

# 22

## Reshaping and Pivoting Pandas DataFrames

# Pandas 重塑和透视

主要介绍 `pivot()`、`stack()`、`unstack()` 方法



善良一点，因为你遇到的每个人都在打一场更艰苦的战斗。

*Be kind, for everyone you meet is fighting a harder battle.*

—— 柏拉图 (Plato) | 古希腊哲学家 | 424/423 ~ 348/347 BC



- ▶ `pandas.DataFrame.pivot()` 用于将数据透视成新的行和列形式的函数
- ▶ `pandas.DataFrame.stack()` 将 `DataFrame` 中的列转换为多级索引的行形式的函数
- ▶ `pandas.DataFrame.unstack()` 将 `DataFrame` 中的多级索引行转换为列形式的函数
- ▶ `pandas.melt()` 将宽格式数据转换为长格式数据的函数，将多个列“融化”成一列
- ▶ `pandas.pivot_table()` 根据指定的索引和列对数据进行透视，并使用聚合函数合并重复值的函数
- ▶ `pandas.wide_to_long()` 将宽格式数据转换为长格式数据的函数，类似于 `melt()`，但可以处理多个标识符列和前缀



## 22.1 数据帧的重塑和透视

在 Pandas 中，数据帧的重塑和透视操作是指通过重新组织数据的方式，使数据呈现出不同的结构，以满足特定的分析需求。

具体来说，数据帧重塑 (reshaping) 是指改变数据的行和列的排列方式。数据帧透视 (pivoting) 是指通过旋转数据的行和列，以重新排列数据，并根据指定的聚合函数来生成新的数据帧。这样做可以更好地展示数据的结构和统计特征。

长格式、宽格式是本章重要概念。如图 1 所示，长格式 (long format) 和宽格式 (wide format) 是两种不同的数据存储形式。如图 1 (a) 所示，长格式类似流水账，每一行代表一个观察值，比如某个学生某科目期中考试成绩。如图 1 (b) 所示，宽格式更像是“矩阵”，每一行代表一个特定观察条件，比如某个特定学生的学号。此外，宽格式数据的列用于表示不同的特征或维度，比如特定科目。显然，长格式、宽格式之间可以很容易相互转化。Pandas 提供很多方法用来完成数据帧的重塑和透视。

(a) long format			(b) wide format			
Student ID	Subject	Midterm	Subject	Art	Math	Science
1	Math	3	Student ID			
1	Art	4	1	5	4	NaN
2	Science	5	2	5	NaN	3
2	Art	3	3	NaN	4	5
3	Math	4	4	3	5	NaN
3	Science	4				
4	Art	4				
4	Math	5				

图 1. 比较长格式、宽格式

本章要介绍的重塑和透视操作如下。

`pivot()` 函数用于根据一个或多个列创建一个新的数据透视表。`pivot_table()` 与 `pivot()` 类似，它也可以执行透视操作，但是它允许对重复的索引值进行聚合，产生一个透视表。它对于处理有重复数据的情况更加适用。

`stack()` 函数用于将数据帧从宽格式转换为长格式。`melt()` 函数也可以用于将数据从宽格式转换为长格式，类似于 `stack()`。

`unstack()` 函数是 `stack()` 的逆操作，用于将数据从长格式转换为宽格式，也就是将数据从索引转换为列。

下面，我们分别介绍这几种方法。

## 22.2 长格式转换为宽格式：pivot()

`pivot()` 可以理解为一种长格式转换为宽格式的特殊情况。`pivot()` 需要指定三个参数：`index`，`columns` 和 `values`，它们分别代表新 `DataFrame` 的行索引、列索引和填充数据的值。

举个例子，图 2 左图表格为一个班级四名同学（学号分别为 1、2、3、4）的各科（Math、Art、Science）期中、期末成绩，这个表格就是所谓的长格式，相当于“流水账”。

图 2 右图则是期中考试成绩“矩阵”，行标签（`index`）为学生学号 'ID'，列标签（`columns`）为三门科目 'Subject'，数据（`values`）为其中考试成绩 'Midterm'。

