Lecture 16: Reshaping data in pandas

Reshaping data

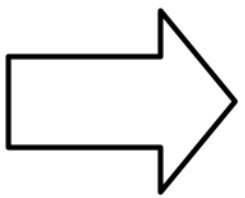
Recall the literacy rate data:

```
# A tibble: 260 × 38
   Adult (15+) literacy rate …¹ `1975` `1976` `1977` `1978` `1979`
`1980` `1981`
   <chr>                        <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
<dbl>  <dbl>
 1 Afghanistan                     NA     NA     NA     NA   4.99    NA
NA
 2 Albania                         NA     NA     NA     NA   NA      NA
NA
 3 Algeria                         NA     NA     NA     NA   NA      NA
NA
 4 Andorra                         NA     NA     NA     NA   NA      NA
NA
 5 Angola                          NA     NA     NA     NA   NA      NA
```

How did we want to restructure this data?

Reshaping data in R

id	bp1	bp2
A	100	120
B	140	115
C	120	125

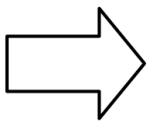


id	measurement	value
A	bp1	100
A	bp2	120
B	bp1	140
B	bp2	115
C	bp1	120
C	bp2	125

What did our R code look like to reshape this data?

Reshaping data in R

id	bp1	bp2
A	100	120
B	140	115
C	120	125



id	measurement	value
A	bp1	100
A	bp2	120
B	bp1	140
B	bp2	115
C	bp1	120
C	bp2	125

```
df |>
  pivot_longer(
    cols = bp1:bp2,
    names_to = "measurement",
    values_to = "value"
  )
```

(Image from *R for Data Science*)

Lengthing data in Python

```
1 import pandas as pd
2
3 df1 = pd.DataFrame({
4   'id' : ['A', 'B', 'C'],
5   'bp1' : [100, 140, 120],
6   'bp2' : [120, 115, 125]
7 })
8
9 df1
```

	id	bp1	bp2
0	A	100	120
1	B	140	115
2	C	120	125

```
1 df1.melt(id_vars = 'id', var_name = 'measurement', value_name = 'valu
```

	id	measurement	value
0	A	bp1	100
1	B	bp1	140
2	C	bp1	120
3	A	bp2	120
4	B	bp2	115
5	C	bp2	125

Reshaping data in R

```r
litF |>
  rename(country = starts_with("Adult")) |>
  pivot_longer(
    cols = -country,
    names_to = "year",
    values_to = "literacy_rate"
  ) |>
  drop_na(literacy_rate)
```

```
# A tibble: 571 × 3
   country     year  literacy_rate
   <chr>       <chr>         <dbl>
 1 Afghanistan 1979           4.99
 2 Afghanistan 2011          13
 3 Albania     2001          98.3
 4 Albania     2008          94.7
 5 Albania     2011          95.7
 6 Algeria     1987          35.8
 7 Algeria     2002          60.1
 8 Algeria     2006          63.9
 9 Angola      2001          54.2
10 Angola      2011          58.6
# ℹ 561 more rows
```

What would the corresponding Python code look like?

Reshaping data in Python

```
litF = r.litF

# first, need to rename the first column
litF.rename(columns={litF.columns[0]: 'Country'})
```

```
                          Country  1975  1976  ...  2009  2010       2011
0                     Afghanistan   NaN   NaN  ...   NaN   NaN   13.00000
1                         Albania   NaN   NaN  ...   NaN   NaN   95.69148
2                         Algeria   NaN   NaN  ...   NaN   NaN        NaN
3                         Andorra   NaN   NaN  ...   NaN   NaN        NaN
4                          Angola   NaN   NaN  ...   NaN   NaN   58.60846
..                            ...   ...   ...  ...   ...   ...        ...
255          Virgin Islands (U.S.)   NaN   NaN  ...   NaN   NaN        NaN
256  Yemen Arab Republic (Former)   NaN   NaN  ...   NaN   NaN        NaN
257     Yemen Democratic (Former)   NaN   NaN  ...   NaN   NaN        NaN
258                      Yugoslavia   NaN   NaN  ...   NaN   NaN        NaN
259                           Åland   NaN   NaN  ...   NaN   NaN        NaN

[260 rows x 38 columns]
```

