Durga Swaroop Perla

medium.com/@durgaswaroop/reshaping-pandas-dataframes-melt-and-unmelt-9f57518c7738

November 27, 2019

Reshaping Pandas Data frames with Melt & Pivot

Pandas is a wonderful data manipulation library in python. Working in the field of Data science and Machine learning, I find myself using Pandas pretty much everyday. It's an invaluable tool for data analysis and manipulation.

In this short article, I will show you what Melt and Reverse melt (Unmelt) are in Pandas, and how you can use them for reshaping data frames.

Happy Panda

Say, I have the data of the closing prices of stock market data of stock market closing prices of two major companies for last week as follows:

```
+-----+--------+-------+
| Day | Google | Apple |
+:===:+:======:+:=====:+
| MON | 1129   | 191   |
+-----+--------+-------+
| TUE | 1132   | 192   |
+-----+--------+-------+
| WED | 1134   | 190   |
+-----+--------+-------+
| THU | 1152   | 190   |
+-----+--------+-------+
| FRI | 1152   | 188   |
+-----+--------+-------+
```

For an analysis I want to do I need the names of the companies Google & Apple to appear in a single column with the stock price as another column, as shown below.

```
+-----+---------+---------------+
| Day | Company | Closing Price |
+:===:+:=======:+:=============:+
| MON | Google  | 1129          |
+-----+---------+---------------+
| TUE | Google  | 1132          |
+-----+---------+---------------+
| WED | Google  | 1134          |
+-----+---------+---------------+
| THU | Google  | 1152          |
+-----+---------+---------------+
| FRI | Google  | 1152          |
+-----+---------+---------------+
| MON | Apple   | 191           |
+-----+---------+---------------+
| TUE | Apple   | 192           |
+-----+---------+---------------+
| WED | Apple   | 190           |
+-----+---------+---------------+
| THU | Apple   | 190           |
+-----+---------+---------------+
| FRI | Apple   | 188           |
+-----+---------+---------------+
```

This is exactly where `melt` comes into picture. Melt is used for converting a bunch of columns into a single row, which is exactly what I need here.

Let's see how we can do this:

Melt:

First we need to import pandas

```
import pandas as pd
```

Now, we'll create the Dataframe with the data I need:

```
df = pd.DataFrame(data = {
   'Day' : ['MON', 'TUE', 'WED', 'THU', 'FRI'],
   'Google' : [1129,1132,1134,1152,1152],
   'Apple' : [191,192,190,190,188]
})
```

And this will get us the dataframe we need as follows:

```
+---+-----+--------+-------+
|   | Day | Google | Apple |
+:=:+:===:+:=======:+:=====:+
| 0 | MON | 1129   | 191   |
+---+-----+--------+-------+
| 1 | TUE | 1132   | 192   |
+---+-----+--------+-------+
| 2 | WED | 1134   | 190   |
+---+-----+--------+-------+
| 3 | THU | 1152   | 190   |
+---+-----+--------+-------+
| 4 | FRI | 1152   | 188   |
+---+-----+--------+-------+
```

Let's melt this now. To `melt` this dataframe, you call the `melt()` on the dataframe with the `id_vars` parameter set.

```
reshaped_df = df.melt(id_vars=['Day'])
```

And you're done. Your `reshaped_df` would like this now:

```
+---+-----+----------+-------+
|   | Day | variable | value |
+:=:+:===:+:=========:+:=====:+
| 0 | MON | Apple    | 191   |
+---+-----+----------+-------+
| 1 | TUE | Apple    | 192   |
+---+-----+----------+-------+
| 2 | WED | Apple    | 190   |
+---+-----+----------+-------+
| 3 | THU | Apple    | 190   |
+---+-----+----------+-------+
| 4 | FRI | Apple    | 188   |
+---+-----+----------+-------+
| 5 | MON | Google   | 1129  |
+---+-----+----------+-------+
| 6 | TUE | Google   | 1132  |
+---+-----+----------+-------+
| 7 | WED | Google   | 1134  |
+---+-----+----------+-------+
| 8 | THU | Google   | 1152  |
+---+-----+----------+-------+
| 9 | FRI | Google   | 1152  |
+---+-----+----------+-------+
```

The `id_vars` you've passed into the `melt()` method is to specify which column you want to leave untouched. Since we want the `Day` column to stay even after the melt, we set `id_vars=['Day']` .

