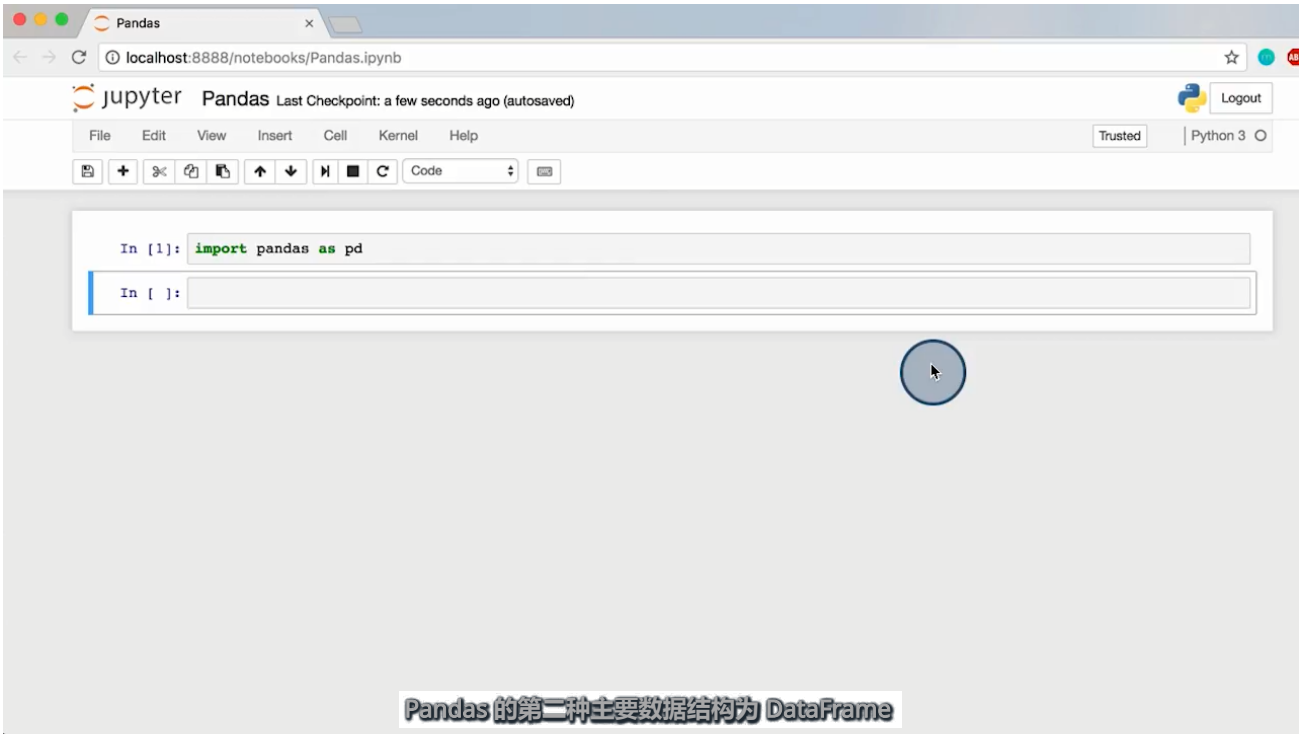
创建 Pandas DataFrame



00:00 / 05:04 1x CC

Pandas DataFrames 是具有带标签的行和列的二维数据结构，可以存储很多类型的数据。如果你熟悉 Excel 的话，可以将 Pandas DataFrames 看做类似于电子表格。在接下来的课程中，我们将开始学习如何手动地通过字典创建 Pandas DataFrame，稍后，我们将学习如何将数据文件中的数据加载到 DataFrame 中。

首先，我们将使用 Pandas Series 字典手动创建一个 DataFrame。第一步是创建 Pandas Series 字典。字典创建完毕后，我们可以将该字典传递给 `pd.DataFrame()` 函数。

我们将创建一个字典，其中包含 Alice 和 Bob 从在线商店中购买的商品。该 Pandas Series 将使用所买商品的价格作为*数据*，所买商品作为*索引*标签。我们来看看如何编写代码：

```
# We import Pandas as pd into Python
import pandas as pd

# We create a dictionary of Pandas Series
items = {'Bob' : pd.Series(data = [245, 25, 55], index = ['bike',
```

```
# We print the type of items to see that it is a dictionary
print(type(items))
```