Reshaping data in Python

```
litF = r.litF

# rename the first column
# then melt to make the data longer
(litF.rename(columns={litF.columns[0]: 'Country'})
     .melt(id_vars = 'Country',
           var_name = 'year',
           value_name = 'count'))
```

```
                          Country  year  count
0                     Afghanistan  1975    NaN
1                         Albania  1975    NaN
2                         Algeria  1975    NaN
3                         Andorra  1975    NaN
4                          Angola  1975    NaN
...                           ...   ...    ...
9615          Virgin Islands (U.S.)  2011    NaN
9616  Yemen Arab Republic (Former)  2011    NaN
9617     Yemen Democratic (Former)  2011    NaN
9618                      Yugoslavia  2011    NaN
9619                           Åland  2011    NaN

[9620 rows x 3 columns]
```

Reshaping data in Python

```
litF = r.litF

# rename the first column
# then melt to make the data longer
(litF.rename(columns={litF.columns[0]: 'Country'})
     .melt(id_vars = 'Country',
           var_name = 'year',
           value_name = 'count')
     .dropna())
```

```
                  Country  year      count
30            Burkina Faso  1975   3.182766
37    Central African Rep.  1975   8.399576
99                  Kuwait  1975  48.015214
191                 Turkey  1975  45.098921
197   United Arab Emirates  1975  38.124870
...                    ...   ...        ...
9562               Vanuatu  2011  81.553540
9564    West Bank and Gaza  2011  92.616180
9565               Vietnam  2011  91.383460
9566          Yemen, Rep.  2011  48.539050
9568              Zimbabwe  2011  80.065659

[571 rows x 3 columns]
```

What if I want to arrange the rows alphabetically by country?

Reshaping data in Python

```
litF = r.litF

# rename the first column
# then melt to make the data longer
(litF.rename(columns={litF.columns[0]: 'Country'})
     .melt(id_vars = 'Country',
           var_name = 'year',
           value_name = 'count')
     .dropna()
     .sort_values(by = ['Country', 'year']))
```

```
         Country  year      count
1040  Afghanistan  1979   4.987460
9360  Afghanistan  2011  13.000000
6761      Albania  2001  98.252274
8581      Albania  2008  94.681814
9361      Albania  2011  95.691480
...           ...   ...        ...
7227       Zambia  2002  61.839278
8527       Zambia  2007  51.786967
2028     Zimbabwe  1982  71.853928
4628     Zimbabwe  1992  78.517018
9568     Zimbabwe  2011  80.065659

[571 rows x 3 columns]
```

`pivot_longer` in R

Consider the following example data:

```
  id bp_1 bp_2 hr_1 hr_2
1  1  100  120   60   77
2  2  120  115   75   81
3  3  125  130   80   93
```

What if we want the data to look like this:

```
# A tibble: 12 × 4
      id measurement stage value
   <dbl> <chr>       <chr> <dbl>
 1     1 bp          1       100
 2     1 bp          2       120
 3     1 hr          1        60
 4     1 hr          2        77
 5     2 bp          1       120
 6     2 bp          2       115
 7     2 hr          1        75
 8     2 hr          2        81
 9     3 bp          1       125
10     3 bp          2       130
```

pivot_longer in R

```r
df2
```

```
  id bp_1 bp_2 hr_1 hr_2
1  1  100  120   60   77
2  2  120  115   75   81
3  3  125  130   80   93
```

```r
df2 |>
  pivot_longer(cols = -id,
               names_to = c("measurement", "stage"),
               names_sep = "_",
               values_to = "value")
```

```
# A tibble: 12 × 4
      id measurement stage value
   <dbl> <chr>       <chr> <dbl>
 1     1 bp          1       100
 2     1 bp          2       120
 3     1 hr          1        60
 4     1 hr          2        77
 5     2 bp          1       120
 6     2 bp          2       115
 7     2 hr          1        75
 8     2 hr          2        81
 9     3 bp          1       125
10     3 bp          2       130
11     3 hr          1        80
```

pivot_longer in R

```r
df2 |>
  pivot_longer(cols = -id,
               names_to = c("measurement", "stage"),
               names_sep = "_",
               values_to = "value")
```

