

Appending and concatenating Series

MERGING DATAFRAMES WITH PANDAS



Anaconda
Instructor

append()

- `.append()` : Series and DataFrame method
- Invocation:
 - `s1.append(s2)`
 - Stacks rows of `s2` below `s1`
 - Method for Series and DataFrames

concat()

- `concat()` : `pandas` module function
- Invocation:
 - `pd.concat([s1, s2, s3])`
 - Can stack row-wise or column-wise

concat() and .append()

- Equivalence of `concat()` and `.append()` :
- `result1 = pd.concat([s1, s2, s3])`
- `result2 = s1.append(s2).append(s3)`
- `result1 == result2` elementwise

Series of US states

```
import pandas as pd

northeast = pd.Series(['CT', 'ME', 'MA', 'NH', 'RI', 'VT',
                       'NJ', 'NY', 'PA'])

south = pd.Series(['DE', 'FL', 'GA', 'MD', 'NC', 'SC', 'VA',
                   'DC', 'WV', 'AL', 'KY', 'MS', 'TN', 'AR', 'LA', 'OK', 'TX'])

midwest = pd.Series(['IL', 'IN', 'MN', 'MO', 'NE', 'ND',
                     'SD', 'IA', 'KS', 'MI', 'OH', 'WI'])

west = pd.Series(['AZ', 'CO', 'ID', 'MT', 'NV', 'NM',
                  'UT', 'WY', 'AK', 'CA', 'HI', 'OR', 'WA'])
```

Using .append()

```
east = northeast.append(south)
print(east)
```

```
0    CT    7    DC
1    ME    8    WV
2    MA    9    AL
3    NH   10    KY
4    RI   11    MS
5    VT   12    TN
6    NJ   13    AR
7    NY   14    LA
8    PA   15    OK
9    DE   16    TX
10   FL      dtype: object
11   GA
12   MD
13   NC
14   SC
15   VA
```

The appended Index

```
print(east.index)
```

```
Int64Index([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  0,  1,  2,  3,  4,  
            5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16], dtype='i')
```

```
print(east.loc[3])
```

```
3    NH  
3    MD  
dtype: object
```

Using .reset_index()

```
new_east = northeast.append(south).reset_index(drop=True)  
print(new_east.head(11))
```

```
0    CT  
1    ME  
2    MA  
3    NH  
4    RI  
5    VT  
6    NJ  
7    NY  
8    PA  
9    DE  
10   FL  
dtype: object
```

```
print(new_east.index)
```

```
RangeIndex(start=0, stop=26, step=1)
```


Using concat()

```
east = pd.concat([northeast, south])  
print(east.head(11))
```

```
0    CT  
1    ME  
2    MA  
3    NH  
4    RI  
5    VT  
6    NJ  
7    NY  
8    PA  
0    DE  
1    FL  
dtype: object
```

```
print(east.index)
```

```
Int64Index([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  0,  1,  2,  3,  4,  
            5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16], dtype='int64')
```

Using ignore_index

```
new_east = pd.concat([northeast, south],  
                     ignore_index=True)  
  
print(new_east.head(11))
```

```
0    CT  
1    ME  
2    MA  
3    NH  
4    RI  
5    VT  
6    NJ  
7    NY  
8    PA  
9    DE  
10   FL  
dtype: object
```

```
print(new_east.index)
```

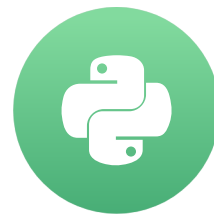
```
RangeIndex(start=0, stop=26, step=1)
```

Let's practice!

MERGING DATAFRAMES WITH PANDAS

Appending and concatenating DataFrames

MERGING DATAFRAMES WITH PANDAS



Anaconda
Instructor

Loading population data

```
import pandas as pd
pop1 = pd.read_csv('population_01.csv', index_col=0)
pop2 = pd.read_csv('population_02.csv', index_col=0)
print(type(pop1), pop1.shape)
```

```
<class 'pandas.core.frame.DataFrame'> (4, 1)
```

```
print(type(pop2), pop2.shape)
```

```
<class 'pandas.core.frame.DataFrame'> (4, 1)
```

Examining population data

```
print(pop1)
```

```
          2010 Census Population
Zip Code ZCTA
66407          479
72732         4716
50579         2405
46241        30670
```

```
print(pop2)
```

```
          2010 Census Population
Zip Code ZCTA
12776          2180
76092         26669
98360         12221
49464         27481
```

Appending population DataFrames

```
pop1.append(pop2)
```

```
      2010 Census Population
Zip Code ZCTA
66407          479
72732         4716
50579         2405
46241        30670
12776         2180
76092        26669
98360        12221
49464        27481
```

```
print(pop1.index.name, pop1.columns)
```

```
Zip Code ZCTA Index(['2010 Census Population'], dtype='object')
```

```
print(pop2.index.name, pop2.columns)
```

```
Zip Code ZCTA Index(['2010 Census Population'], dtype='object')
```

