Index objects and labeled data

MANIPULATING DATAFRAMES WITH PANDAS

Anaconda

Instructor

pandas data structures

- Key building blocks
- `Index` es: Sequence of labels
    - Immutable (Like dictionary keys)
    - Homogeneous in data type (Like NumPy arrays)
- `Series` : 1D array with Index
- `DataFrame` s: 2D array with Series as columns

Creating a Series

```python
import pandas as pd
prices = [10.70, 10.86, 10.74, 10.71, 10.79]
shares = pd.Series(prices)
print(shares)
```

```
0    10.70
1    10.86
2    10.74
3    10.71
4    10.79
dtype: float64
```

Creating an index

```
days = ['Mon', 'Tue', 'Wed', 'Thur', 'Fri']
shares = pd.Series(prices, index=days)
print(shares)
```

```
Mon     10.70
Tue     10.86
Wed     10.74
Thur    10.71
Fri     10.79
dtype: float64
```

Examining an index

```
print(shares.index)
```

```
Index(['Mon', 'Tue', 'Wed',
       'Thur', 'Fri'],
       dtype='object')
```

```
print(shares.index[2])
```

```
Wed
```

```
print(shares.index[:2])
```

```
Index(['Mon', 'Tue'],
      dtype='object')
```

```
print(shares.index[-2:])
```

```
Index(['Thur', 'Fri'],
      dtype='object')
```

```
print(shares.index.name)
```

```
None
```

Modifying index name

```
shares.index.name = 'weekday'
print(shares)
```

```
weekday
Monday       10.70
Tuesday      10.86
Wednesday    10.74
Thursday     10.71
Friday       10.79
dtype: float64
```

Modifying index entries

```
shares.index[2] = 'Wednesday'
```

```
TypeError: Index does not support mutable operations
```

```
shares.index[:4] = ['Monday', 'Tuesday', 'Wednesday',
                    'Thursday']
```

```
TypeError: Index does not support mutable operations
```

Modifying all index entries

```
shares.index = ['Monday', 'Tuesday', 'Wednesday',
                'Thursday', 'Friday']
```

```
print(shares)
```

```
Monday       10.70
Tuesday      10.86
Wednesday    10.74
Thursday     10.71
Friday       10.79
dtype: float64
```

Unemployment data

```
unemployment = pd.read_csv('Unemployment.csv')
unemployment.head()
```

```
    Zip  unemployment  participants
0  1001          0.06         13801
1  1002          0.09         24551
2  1003          0.17         11477
3  1005          0.10          4086
4  1007          0.05         11362
```

Unemployment data

```
unemployment.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 33120 entries, 0 to 33119
Data columns (total 3 columns):
Zip                33120 non-null int64
unemployment       32556 non-null float64
particpants        33120 non-null int64
dtypes: float64(1), int64(2)
memory usage: 776.3 KB
```

Assigning the index

```
unemployment.index = unemployment['Zip']
unemployment.head()
```

```
       Zip  unemployment  participants
Zip
1001  1001          0.06         13801
1002  1002          0.09         24551
1003  1003          0.17         11477
1005  1005          0.10          4086
1007  1007          0.05         11362
```

Removing extra column

```
unemployment.head(3)
```

```
       Zip  unemployment  participants
Zip
1001  1001          0.06         13801
1002  1002          0.09         24551
1003  1003          0.17         11477
```

```
del unemployment['Zip']
unemployment.head(3)
```

```
       unemployment  participants
Zip
1001           0.06         13801
1002           0.09         24551
1003           0.17         11477
```

Examining index and columns

```
print(unemployment.index)
```

```
Int64Index([1001, 1002, 1003, ...],
           dtype='int64',
           name='Zip',
           length=33120)
```

```
print(type(unemployment.index))
```

```
<class
'pandas.indexes.numeric.Int64Index'>
```

```
print(unemployment.index.name)
```

```
Zip
```

```
print(unemployment.columns)
```

```
Index(['unemployment',
       'participants'],
       dtype='object')
```

read_csv() with index_col

```
unemployment = pd.read_csv('Unemployment.csv',
                            index_col='Zip')
```

```
unemployment.head()
```

```
      unemployment  participants
Zip
1001          0.06         13801
1002          0.09         24551
1003          0.17         11477
1005          0.10          4086
1007          0.05         11362
```

Let's practice!