Step 1: Pivot

```
# A tibble: 6 × 3
     id measurement value
  <dbl> <chr>       <dbl>
1     1 bp_1          100
2     1 bp_2          120
3     1 hr_1           60
4     1 hr_2           77
5     2 bp_1          120
6     2 bp_2          115
```

Step 2: Separate columns

```
# A tibble: 6 × 4
     id measurement stage value
  <dbl> <chr>       <chr> <dbl>
1     1 bp          1       100
2     1 bp          2       120
3     1 hr          1        60
4     1 hr          2        77
5     2 bp          1       120
6     2 bp          2       115
```

In Python

Step 1: Melt

```
1 df2 = r.df2
2
3 df2_new = df2.melt(id_vars = 'id', var_name = 'measurement', value_na
4 df2_new
```

	id	measurement	value
0	1.0	bp_1	100.0
1	2.0	bp_1	120.0
2	3.0	bp_1	125.0
3	1.0	bp_2	120.0
4	2.0	bp_2	115.0
5	3.0	bp_2	130.0
6	1.0	hr_1	60.0
7	2.0	hr_1	75.0
8	3.0	hr_1	80.0
9	1.0	hr_2	77.0
10	2.0	hr_2	81.0
11	3.0	hr_2	93.0

In Python

Step 2: Separate columns

```
df2 = r.df2

df2_new = df2.melt(id_vars = 'id', var_name = 'measurement', value_na
df2_new['measurement'].str.split('_', expand=True)
```

	0	1
0	bp	1
1	bp	1
2	bp	1
3	bp	2
4	bp	2
5	bp	2
6	hr	1
7	hr	1
8	hr	1
9	hr	2
10	hr	2
11	hr	2

In Python

Step 2: Separate columns

```
1 df2 = r.df2
2
3 df2_new = df2.melt(id_vars = 'id', var_name = 'measurement', value_na
4 df2_new[['measurement', 'stage']] = df2_new['measurement'].str.split(
5 df2_new
```

```
     id measurement  value stage
0   1.0          bp  100.0     1
1   2.0          bp  120.0     1
2   3.0          bp  125.0     1
3   1.0          bp  120.0     2
4   2.0          bp  115.0     2
5   3.0          bp  130.0     2
6   1.0          hr   60.0     1
7   2.0          hr   75.0     1
8   3.0          hr   80.0     1
9   1.0          hr   77.0     2
10  2.0          hr   81.0     2
11  3.0          hr   93.0     2
```

Pivot wider

```
1 air_quality
```

```
                      date.utc location  value
1825  2019-06-21 00:00:00+00:00  FR04014   20.0
1826  2019-06-20 23:00:00+00:00  FR04014   21.8
1827  2019-06-20 22:00:00+00:00  FR04014   26.5
1828  2019-06-20 21:00:00+00:00  FR04014   24.9
1829  2019-06-20 20:00:00+00:00  FR04014   21.4
```

What if I want a separate column for each location?

Pivot wider

```
1 air_quality.pivot(index = 'date.utc', columns = 'location', values =
```

location	BETR801	FR04014	London Westminster
date.utc			
2019-04-09 01:00:00+00:00	22.5	24.4	NaN
2019-04-09 02:00:00+00:00	53.5	27.4	67.0
2019-04-09 03:00:00+00:00	54.5	34.2	67.0
2019-04-09 04:00:00+00:00	34.5	48.5	41.0
2019-04-09 05:00:00+00:00	46.5	59.5	41.0
...	...	...	...
2019-06-20 20:00:00+00:00	NaN	21.4	NaN
2019-06-20 21:00:00+00:00	NaN	24.9	NaN
2019-06-20 22:00:00+00:00	NaN	26.5	NaN
2019-06-20 23:00:00+00:00	NaN	21.8	NaN
2019-06-21 00:00:00+00:00	NaN	20.0	NaN

Class activity

https://sta279-f23.github.io/class_activities/ca_lecture_16.html