由于每名学生仅仅选修两门科目，因此在图 2 右图中会看到 NaN。

进一步，图 2 右图数据帧横向求和，得到学生总成绩；而纵向求平均值，便是各科平均成绩。这是下一章要介绍的操作。

图 3 对应上述操作的代码。请大家自行提取同学各科期末考试成绩，科目为行标签，学号为列标签。

注意，使用 `pivot()` 时，必须指定 `index` 和 `columns`，这两列的值将用于创建新的行和列。

此外，请大家思考如果，如果参数 `values = ['Midterm', 'Final']`，结果会怎样？

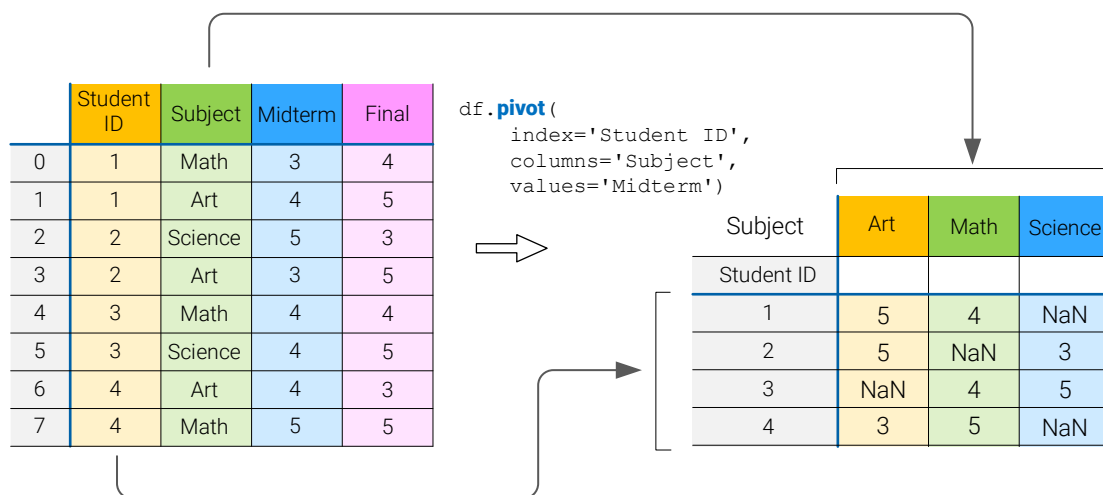


图 2. 利用 `pivot()` 提取学生各科期中考试成绩，学号为行标签，科目为列标签

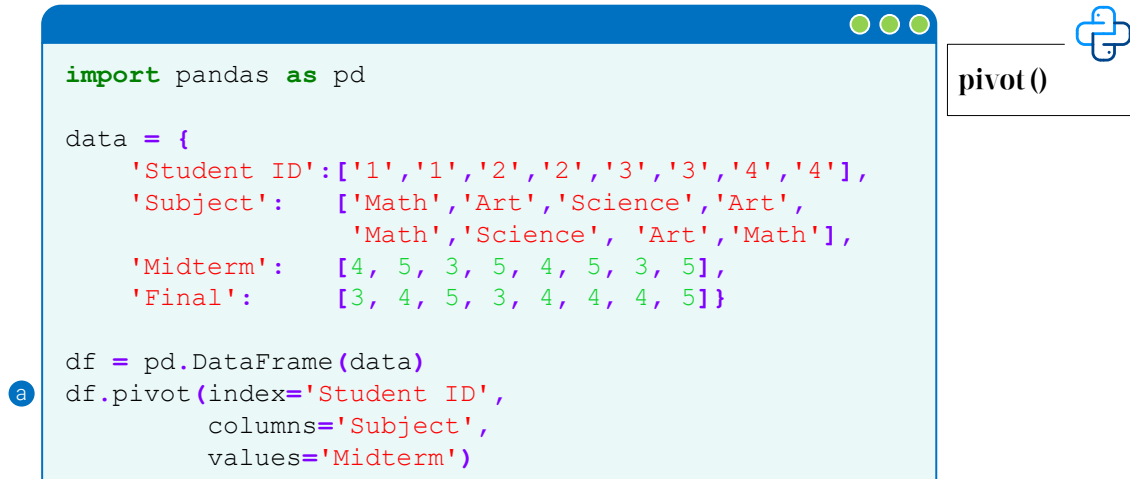


图 3. 利用 pivot() 将长格式转换为宽格式，代码

我们可以用 pivot\_table() 完成和图 2 一样的操作，df.pivot\_table(index='Student ID', columns='Subject', values='Midterm')。