Population and unemployment data

```
population = pd.read_csv('population_00.csv',  
                          index_col=0)  
unemployment = pd.read_csv('unemployment_00.csv',  
                           index_col=0)  
print(population)
```

```
                2010 Census Population  
Zip Code ZCTA  
57538                322  
59916                130  
37660            40038  
2860             45199
```


Population and unemployment data

```
print(unemployment)
```

```
      unemployment  participants
Zip
2860             0.11         34447
46167            0.02          4800
1097             0.33           42
80808            0.07          4310
```

Appending population and unemployment

```
population.append(unemployment)
```

	2010 Census Population	participants	unemployment
57538	322.0	NaN	NaN
59916	130.0	NaN	NaN
37660	40038.0	NaN	NaN
2860	45199.0	NaN	NaN
2860	NaN	34447.0	0.11
46167	NaN	4800.0	0.02
1097	NaN	42.0	0.33
80808	NaN	4310.0	0.07

Repeated index labels

```
population.append(unemployment)
```

	2010 Census Population	participants	unemployment
57538	322.0	NaN	NaN
59916	130.0	NaN	NaN
37660	40038.0	NaN	NaN
2860	45199.0	NaN	NaN
2860	NaN	34447.0	0.11
46167	NaN	4800.0	0.02
1097	NaN	42.0	0.33
80808	NaN	4310.0	0.07

Concatenating rows

```
pd.concat([population, unemployment], axis=0)
```

	2010 Census Population	participants	unemployment
57538	322.0	NaN	NaN
59916	130.0	NaN	NaN
37660	40038.0	NaN	NaN
2860	45199.0	NaN	NaN
2860	NaN	34447.0	0.11
46167	NaN	4800.0	0.02
1097	NaN	42.0	0.33
80808	NaN	4310.0	0.07

Concatenating columns

```
pd.concat([population, unemployment], axis=1)
```

	2010 Census Population	unemployment	participants
1097	NaN	0.33	42.0
2860	45199.0	0.11	34447.0
37660	40038.0	NaN	NaN
46167	NaN	0.02	4800.0
57538	322.0	NaN	NaN
59916	130.0	NaN	NaN
80808	NaN	0.07	4310.0

Let's practice!

MERGING DATAFRAMES WITH PANDAS

Concatenation, keys, and MultiIndexes

MERGING DATAFRAMES WITH PANDAS



Anaconda
Instructor

Loading rainfall data

```
import pandas as pd

file1 = 'q1_rainfall_2013.csv'
rain2013 = pd.read_csv(file1,
                        index_col='Month',
                        parse_dates=True)

file2 = 'q1_rainfall_2014.csv'
rain2014 = pd.read_csv(file2,
                        index_col='Month',
                        parse_dates=True)
```


Examining rainfall data

```
print(rain2013)
```

```
      Precipitation
Month
Jan      0.096129
Feb      0.067143
Mar      0.061613
```

```
print(rain2014)
```

```
      Precipitation
Month
Jan      0.050323
Feb      0.082143
Mar      0.070968
```

Concatenating rows

```
pd.concat([rain2013, rain2014], axis=0)
```

	Precipitation
Jan	0.096129
Feb	0.067143
Mar	0.061613
Jan	0.050323
Feb	0.082143
Mar	0.070968

Using multi-index on rows

```
rain1314 = pd.concat([rain2013, rain2014], keys=[2013, 2014], axis=0)
print(rain1314)
```

		Precipitation
2013	Jan	0.096129
	Feb	0.067143
	Mar	0.061613
2014	Jan	0.050323
	Feb	0.082143
	Mar	0.070968

Accessing a multi-index

```
print(rain1314.loc[2014])
```

	Precipitation
Jan	0.050323
Feb	0.082143
Mar	0.070968

Concatenating columns

```
rain1314 = pd.concat([rain2013, rain2014], axis='columns')  
print(rain1314)
```

	Precipitation	Precipitation
Jan	0.096129	0.050323
Feb	0.067143	0.082143
Mar	0.061613	0.070968

Using a multi-index on columns

```
rain1314 = pd.concat([rain2013, rain2014], keys=[2013, 2014], axis='columns')  
print(rain1314)
```

	2013	2014
	Precipitation	Precipitation
Jan	0.096129	0.050323
Feb	0.067143	0.082143
Mar	0.061613	0.070968

```
rain1314[2013]
```

	Precipitation
Jan	0.096129
Feb	0.067143
Mar	0.061613

pd.concat() with dict

```
rain_dict = {2013: rain2013, 2014: rain2014}
rain1314 = pd.concat(rain_dict, axis='columns')
print(rain1314)
```

	2013	2014
	Precipitation	Precipitation
Jan	0.096129	0.050323
Feb	0.067143	0.082143
Mar	0.061613	0.070968

Let's practice!

