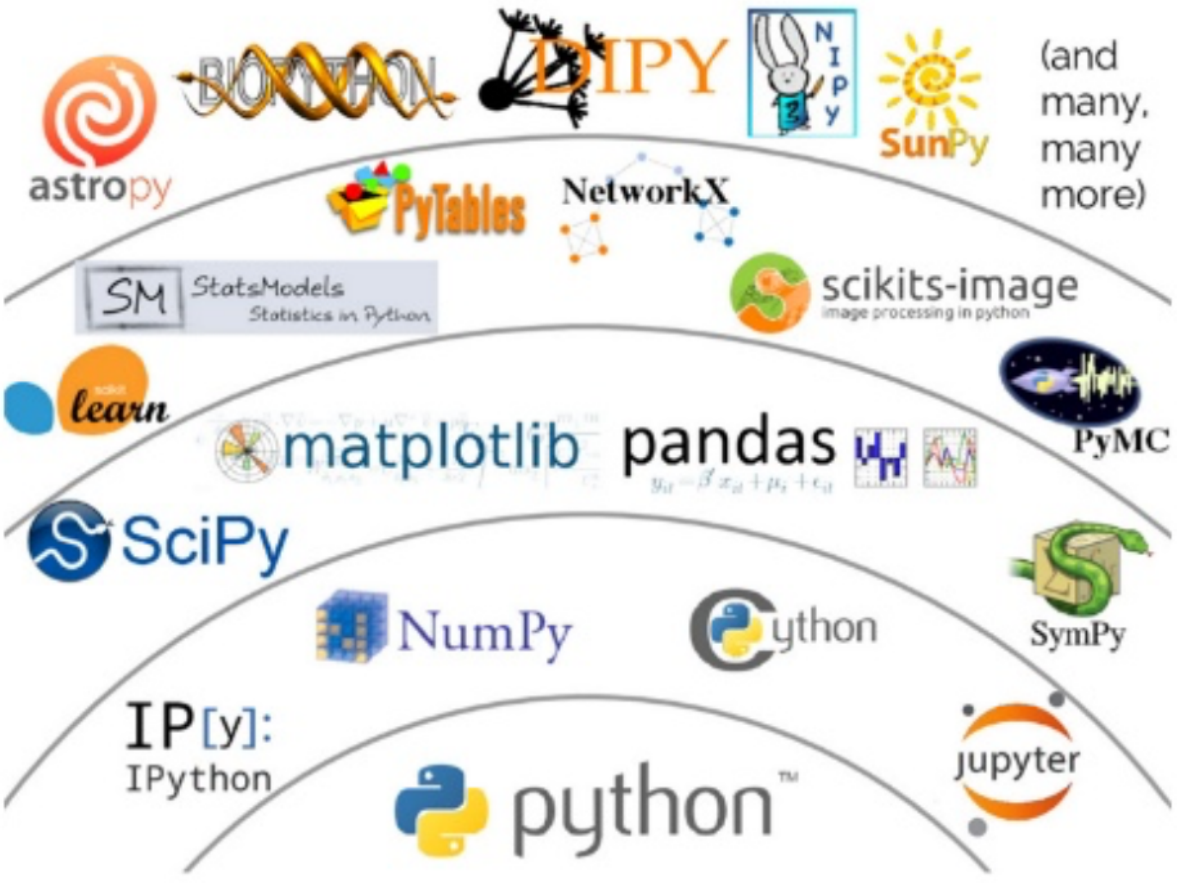
Agenda

1. Where does Pandas fit?
2. Demo
3. Introduction to Pandas
    - pd.Series
    - pd.DataFrame
4. Time Series Data
5. Exercises (pd.Series and pd.DataFrame)

Break

1. Chicago Federal Reserve Bank Data (Excel)
2. Working with **medium** sized data
3. Web Data
4. Exercises (Working with Data)

Where does Pandas Fit?



https://speakerdeck.com/jakevdp/the-state-of-the-stack-scipy-2015-keynote?slide=8

Some great packages for working with data

1. **pandas**
2. dask
    - flexible parallel computing library for analytics
    - dask.DataFrame
3. odo - Data Conversions
4. statsmodels - Regression and Statistics
5. scikit-learn - Machine Learning
6. NetworkX

Additional packages for working with data

New and interesting

1. xarray - N-dimensional Pandas

Plotting

1. matplotlib
2. Plotly
3. Bokeh
4. Myavi, Chaco, ... many others

Rpy2, BeautifulSoup, Requests, ...

+++ many more

Quick Pandas Demo

1. Random Time Series
2. Chicago Federal Reserve - CFNAI Data
3. FRED Data

See: **intro-python-data-analysis.ipynb**

Pandas

Pandas is the key library for data work in Python and it is built on top of **NumPy**

Some things that Pandas is very good at:

1. Easy handling of missing data (represented as NaN)
2. Automatic and explicit data alignment
3. Hierarchical labeling of axes

Reference: `http://pandas.pydata.org/` [Docs are 2,017 pages long]

Pandas

Pandas is focused on two primary abstractions:

1. pd.Series() - Array Like Data
2. pd.DataFrame() - Tabular Data

Pandas - Continued

Operations:

1. Powerful, flexible group by functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data
2. Intelligent label-based slicing, fancy indexing, and sub-setting of large data sets
3. Intuitive merging and joining of data sets
4. Flexible reshaping and pivoting of data sets

Reference:

`http://pandas.pydata.org/pandas-docs/version/0.18.1/index.html`

Pandas - Continued

IO:

1. Robust IO tools for loading data from
   - flat files (CSV and delimited),
   - Excel files,
   - databases,
   - and saving / loading data from the fast HDF5 format

Reference:

`http://pandas.pydata.org/pandas-docs/version/0.18.1/io.html`

Pandas - Continued

Specialized Data Types: TimeSeries

1. Time series specific functionality:
   - date range generation and frequency conversion,
   - moving window statistics,
   - moving window linear regressions,
   - date shifting and lagging, etc.
   - time zone handling

Reference:

`http://pandas.pydata.org/pandas-docs/version/0.18.1/timeseries.html`

pd.Series Object

A **Pandas** Series is a one-dimensional labeled array capable of holding any data type (integers, strings, floating point numbers, Python objects, etc.).

```
import pandas as pd
s = pd.Series([5,4,3,2,1], index=['a', 'b', 'c', 'd', 'e'])
```

Produces the following object:

```
a    5
b    4
c    3
d    2
e    1
dtype: int64
```

pd.Series Object

```
s.sort_values()
```

```
e    1
d    2
c    3
b    4
a    5
dtype: int64
```

pd.Series Object

```
s[s > 2]
```

```
a    5
b    4
c    3
dtype: int64
```

pd.DataFrame Object

```
d = {'one' : pd.Series([1., 2., 3.],
                       index=['a', 'b', 'c']),
     'two' : pd.Series([1., 2., 3., 4.],
                       index=['a', 'b', 'c', 'd'])}
df = pd.DataFrame(d)
```

Produces the DataFrame:

	one	two
a	1.0	1.0
b	2.0	2.0
c	3.0	3.0
d	NaN	4.0

Exercises - pd.Series and pd.DataFrame

Refer to notebook: **exercises-pandas-series-dataframes.ipynb**

Applications

1. Working with Time Series Data
    - Closer look at the Chicago Fed Data
    - Financial Data
2. Working with **medium** sized data
    - International Export Data
3. Web Data

Exercises - Applications

Refer to notebook: **exercises-pandas-applications.ipynb**