和 pivot() 不同的是，pivot\_table() 可以不用指定 columns。如图 4 所示。利用 pivot\_table()，我们可以把数据帧学号、科目转化为双层行索引。

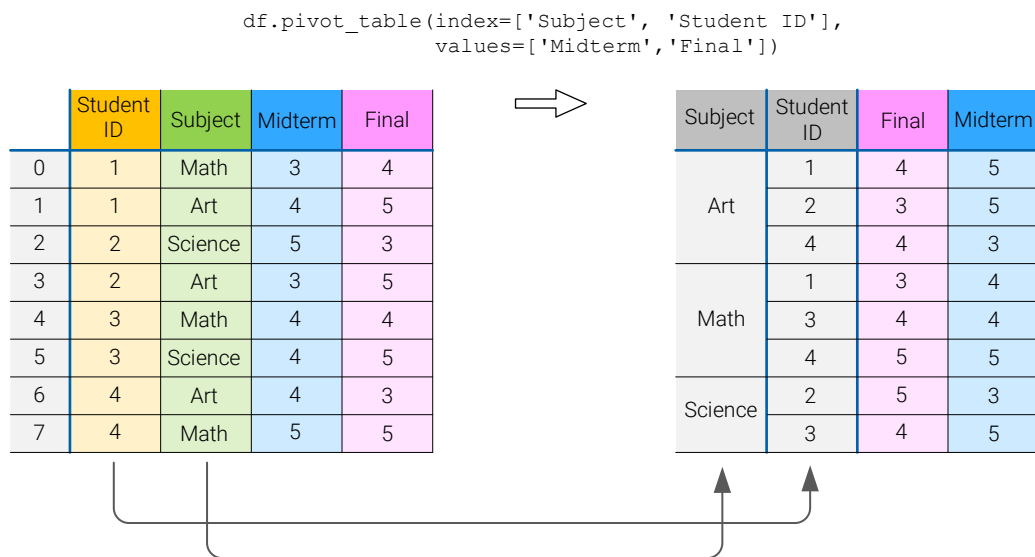
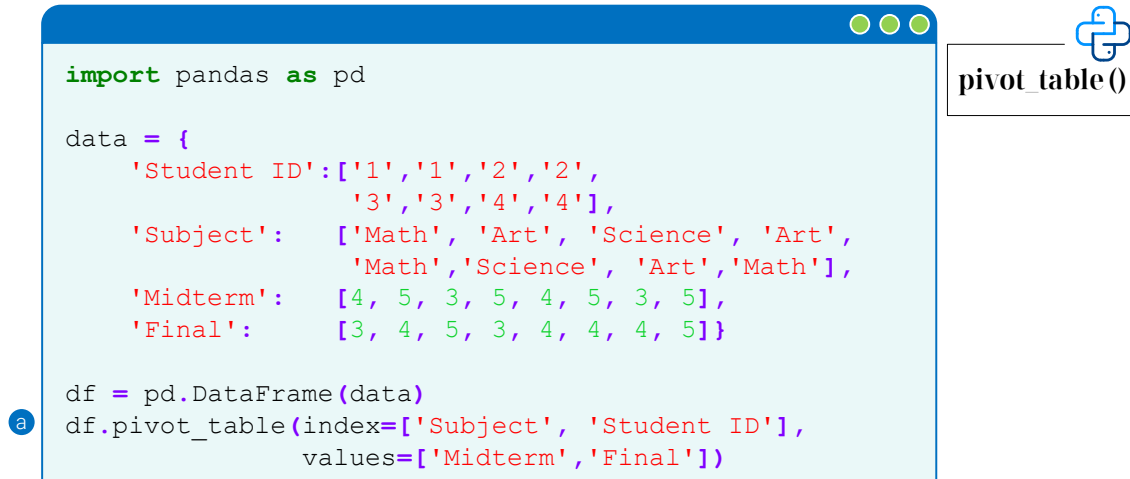
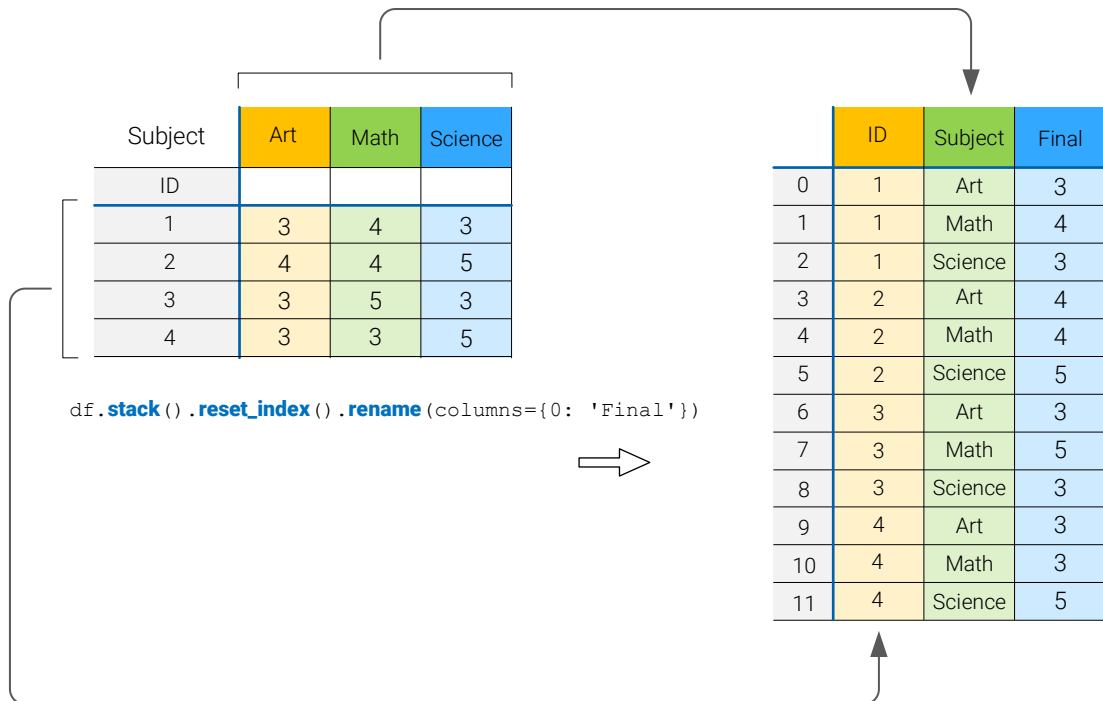


