Pandas: DataFrame

Akitaka Matsuo
Essex IADS

DATAFRAME BASICS

DataFrame

- DataFrame in Pandas is a collection of Series with shared index
- DataFrame looks like a spreadsheet, and spread sheet like operation works (e.g. filtering by rows, selecting columns)
- Construction:
  - `pd.DataFrame()`
    - Use: `data, index, name`
    - From a series object
  - `pd.read_csv()`

DataFrame Index

- Indexes can be set by the following methods
  - Use `index` argument when constructing a DataFrame
  - Use `set_index()` method for DataFrame
- Index can be moved to a column using `reset_index()` method
- Rearrange the indexes using `reindex()` method

University of Essex

DATAFRAME FILTERING/SELECTING/ CREATING VARIABLES

DataFrame: Filtering (select rows)

- There are several ways to select rows
  1. Using the row numbers with `:`
  2. Boolean selection
     - Example: selecting rows based on the value of some columns in the dataframe (c.f. Boolean selection of NumPy arrays)
- Both return a *copied* DataFrame

DataFrame: select columns

- If you supply a string or list of strings in the box bracket, it indicates column selection
  - This will create a new, copied DataFrame

df.drop()

- This will remove some rows/columns from a dataset, using index
  - The returning object is a copied DataFrame
  - Use `axis` argument to indicate dropping from rows/columns
  - You can also supply `inplace=True` option

df.loc[]/df.iloc[]

- These methods provide fancy-indexing-like functionality
  - `df.loc[]`: Fancy indexing with row/column *names*
  - `df.iloc[]`: Fancy indexing with row/column *numbers*
- As these are like fancy indexing, it will create a reference, not a copy

DataFrame: Assigning Columns

- To create a new column, you can use an assignment operation such as:
  - `df['new_var_name'] = values`
  - `values` can be
    - A scaler variable
    - A named series (usually using a existing variable in the DataFrame)
- Manipulation of existing variable is the same operation
  - `df['existing_var_name'] = values`
- With `.loc/.iloc,` we can change specific part of column
  - `df.loc[..,..] = values`