

Customarily import as: `import pandas as pd`

pandas Data Structures

Series

One-dimensional labeled array that can hold any data type

Syntax to create a Series

`s = pd.Series(data, index)`

Example:

`s = pd.Series([1,2,-1,4], index=['a','b','c','d'])`

a	1
b	2
c	-1
d	4

index

DataFrame

2-D labeled data structure that can have columns (different type per column)

Syntax to create a DataFrame

`df = pd.DataFrame(data, index, columns)`

Example:

`df = pd.DataFrame([[1,2,3], [4,4,7], [5,8,9]], index=['a','b','c'], columns=['A','B','C'])`

	A	B	C
a	1	2	3
b	4	4	7
c	5	8	9

columns

index

Viewing Data

View top, bottom rows

`df.head()`
`df.tail(2)`

Transpose data

`df.T`

Display index, columns

`df.index`
`df.columns`

Sort by an axis (axis=0 for rows, axis=1 for columns)

`df.sort_index(axis=0, ascending=True)`

NumPy representation of data

`df.to_numpy()`

Sort by values

`df.sort_values(by='A')`

Statistics

Get quick overall statistics

`df.describe()`

Count non-NA values

`df.count()`

Count unique values in column

`df['A'].value_counts()`

Sum of values

`df.sum()`

Count distinct values in an axis

`df['A'].nunique()`

Median of values

`df.median()`

*Note that for the functions to the right, in order to get values of columns use axis=0, but for rows use axis=1. See example below for `df.mean()`

Minimum

`df.min()`

Maximum

`df.max()`

Get the mean of the columns

`df.mean(axis=0)` or `df.mean()`

Get the mean of the rows

`df.mean(axis = 1)`

Mode

`df.mode()`

Standard deviation

`df.std()`

Missing Data

Reindexing can change/add/delete

`df1 = df.reindex(index=['a', 'b', 'c', 'd'], columns=['A', 'B', 'C', 'D'])`

Fill NaN's with a value

`df1.fillna(0)`

Set a value to labels in a column

`df1.loc['a':'b', 'D'] = 1`

Replace values with others

`df1.replace(1.0, 2.0)`

Drop NaN values

`df1.dropna(how='any')`

Get rows with NaN values

`df1[df1.isna().any(axis=1)]`

Get columns with NaN values

`df1.isna().any()`

Merge

Concat

Break up the DataFrame

`pieces = [df[0:2], df[2:3]]`

Concatenate pandas objects

`pd.concat(pieces)`

Join

`left = df.iloc[:, 0:2]`

`right = df.iloc[:, 2:4]`

Join two DataFrames

`pd.merge(left, right, left_index=True, right_index=True)`

Append

Append a row to a pandas

DataFrame

`s = df.iloc[2].astype(int)`
`df.append(s)`

Setting/Grouping

Setting a new column

`df['D'] = ['one', 'two', 'one']`

	A	B	C	D
a	1	2	3	one
b	4	4	7	two
c	5	8	9	one

*Note that this example groups the data based on column 'D', applies the `sum()` function to each group, and returns a DataFrame

Grouping example

`df2 = df.groupby('D').sum()`

D	A	B	C
one	6	10	12
two	4	4	7

Reshaping

Example DataFrame

`df3 = pd.DataFrame([[1,2], [4,4], [5,8], [2,2]], index=['a','b','a','b'], columns=['A','B'])`

Reshape by stacking

`df3.stack()`

	A	B
a	1	2
b	4	4
a	5	8
b	2	2



a	A	1
	B	2
b	A	4
	B	4
a	A	5
	B	8
b	A	2
	B	2

*The inverse operation of `stack()` is `unstack()`

*Other ways to reshape are to use `melt()` or `pivot()`

Indexing/Selecting

pandas generally has **three** ways to index:

[] (use square brackets):

Select a column

`df['A']`

Use **.iloc** to positionally index:

Select via a passed integer

`df.iloc[2]`

Select by label (i.e., rows)

`df[0:2]`

Slice integers

`df.iloc[2:3, 0:3]`

Use **.loc** to label index:

Get row by label

`df.loc['b'][0]`

Use lists of position locations

`df.iloc[[1, 2], [0, 2]]`

Get multi-axis by label

`df.loc[:, ['A', 'B']]`

Slice rows

`df.iloc[1:3, :]`

Slice columns

`df.iloc[:, 0:2]`

Slicing with the label

`df.loc['b':'c', ['A', 'B']]`

Get a specific value

`df.iloc[1,2]`

Get a scalar value

`df.loc['a'][0], 'A']`

*Boolean indexing can be used:

Use a column's values to select data:
`df[df['A'] > 0]`

*Use the `isin()` method to filter:

`df.loc['a':'b', ['A', 'B']].isin(['1'])`

Method chaining

Methods can be called on an object one after another (i.e., method chaining).

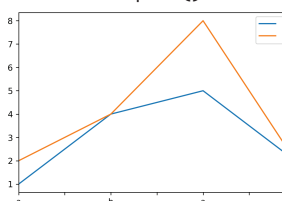
For example:

`df.groupby('D').sum().astype(float)`

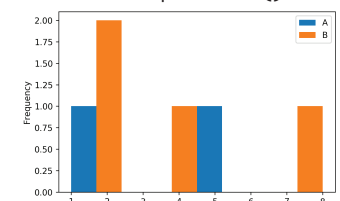
D	A	B	C
one	6.0	10.0	12.0
two	4.0	4.0	7.0

Plotting

`df3.plot()`



`df3.plot.hist()`



I/O

Syntax for reader/writer functions

`pandas.read_filetype()`
`DataFrame.to_filetype()`

Example

`pandas.read_csv()`
`DataFrame.to_csv()`