图 4. 利用 pivot\_table() 将学号、科目转化为双层行索引

图 5. 利用 `pivot_table()` 将长格式转换为宽格式，代码

## 22.3 宽格式转换为长格式：stack()

方法 `stack()` 是一种将列逐级转换为层次化索引的操作。如果 `DataFrame` 的列是层次化索引，那么 `stack()` 会将最内层的列转换为最内层的索引。该函数返回一个 `Series` 或 `DataFrame`，具体取决于原始数据的维度。

图 6. 利用 `stack()` 将宽格式转换为长格式

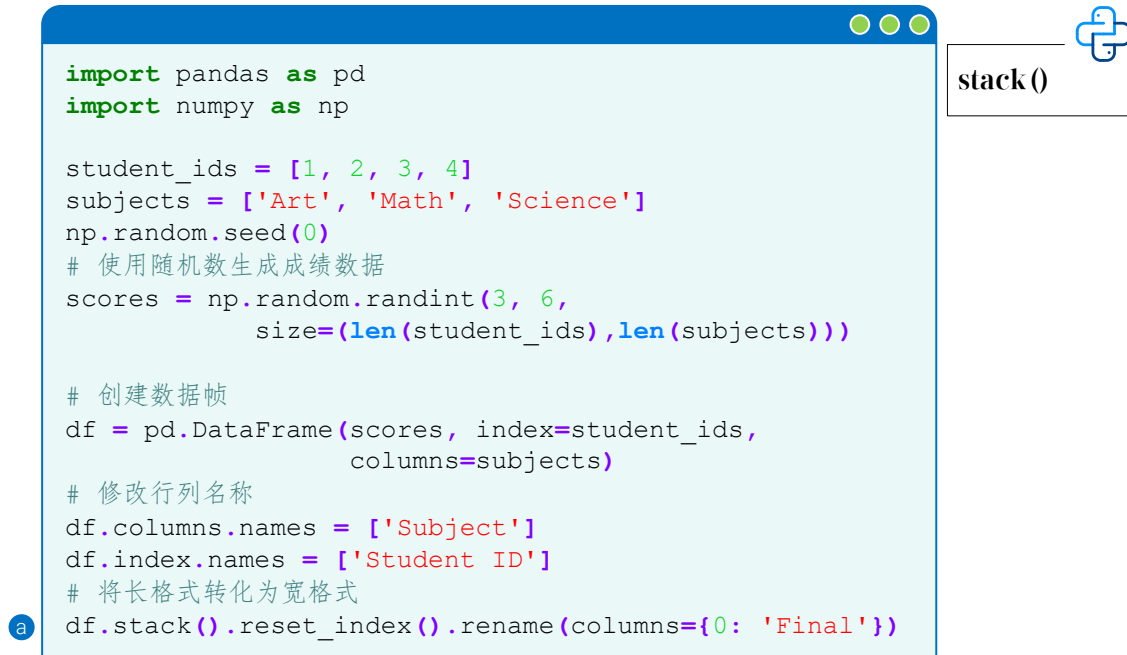


图 7. 利用 stack() 将宽格式转换为长格式，代码

melt() 将原始数据中的多列合并为一列，并根据其他列的值对新列进行重复。可以理解为 stack() 的一种泛化形式。melt() 需要指定 id\_vars 参数，表示保持不变的列，同时还可以选择 value\_vars 参数来指定哪些列需要被转换。请大家自行练习图 8 给出的示例。