MANIPULATING DATAFRAMES WITH PANDAS

Hierarchical Indexing

MANIPULATING DATAFRAMES WITH PANDAS

Anaconda

Instructor

Stock data

```python
import pandas as pd
stocks = pd.read_csv('datasets/stocks.csv')
print(stocks)
```

```
         Date   Close    Volume Symbol
0  2016-10-03   31.50  14070500   CSCO
1  2016-10-03  112.52  21701800   AAPL
2  2016-10-03   57.42  19189500   MSFT
3  2016-10-04  113.00  29736800   AAPL
4  2016-10-04   57.24  20085900   MSFT
5  2016-10-04   31.35  18460400   CSCO
6  2016-10-05   57.64  16726400   MSFT
7  2016-10-05   31.59  11808600   CSCO
8  2016-10-05  113.05  21453100   AAPL
```

Setting index

```
stocks = stocks.set_index(['Symbol', 'Date'])
print(stocks)
```

```
                    Close    Volume
Symbol Date
CSCO   2016-10-03   31.50  14070500
AAPL   2016-10-03  112.52  21701800
MSFT   2016-10-03   57.42  19189500
AAPL   2016-10-04  113.00  29736800
MSFT   2016-10-04   57.24  20085900
CSCO   2016-10-04   31.35  18460400
MSFT   2016-10-05   57.64  16726400
CSCO   2016-10-05   31.59  11808600
AAPL   2016-10-05  113.05  21453100
```

```
print(stocks.index)
```

```
MultiIndex(levels=[['AAPL', 'CSCO', 'MSFT'],
           ['2016-10-03', '2016-10-04', ‘2016-10-05']],
           labels=[[1, 0, 2, 0, 2, 1, 2, 1, 0],
           [0, 0, 0, 1, 1, 1, 2, 2, 2]],
           names=['Symbol', 'Date'])
```

```
print(stocks.index.name)
```

```
None
```

```
print(stocks.index.names)
```

```
['Symbol', 'Date']
```

Sorting index

```
stocks = stocks.sort_index()
print(stocks)
```

```
                    Close    Volume
Symbol Date
AAPL   2016-10-03  112.52  21701800
       2016-10-04  113.00  29736800
       2016-10-05  113.05  21453100
CSCO   2016-10-03   31.50  14070500
       2016-10-04   31.35  18460400
       2016-10-05   31.59  11808600
MSFT   2016-10-03   57.42  19189500
       2016-10-04   57.24  20085900
       2016-10-05   57.64  16726400
```

Indexing (individual row)

```
stocks.loc[('CSCO', '2016-10-04')]
```

```
Close           31.35
Volume    18460400.00
Name: (CSCO, 2016-10-04), dtype: float64
```

```
stocks.loc[('CSCO', '2016-10-04'), 'Volume']
```

```
18460400.0
```

Slicing (outermost index)

```python
stocks.loc['AAPL']
```

```
             Close    Volume
Date
2016-10-03  112.52  21701800
2016-10-04  113.00  29736800
2016-10-05  113.05  21453100
```

Slicing (outermost index)

```python
stocks.loc['CSCO':'MSFT']
```

```
                   Close    Volume
Symbol Date
CSCO   2016-10-03  31.50  14070500
       2016-10-04  31.35  18460400
       2016-10-05  31.59  11808600
MSFT   2016-10-03  57.42  19189500
       2016-10-04  57.24  20085900
       2016-10-05  57.64  16726400
```

Fancy indexing (outermost index)

```
stocks.loc[(['AAPL', 'MSFT'], '2016-10-05'), :]
```

```
                    Close    Volume
Symbol Date
AAPL   2016-10-05  113.05  21453100
MSFT   2016-10-05   57.64  16726400
```

```
stocks.loc[(['AAPL', 'MSFT'], '2016-10-05'), 'Close']
```

```
Symbol  Date
AAPL    2016-10-05    113.05
MSFT    2016-10-05     57.64
Name: Close, dtype: float64
```

Fancy indexing (innermost index)

```
stocks.loc[('CSCO', ['2016-10-05', '2016-10-03']), :]
```

```
                   Close    Volume
Symbol Date
CSCO   2016-10-03  31.50  14070500
       2016-10-05  31.59  11808600
```

Slicing (both indexes)

```
stocks.loc[(slice(None), slice('2016-10-03', '2016-10-04')),:]
```

```
                    Close    Volume
Symbol Date
AAPL   2016-10-03  112.52  21701800
       2016-10-04  113.00  29736800
CSCO   2016-10-03   31.50  14070500
       2016-10-04   31.35  18460400
MSFT   2016-10-03   57.42  19189500
       2016-10-04   57.24  20085900
```

Let's practice!

MANIPULATING DATAFRAMES WITH PANDAS