Pivot Tables

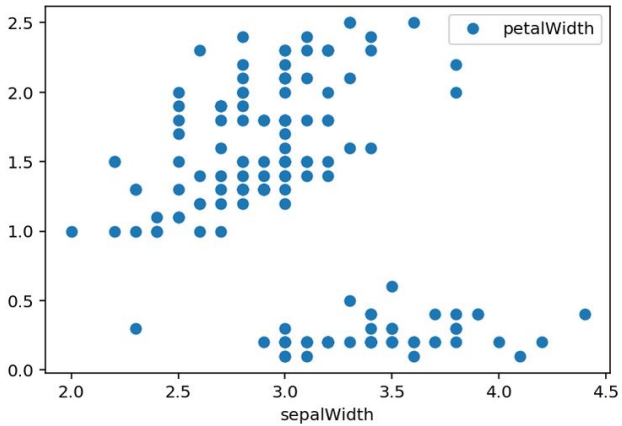
Python Module 10

Template by: Laurent Lessard

Data format matters!

So far, it has been relatively easy to extract the information we need from our datasets:

```
df.plot(x="sepalWidth", y="petalWidth", style="o")
```




```
df.sepalWidth.mean()
```

```
3.057333333333334
```

```
i = df.sepalWidth < df.sepalWidth.mean()
df[i].petalWidth.median()
```

```
1.4
```

What enables this?

Having one example per row:

```
# unemployment data
df.head()
```

	year	month	industry	unemployed	rate
0	2000	1	Government	430	2.1
1	2000	2	Government	409	2.0
2	2000	3	Government	311	1.5
3	2000	4	Government	269	1.3
4	2000	5	Government	370	1.9

This is tidy data, where each row is a single data point and each column is a single measured variable.

Reference: https://en.wikipedia.org/wiki/Tidy_data

Unemployment dataset

	year	month	industry	unemployed	rate
0	2000	1	Government	430	2.1
1	2000	2	Government	409	2.0
2	2000	3	Government	311	1.5
3	2000	4	Government	269	1.3
4	2000	5	Government	370	1.9

- **Year:** 2000, 2001, …, 2010
- **Month:** 1, 2, …, 12
- **Industry:** 14 categories total:
  Government, Finance, Construction,…
- **Unemployed:** Number unemployed (in thousands)
- **Rate:** Unemployment rate (in percent)

How do we aggregate data?

	year	month	industry	unemployed	rate
0	2000	1	Government	430	2.1
1	2000	2	Government	409	2.0
2	2000	3	Government	311	1.5
3	2000	4	Government	269	1.3
4	2000	5	Government	370	1.9

Unemployment by industry and year:

year	2000	2001	2002	2003	2004	2005
industry						
Agriculture	134.000000	153.083333	138.750000	139.833333	128.833333	104.250000
Business services	573.500000	768.250000	1008.916667	1041.833333	860.916667	792.416667
Construction	512.833333	609.166667	799.666667	809.583333	769.000000	712.250000
Education and Health	383.000000	463.500000	570.083333	639.833333	617.083333	626.916667
Finance	208.333333	251.833333	319.583333	318.583333	332.416667	271.666667
Government	422.333333	430.416667	512.250000	567.500000	547.583333	534.250000

Every row has *multiple* observations (and every cell too!)

Aggregate with `pivot_table`

	year	month	industry	unemployed	rate
0	2000	1	Government	430	2.1
1	2000	2	Government	409	2.0
2	2000	3	Government	311	1.5
3	2000	4	Government	269	1.3
4	2000	5	Government	370	1.9

```
df.pivot_table(
    index='industry',
    columns='year',
    values='unemployed',
    aggfunc='mean')
```

year	2000	2001	2002	2003	2004	2005
industry						
Agriculture	134.000000	153.083333	138.750000	139.833333	128.833333	104.250000
Business services	573.500000	768.250000	1008.916667	1041.833333	860.916667	792.416667
Construction	512.833333	609.166667	799.666667	809.583333	769.000000	712.250000
Education and Health	383.000000	463.500000	570.083333	639.833333	617.083333	626.916667
Finance	208.333333	251.833333	319.583333	318.583333	332.416667	271.666667
Government	422.333333	430.416667	512.250000	567.500000	547.583333	534.250000

Pivot table syntax

```
df.pivot_table( index='industry', columns='year', values='unemployed', aggfunc='mean')
```

row categories

column categories

what to aggregate

how to aggregate values ('mean' is the default)

year	2000	2001	2002	2003	2004	2005
industry						
Agriculture	134.000000	153.083333	138.750000	139.833333	128.833333	104.250000
Business services	573.500000	768.250000	1008.916667	1041.833333	860.916667	792.416667
Construction	512.833333	609.166667	799.666667	809.583333	769.000000	712.250000
Education and Health	383.000000	463.500000	570.083333	639.833333	617.083333	626.916667
Finance	208.333333	251.833333	319.583333	318.583333	332.416667	271.666667
Government	422.333333	430.416667	512.250000	567.500000	547.583333	534.250000

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.pivot_table.html

Example: No columns

	year	month	industry	unemployed	rate
0	2000	1	Government	430	2.1
1	2000	2	Government	409	2.0
2	2000	3	Government	311	1.5
3	2000	4	Government	269	1.3
4	2000	5	Government	370	1.9

```
df.pivot_table( index='year', values='unemployed' )
```

year	unemployed
2000	375.553571
2001	452.958333
2002	560.160714
2003	580.904762
2004	533.089286
2005	494.648810

- aggfunc defaults to using 'mean'.
- For each year, shows unemployment averaged across month and industry.