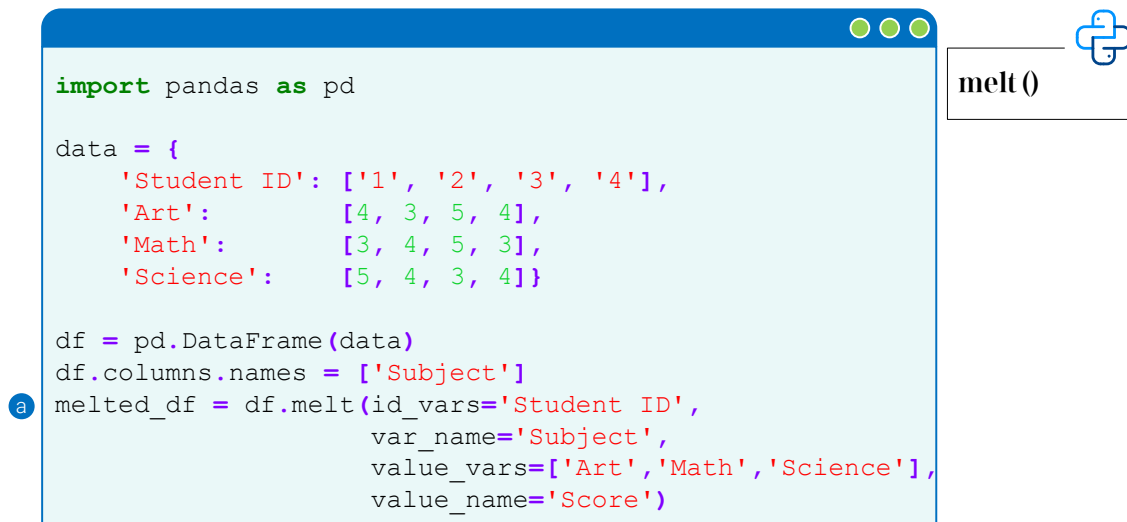


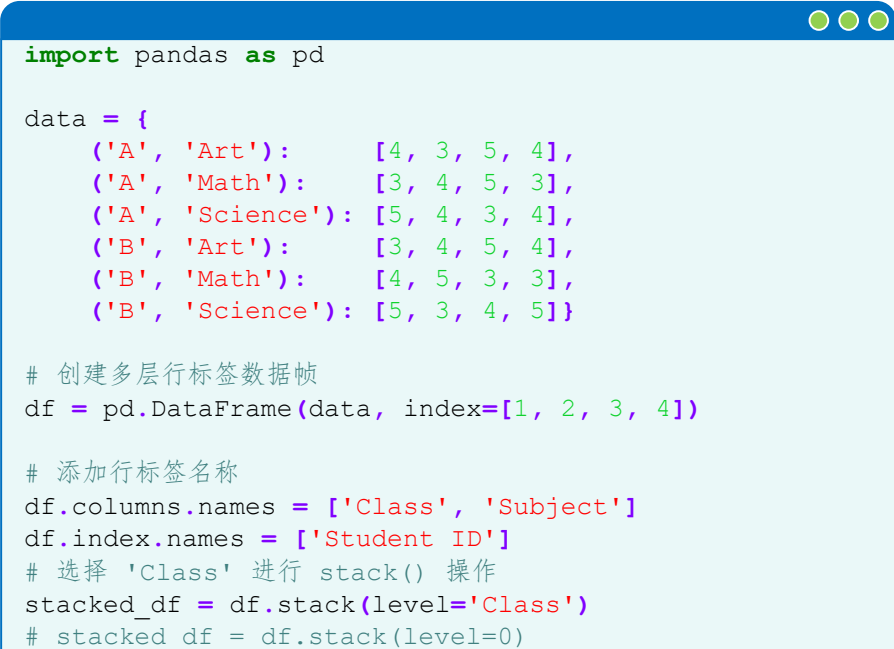
图 8. 利用 melt() 将宽格式转换为长格式，代码

### 多层列标签

如果数据帧有多层列标签，可以有选择地选取特定级别列标签完成 stack() 操作。

数据帧中 A、B 代表两个班级，每个班级 Class 有 4 名同学 (学号 1、2、3、4)，这些同学都选了 3 门课程 (Art、Math、Science)。数据帧的数据部分为同学们的期末成绩。

请大家思考如果采用 df.stack(level=["Subject"]), 结果会怎样？



```
import pandas as pd

data = {
    ('A', 'Art'): [4, 3, 5, 4],
    ('A', 'Math'): [3, 4, 5, 3],
    ('A', 'Science'): [5, 4, 3, 4],
    ('B', 'Art'): [3, 4, 5, 4],
    ('B', 'Math'): [4, 5, 3, 3],
    ('B', 'Science'): [5, 3, 4, 5]}

# 创建多层行标签数据帧
df = pd.DataFrame(data, index=[1, 2, 3, 4])

# 添加行标签名称
df.columns.names = ['Class', 'Subject']
df.index.names = ['Student ID']

# 选择 'Class' 进行 stack() 操作
stacked_df = df.stack(level='Class')
# stacked_df = df.stack(level=0)
```

图 9. 利用 stack() 将宽格式转换为长格式，选择特定列级别，代码

Class	A			B		
Subject	Art	Math	Science	Art	Math	Science
ID						
1	3	4	3	3	4	3
2	4	4	5	4	4	5
3	3	5	3	3	5	3
4	3	3	5	3	3	5

df.stack(level='Class')



ID	Subject	Art	Math	Science
	Class			
1	A	3	4	3
	B	4	4	5
2	A	3	5	3
	B	3	3	5
3	A	3	4	3
	B	4	4	5
4	A	3	5	3
	B	3	3	5

图 10. 利用 stack() 将宽格式转换为长格式，选择特定列级别

## 22.4 长格式转换为宽格式：unstack()

在 Pandas 中，`unstack()` 是一个用于数据透视的方法，它用于将一个多级索引的 Series 或 DataFrame 中的其中选定级别转换为列。这在处理分层索引数据时非常有用。

如图 11 所示，左侧的数据帧 `df` 有 3 层行索引。第 0 层为 `Class`，第 1 层为 `Student ID`，第 2 层为 `Subject`。第 0 层 `Class` 有两个值 A、B，代表有两个班级。第 1 层 `Student ID` 有四个值 1、2、3、4，代表每个班级学生的学号。第 2 层有三个值 `Art`、`Math`、`Science`，代表三个科目。