Also, you would have noticed that the output dataframe of melt has the columns `variable` and `value` . These are the default names given by pandas for the columns. We can change this either manually with something like

```
reshaped_df.columns = [['Day', 'Company', 'Closing Price']]
```

Or we can specify the values for these columns in the `melt()` itself. Melt takes arguments `var_name` and `value_name` apart from `id_vars`. These options specify the names for the Variables column and the value column respectively.

```
reshaped_df = df.melt(id_vars=['Day'], var_name='Company', value_name='Closing Price')
```

That will give us:

```
+---+-----+---------+---------------+
|   | Day | Company | Closing Price |
+:=:+:===:+:=======:+:=============:+
| 0 | MON | Apple   | 191           |
+---+-----+---------+---------------+
| 1 | TUE | Apple   | 192           |
+---+-----+---------+---------------+
| 2 | WED | Apple   | 190           |
+---+-----+---------+---------------+
| 3 | THU | Apple   | 190           |
+---+-----+---------+---------------+
| 4 | FRI | Apple   | 188           |
+---+-----+---------+---------------+
| 5 | MON | Google  | 1129          |
+---+-----+---------+---------------+
| 6 | TUE | Google  | 1132          |
+---+-----+---------+---------------+
| 7 | WED | Google  | 1134          |
+---+-----+---------+---------------+
| 8 | THU | Google  | 1152          |
+---+-----+---------+---------------+
| 9 | FRI | Google  | 1152          |
+---+-----+---------+---------------+
```

Unmelt/Reverse Melt/Pivot

We can also do the reverse of the melt operation which is also called as `pivoting`. In Pivoting or Reverse Melting, we convert a column with multiple values into several columns of their own.

The `pivot` method on the dataframe takes two main arguments `index` and `columns`. The `index` parameter is similar to `id_vars` we have seen before i.e., It is used to specify which column you don't want to touch. The `columns` parameter is to specify which column should be used to create the new columns.

```
reshaped_df.pivot(index='Day', columns='Company')
```

Running the above command gives you the following:

```
+---------+--------------+-------+
|         |    Closing Price    |
+==========+:=============:+:=====:+
| Company | Google       | Apple |
+---------+--------------+-------+
| Day     |              |       |
+---------+--------------+-------+
| MON     | 1129         | 191   |
+---------+--------------+-------+
| TUE     | 1132         | 192   |
+---------+--------------+-------+
| WED     | 1134         | 190   |
+---------+--------------+-------+
| THU     | 1152         | 190   |
+---------+--------------+-------+
| FRI     | 1152         | 188   |
+---------+--------------+-------+
```

This is close but probably not exactly what you wanted. The `Closing Price` is an extra stacked column on top of Google & Apple. So to get exactly the reverse of melt and get the original `df` dataframe we started with, we do the following:

```
original_df = reshaped_df.pivot(index='Day', columns='Company')['Closing Price'].reset_index()
original_df.columns.name = None
```

And that gets us back to what we have started with.

```
+-----+--------+-------+
| Day | Google | Apple |
+:===:+:======:+:=====:+
| MON | 1129   | 191   |
+-----+--------+-------+
| TUE | 1132   | 192   |
+-----+--------+-------+
| WED | 1134   | 190   |
+-----+--------+-------+
| THU | 1152   | 190   |
+-----+--------+-------+
| FRI | 1152   | 188   |
+-----+--------+-------+
```

That is all for this article. I hope this was useful for you and that you'll try to use this in your data processing workflow.