Example: Multiple values

	year	month	industry	unemployed	rate
0	2000	1	Government	430	2.1
1	2000	2	Government	409	2.0
2	2000	3	Government	311	1.5
3	2000	4	Government	269	1.3
4	2000	5	Government	370	1.9

```
df.pivot_table( index='industry',
                values=['rate','unemployed'] )
```

industry	rate	unemployed
Agriculture	9.840984	132.270492
Business services	6.919672	910.344262
Construction	9.426230	868.221311
Education and Health	3.420492	633.418033
Finance	3.540164	329.065574
Government	2.581148	538.795082

- Specifying a list of values shows them as separate columns in the output

Example: Multiple functions

	year	month	industry	unemployed	rate
0	2000	1	Government	430	2.1
1	2000	2	Government	409	2.0
2	2000	3	Government	311	1.5
3	2000	4	Government	269	1.3
4	2000	5	Government	370	1.9

```
summary = df.pivot_table(
     index='industry',
      values=['rate','unemployed'],
      aggfunc=['mean','median'] )
```

	mean		median	
	rate	unemployed	rate	unemployed
industry				
Agriculture	9.840984	132.270492	9.60	125.0
Business services	6.919672	910.344262	6.45	843.0
Construction	9.426230	868.221311	8.10	712.5
Education and Health	3.420492	633.418033	3.20	592.0
Finance	3.540164	329.065574	3.30	303.0
Government	2.581148	538.795082	2.40	501.0

Hierarchical index: use extra slices to reach the desired level.

```
summary['median']['rate']
```

```
industry
Agriculture                 9.60
Business services           6.45
Construction                8.10
Education and Health        3.20
```

Hierarchical indexing

	year	month	industry	unemployed	rate
0	2000	1	Government	430	2.1
1	2000	2	Government	409	2.0
2	2000	3	Government	311	1.5
3	2000	4	Government	269	1.3
4	2000	5	Government	370	1.9

```
summary = df.pivot_table( index=['year','month'],
                          aggfunc=['mean','median'] )
```

		mean		median	
		rate	unemployed	rate	unemployed
year	month				
2000	1	4.835714	426.714286	4.10	313.5
	2	4.957143	422.857143	4.05	305.5
	3	4.592857	405.857143	3.65	279.0
	4	3.921429	346.071429	3.90	254.5
	5	3.864286	357.714286	3.90	312.0
	6	3.792857	379.714286	3.15	307.0
	7	3.864286	386.285714	3.65	306.0
	8	4.035714	381.142857	3.60	329.5
	9	4.057143	355.357143	4.00	308.5
	10	3.971429	341.857143	3.25	282.5
	11	4.064286	353.857143	3.20	312.0
	12	4.257143	349.214286	3.50	246.5
2001	1	4.942857	447.357143	3.85	330.0
	2	5.192857	442.071429	4.40	270.5

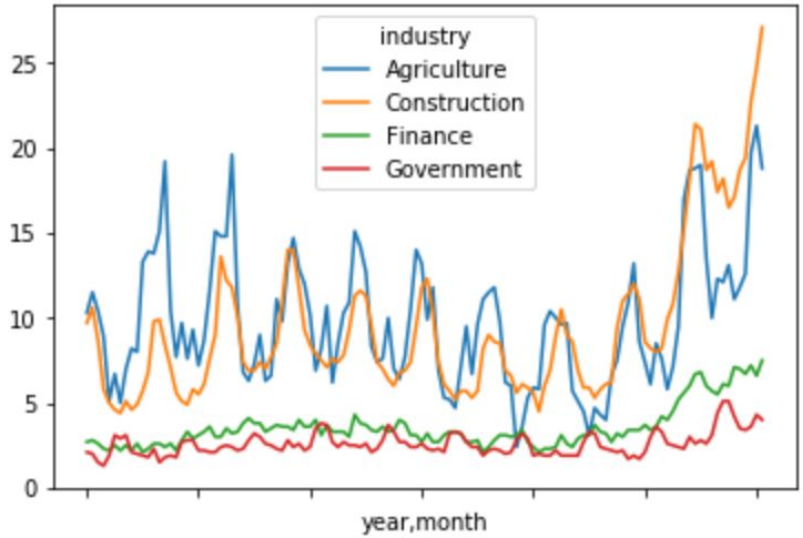
```
# extract means for year 2000
summary.loc[2000,'mean']
```

	rate	unemployed
month		
1	4.835714	426.714286
2	4.957143	422.857143
3	4.592857	405.857143
4	3.921429	346.071429
5	3.864286	357.714286

Visualization

	year	month	industry	unemployed	rate
0	2000	1	Government	430	2.1
1	2000	2	Government	409	2.0
2	2000	3	Government	311	1.5
3	2000	4	Government	269	1.3
4	2000	5	Government	370	1.9

```
summary = df.pivot_table(
    index=['year', 'month'],
    columns='industry',
    values='rate' )
summary[ ['Agriculture','Construction','Finance','Government'] ].plot()
```



`pivot_table` sorts the index by the list ordering: It sorts `year` first, then `month`. Specifying `index=['month','year']` would result in a table with index [Jan. 2000, Jan. 2001, …].