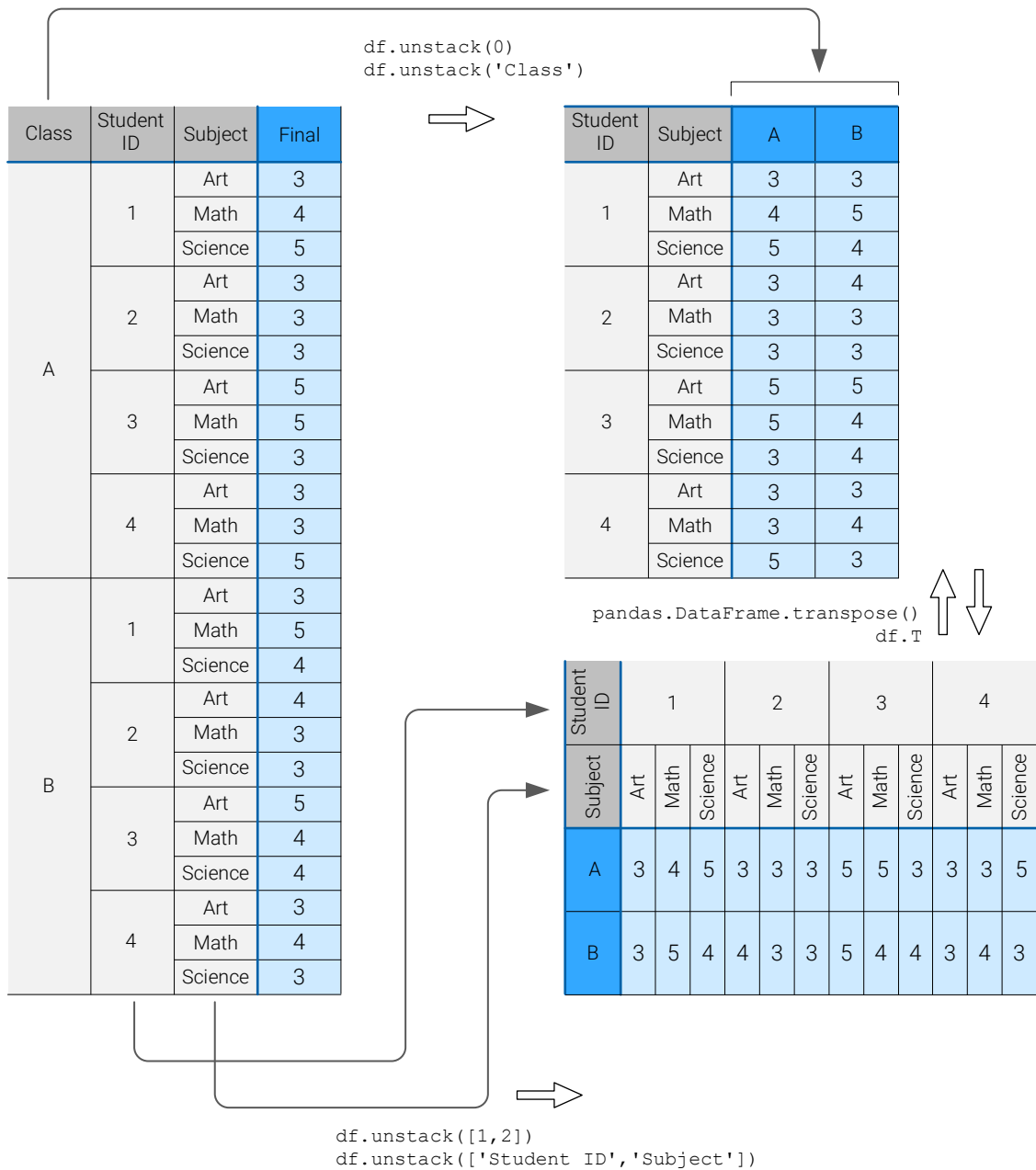


图 11. 利用 `unstack()` 将长格式转换为宽格式

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)



`df.unstack(0)` 或 `df.unstack('Class')` 将第 0 层 Class 行索引转换成两列——A、B。请大家尝试，`df.unstack(1)`、`df.unstack('Student ID')`、`df.unstack(2)`、`df.unstack('Subject')`，并比较结果。

`df.unstack([1,2])` 或 `df.unstack(['Student ID', 'Subject'])` 将第 1、2 层行索引转换成两层列标签。请大家尝试 `df.unstack([2,1])` 或 `df.unstack(['Subject', 'Student ID'])`，以及尝试其他组合，比如 `[0, 2]`、`[2, 0]`、`[0, 1]`、`[1, 0]`，并比较结果。



图 12. 利用 `unstack()` 将长格式转换为宽格式，代码

Pandas 中重塑和透视操作灵活多样，本章介绍的方法仅仅是冰山一角而已。实践中，大家可以根据需求自行学习使用其他方法操作，建议大家继续阅读如下链接。

[https://pandas.pydata.org/pandas-docs/stable/user\\_guide/reshaping.html](https://pandas.pydata.org/pandas-docs/stable/user_guide/reshaping.html)