Financial Data Analysis with Python

Lecture 10. Review

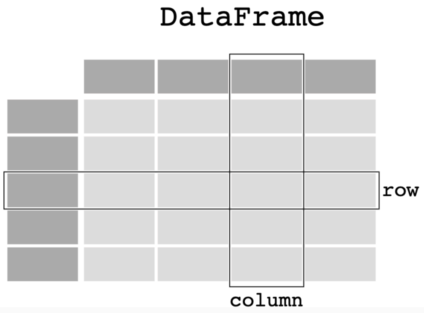
Luping Yu (俞路平)

Xiamen University

June 6, 2023

Summary

- An introductory course in working with data in Python
  - Much of this course focuses on table-based (structured) data
  - pandas is a major tool throughout much of the course
  - pandas contains data structures and data manipulation tools designed to make data cleaning and analysis fast and easy in Python



What is Pandas for?

- 4 typical steps: load, clean, wrangling, and analyze
  - Data loading and storage (L3)
    - Reading and writing data in multiple formats (**.csv** .xls .txt .json)
    - Indexing & reindexing
  - Data cleaning and preparation (L3)
    - Handling missing data
    - Data transformation
  - Data wrangling: join, combine, and reshape
    - Aggregation and group operations (L4)
    - Combining and merging datasets (L5)
  - Data analysis
    - Plotting and visualization (L6)
    - Time series data analysis (L7)

Lecture 02. Data Structure

- Python built-in types:
  - Scalar types: numeric types (int, float), string, boolean
  - Data structures: list, set, dict
- Pandas data structures:
  - Series: pd.Series()
    - Series is a one-dimensional array-like object containing a sequence of **values** and an associated array of data labels (a.k.a. **index**)
  - DataFrame: pd.DataFrame()
    - DataFrame is **two-dimensional**
    - DataFrame represents a rectangular table of data and contains an ordered collection of columns
  - Essential functionality
    - Selection and filtering: loc[], iloc[]
    - Sorting and ranking: sort_index(), sort_values()
    - Arithmetic and data alignment

Lecture 03. Data Loading and Cleaning

- Data preparation: loading, cleaning, transforming, and rearranging
  - Reading and writing **tabular data** as a DataFrame object
    - read_csv(), to_csv()
    - Parameters of data loading functions (header, names, index_col, etc.)
  - Data cleaning and preparation
    - Missing data: dropna(), fillna()
    - Duplicates: drop_duplicates()
    - Replacing values: replace()
    - Vectorized string functions: str.contains(), str.split()

Lecture 04. Data Aggregation and Group Operations

- Split-apply-combine
  - A Series/DataFrame is **splitted** into groups based on one or more keys
  - A function is **applied** to each group, producing a new value
  - The results of all those applications are **combined** into a result object
- GroupBy mechanics
  - groupby(): slice, dice, and summarize datasets
    - Built-in functions: mean(), size(), sum(), count()
    - Data aggregation: agg(), apply
    - Data transformation: transfrom()

Lecture 05. Data Wrangling: Combine and Merge

- Combining and merging datasets
  - pandas.concat() concatenates or "stacks" together objects along an axis
    - Concatenating along the row: axis=0
    - Concatenating along the column: axis=1
  - pandas.merge(): connects rows in DataFrames based on one or more keys
    - inner join, outer join
    - left join, right join
    - many-to-one join, many-to-many join
    - merge on column, merge on index

Lecture 06. Plotting and Visualization

- Basic data visualization using pandas, matplotlib, and seaborn
  - Plotting with pandas
    - Line plot: plot()
    - Bar plot: plot.bar(), plot.barh()
    - Histograms: plot.hist()
    - Density plot: plot.density()
  - Plotting with matplotlib
    - Create one or more subplots: plt.subplots()
  - Plotting with seaborn
    - Grouping dimension: sns.barplot(hue)
    - Additional grouping dimension: sns.catplot(hue, col, kind)
    - Histogram and density estimate: sns.histplot(kde)
    - Scatter plot and linear regression: sns.regplot()

Lecture 07. Time Series

- Time series data: data that is observed at many points in time forms
  - Data types of date and time
    - datetime.datetime(): stores both the date and time
    - datetime.timedelta(): difference between two datetime objects
  - Converting between string and datetime
    - datetime.strptime(), dateutil.parser(), pd.to_datetime()
  - Time series basics
    - Time series object as index: **DatetimeIndex**
    - Fixed-frequency date ranges: pd.date_range()
    - Moving data backward and forward through time: pd.shift()
  - Resampling and frequency conversion
    - Downsampling: resample()
    - Upsampling resample().asfreq(), resample().ffill()
    - Moving window: rolling()

Lecture 08 (S1). Web Page and Crawler

- Web page
  - **HTML**, CSS, Javascript
    - HTML element: defined by a **start tag**, some **content**, and an **end tag**
    - HTML attributes: id, class, style
- Crawler
  - Common tools: requests, BeautifulSoup, Selenium, pd.read_html()
  - Four-step rule:
    - **Request** the content of a specific URL from the server
    - **Download** the content (source code)
    - **Identify** the elements of the page
    - **Extract** and (if necessary) reformat those elements into a dataset

Lecture 09 (S2). Text Processing

- Regular expression (RE)
  - RE characters
    - Metacharacters
    - Quantifiers
    - Groups and ranges
    - Escape characters
  - re module functions: pattern matching, substitution, and splitting
- Fuzzy Match
  - **Edit distance** (aka. **Levenshtein** distance)
    - Simple Ratio: fuzz.ratio(str1, str2)
    - Partial Ratio: fuzz.partial_ratio(str1, str2)
    - Token Sort Ratio: fuzz.token_sort_ratio(str1, str2)
    - Token Set Ratio: fuzz.token_set_ratio(str1, str2)

Good Luck with the Final Exam