MERGING DATAFRAMES WITH PANDAS

Outer and inner joins

MERGING DATAFRAMES WITH PANDAS



Anaconda
Instructor

```
import numpy as np
import pandas as pd
A = np.arange(8).reshape(2,4) + 0.1
print(A)
```

```
[[ 0.1  1.1  2.1  3.1]
 [ 4.1  5.1  6.1  7.1]]
```

```
B = np.arange(6).reshape(2,3) + 0.2
print(B)
```

```
[[ 0.2  1.2  2.2]
 [ 3.2  4.2  5.2]]
```

```
C = np.arange(12).reshape(3,4) + 0.3
print(C)
```

```
[[ 0.3  1.3  2.3  3.3]
 [ 4.3  5.3  6.3  7.3]
 [ 8.3  9.3 10.3 11.3]]
```

Stacking arrays horizontally

```
np.hstack([B, A])
```

```
array([[ 0.2,  1.2,  2.2,  0.1,  1.1,  2.1,  3.1],  
       [ 3.2,  4.2,  5.2,  4.1,  5.1,  6.1,  7.1]])
```

```
np.concatenate([B, A], axis=1)
```

```
array([[ 0.2,  1.2,  2.2,  0.1,  1.1,  2.1,  3.1],  
       [ 3.2,  4.2,  5.2,  4.1,  5.1,  6.1,  7.1]])
```

Stacking arrays vertically

```
np.vstack([A, C])
```

```
array([[ 0.1,  1.1,  2.1,  3.1],  
       [ 4.1,  5.1,  6.1,  7.1],  
       [ 0.3,  1.3,  2.3,  3.3],  
       [ 4.3,  5.3,  6.3,  7.3],  
       [ 8.3,  9.3, 10.3, 11.3]])
```

```
np.concatenate([A, C], axis=0)
```

```
array([[ 0.1,  1.1,  2.1,  3.1],  
       [ 4.1,  5.1,  6.1,  7.1],  
       [ 0.3,  1.3,  2.3,  3.3],  
       [ 4.3,  5.3,  6.3,  7.3],  
       [ 8.3,  9.3, 10.3, 11.3]])
```

```
np.concatenate([A, B], axis=0) # incompatible columns
```

```
ValueError                                Traceback (most recent call last)
  1 np.concatenate([A, B], axis=0) # incompatible columns
ValueError: all the input array dimensions except for
the concatenation axis must match exactly
```

```
np.concatenate([A, C], axis=1) # incompatible rows
```

```
ValueError                                Traceback (most recent call last)
  1 np.concatenate([A, C], axis=1) # incompatible rows
ValueError: all the input array dimensions except for
the concatenation axis must match exactly
```

```
population = pd.read_csv('population_00.csv', index_col=0)
unemployment = pd.read_csv('unemployment_00.csv', index_col=0)
print(population)
print(unemployment)
```

```
                2010 Census Population
Zip Code ZCTA
57538                322
59916                130
37660            40038
2860            45199
```

```
      unemployment  participants
Zip
2860            0.11        34447
46167            0.02         4800
1097             0.33           42
80808            0.07        4310
```

Converting to arrays

```
population_array = np.array(population)
print(population_array) # Index info is lost
```

```
[[ 322]
 [ 130]
 [40038]
 [45199]]
```

```
unemployment_array = np.array(unemployment)
print(population_array)
```

```
[[ 1.10000000e-01  3.44470000e+04]
 [ 2.00000000e-02  4.80000000e+03]
 [ 3.30000000e-01  4.20000000e+01]
 [ 7.00000000e-02  4.31000000e+03]]
```

Manipulating data as arrays

```
print(np.concatenate([population_array,  
                      unemployment_array], axis=1))
```

```
[[ 3.22000000e+02  1.10000000e-01  3.44470000e+04]  
 [ 1.30000000e+02  2.00000000e-02  4.80000000e+03]  
 [ 4.00380000e+04  3.30000000e-01  4.20000000e+01]  
 [ 4.51990000e+04  7.00000000e-02  4.31000000e+03]]
```


Joins

- Joining tables: Combining rows of multiple tables
- Outer join
 - Missing fields filled with NaN
 - Union of index sets (all labels, no repetition)
- Inner join
 - Intersection of index sets (only common labels)

Concatenation and inner join

```
pd.concat([population, unemployment], axis=1, join='inner')
```

	2010 Census Population	unemployment	participants
2860	45199	0.11	34447

Concatenation and outer join

```
pd.concat([population, unemployment], axis=1, join='outer')
```

	2010 Census Population	unemployment	participants
1097	NaN	0.33	42.0
2860	45199.0	0.11	34447.0
37660	40038.0	NaN	NaN
46167	NaN	0.02	4800.0
57538	322.0	NaN	NaN
59916	130.0	NaN	NaN
80808	NaN	0.07	4310.0

Inner join on other axis

```
pd.concat([population, unemployment], join='inner', axis=0)
```

```
Empty DataFrame
```

```
Columns: []
```

```
Index: [2860, 46167, 1097, 80808, 57538, 59916, 37660, 2860]
```

Let's practice!

MERGING DATAFRAMES WITH PANDAS