Programming for Analytics

Introduction to data handling with Pandas

S. Kanungo

Outline

- Understanding Pandas
- Series and Time Series
- Data frames

pandas

- **Pan**el **Da**ta **S**ystem
- Key components – series, dataframes
- Built on top of numpy

pandas consists of

- A set of labeled array data structures; main ones include: Series / TimeSeries and DataFrame
- Index objects enabling both simple axis indexing and multi-level / hierarchical axis indexing
- Integrated “group by” engine to aggregate and transform data sets
- Date range generation (date range) and custom date offsets
- i/o tools
- Memory-efficient “sparse” versions of the standard data structures, Moving window statistics
- Static and moving window linear and panel regression

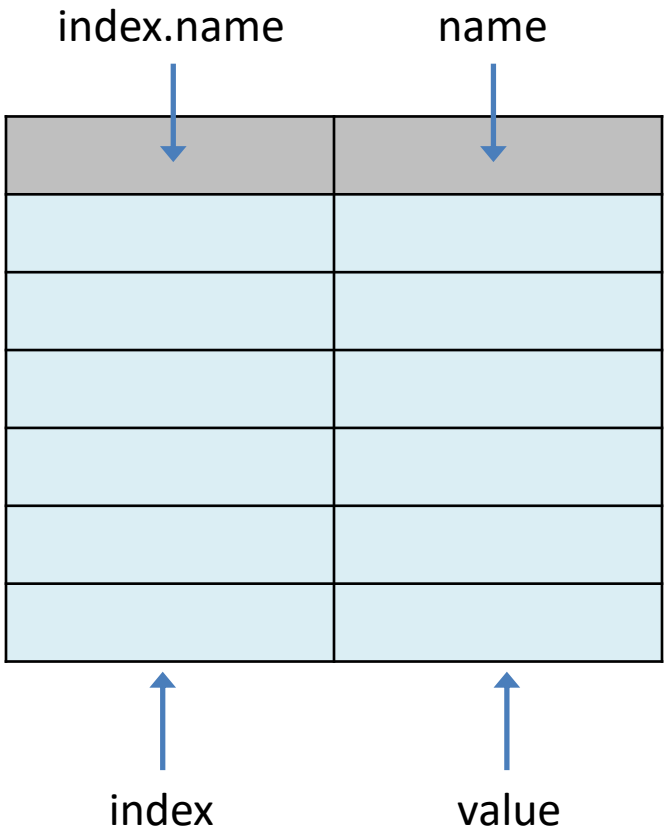
pandas data structures

Dimensions	Name	Description
1	Series	1D labeled homogeneously-typed array
1	TimeSeries	Series with index containing datetimes
2	DataFrame	General 2D labeled, size-mutable tabular structure with potentially heterogeneously-typed columns
3	Panel	General 3D labeled, also size-mutable array

series and time series

- Series is an indexed (labeled) vector
- TimeSeries, in pandas, is a Series where the index is composed of dates
- A Series can only have one dimension (column)
- The data within these data structures are stored in numpy-ndarray objects. So each column can contain any kind of data.

pandas series



- Create series
- Create series with index
- Accessing series values
- Operations on series
- Series as ordered dictionaries
- Create series from dictionary

pandas time series

- Times series are useful in many areas – especially business and finance
- Pandas handles fixed frequency as well as irregular time time-related data
- Types of data marked by time
  - Time stamps
  - Fixed periods
  - Time intervals
  - Experiment or elapsed time

`pandas` data and time data types

- Python standard library contains
  - Data types date and time
  - Functions for date and time
  - `date`, `time`, `datetime` and `timedelta`
- Converting to and from strings is important when using Timestamp objects in pandas
- Need to know date and time formats

Basic time series

- A series that is indexed by time stamps
- Typically, timestamps are external to pandas
- In class
  - Make a list of 5 dates using `datetime` function and save in `mydates`
  - Create a time series called `myTS` using 6 random numbers (`np.random.randn()`) and indexed by `mydates` using the `Series` function of `pandas`
- I do not get the type to be `pandas.core.series.Timeseries`

Indexing, selection, subsetting

- In class exercise

dataframe

- Spreadsheet-like data structure containing an ordered collection of columns
- Contains both row and column indexes
- Think of it as a dict of Series (with shared index)

Creation of dataframe

```
mydata = {
            'state': [],
            'year':[],
            'pop':[]
        }
```

```
mydata = {
            'state': ['FL', 'FL', 'GA', 'GA', 'GA'],
            'year':[2010, 2011, 2008, 2010, 2011],
            'pop':[18.8, 19.1, 9.7, 9.8, 9.9]
        }
```

Creating with dicts of dicts

```
mydata = {
            'FL': {},
            'GA': {}
        }
```

```
mydata = {
            'FL': {2010: 18.8, 2011: 19.1},
            'GA': {2008: 9.7, 2010: 9.8, 2011: 9.9}
        }
```

Accessing data in DataFrames

- Columns can be retrieved as Series using
  - `dict notation -> mydf['state']`
  - `attribute notation -> mydf.state`
- Rows can retrieved by
  - position or by name using using ix attribute
  - `mydf[:2]`

Add new column

- By computation
  - `mydf['anothercol'] = NaN`
- By direct assignment
  - `mydf['calccol'] = mydf['pop']*1.5`

Basic functionality

- Summary and descriptives
  - `mydf.sum()`
  - `mydf.mean()`
    - `Mydf["pop"].mean()`
  - `mydf.describe()`
- Boolean indexing
  - `mydf < 9.9`
  - `mydf.FL < 18.9`
- Subsetting
  - `By column`
  - `By row`

Data importing and handling

- pandas supports several ways to handle data loading
- Text file data
  - `read_csv`
  - `read_table`
- Structured data (JSON, XML, HTML)
  - existing libraries work
- Excel (depends upon `xlrd` and `openpyxl` packages)
- Database
  - pandas.io.sql module (read_frame)

Pandas and plotting

- This is from the pandas site
  - Get a file
  - Inspect it
  - Create crosstabs
  - Sum by an attribute
  - Compute percentages
  - Plot the percentages
  - Draw histogram of some proportion

Other topics

- Covered in the tutorial/assignment
  - GroupBy
  - Pivot Tables
  - SQL-like operations
  - Importing different data formats
    - Merge, join, and concatenate
- More on pandas indexes
  - http://pandas.pydata.org/pandas-docs/stable/indexing.html