> *class 'dict'*

字典已经创建完毕，我们可以通过将其传递给 `pd.DataFrame()` 函数，创建 DataFrame。我们将创建一个可以表示多位用户的购物车的 DataFrame，在此例中只有两位用户，即 Alice 和 Bob。

```
# We create a Pandas DataFrame by passing it a dictionary of Pand
shopping_carts = pd.DataFrame(items)

# We display the DataFrame
shopping_carts
```

	Alice	*Bob*
bike	*500.0*	*245.0*
book	*40.0*	*NaN*
glasses	*110.0*	*NaN*
pants	*45.0*	*25.0*
watch	*NaN*	*55.0*

有几个事项需要注意。我们发现 DataFrame 以表格形式显示，和 Excel 电子表格很像，行和列的标签以**粗体**形式显示。此外注意，DataFrame 的行标签根据构建字典所用的两个 Pandas Series 的索引标签创建而成。DataFrame 的列标签来自字典的*键*。另一个注意事项是，列按照字母顺序排序，而不是字典中的顺序。稍后我们将发现，当我们从数据文件中向 DataFrame 加载数据时，不会发生这种情况。最后要注意的是，我们发现该 DataFrame 中出现了一些 `NaN` 值。`NaN` 是指*非数字*，Pandas 通过这种方式表示该行和列索引没有值。例如，如果我们查看 Alice 列，我们发现手表索引的值是 `NaN`。你可以通

值填充。如果要将此数据馈送到机器学习算法中，我们首先需要删掉这些 `NaN` 值。在后面的课程中，我们将学习如何处理 `NaN` 值以及如何清理数据。暂时先将这些值留在我们的 DataFrame 中。

在上述示例中，我们使用具有定义清晰的索引的 Pandas Series 字典创建了 Pandas DataFrame。如果我们不向 Pandas Series 提供索引标签，Pandas 在创建 DataFrame 时将使用数字行索引。我们来看一个示例：

```
# We create a dictionary of Pandas Series without indexes
data = {'Bob' : pd.Series([245, 25, 55]),
        'Alice' : pd.Series([40, 110, 500, 45])}

# We create a DataFrame
df = pd.DataFrame(data)

# We display the DataFrame
df
```

	Alice	*Bob*
0	*40*	*245.0*
1	*110*	*25.0*
2	*500*	*55.0*
3	*45*	*NaN*

可以看出，Pandas DataFrame 的行索引从 0 开始，就像 NumPy ndarray 的索引一样。

现在，和 Pandas Series 一样，我们也可以使用属性从 DataFrame 中提取信息。我们输出 `shopping_carts` DataFrame 中的一些信息

```
# We print some information about shopping_carts
print('shopping_carts has shape:', shopping_carts.shape)
print('shopping_carts has dimension:', shopping_carts.ndim)
```

```
print('The data in shopping_carts is:\n', shopping_carts.values)
print()
print('The row index in shopping_carts is:', shopping_carts.index)
print()
print('The column index in shopping_carts is:', shopping_carts.co
```

> *shopping_carts has shape: (5, 2)*
> *shopping_carts has dimension: 2*
> *shopping_carts has a total of: 10 elements*

> *The data in shopping_carts is:*
> *[[500. 245.]*
> *[40. nan]*
> *[110. nan]*
> *[45. 25.]*
> *[nan 55.]]*

> *The row index in shopping_carts is: Index(['bike', 'book', 'glasses', 'pants', 'watch'], dtype='object')*

> *The column index in shopping_carts is: Index(['Alice', 'Bob'], dtype='object')*

在 `shopping_carts` DataFrame 时，我们将整个字典传递给了 `pd.DataFrame()` 函数。但是，有时候你可能只对一部分数据感兴趣。在 Pandas 中，我们可以通过关键字 `columns` 和 `index` 选择要将哪些数据放入 DataFrame 中。我们来看一些示例：

```
# We Create a DataFrame that only has Bob's data
bob_shopping_cart = pd.DataFrame(items, columns=['Bob'])

# We display bob_shopping_cart
bob_shopping_cart
```

bike	*245*
pants	*25*
watch	*55*

```
# We Create a DataFrame that only has selected items for both Ali
sel_shopping_cart = pd.DataFrame(items, index = ['pants', 'book']

# We display sel_shopping_cart
sel_shopping_cart
```

	Alice	***Bob***
pants	*45*	*25.0*
book	*40*	*NaN*

```
# We Create a DataFrame that only has selected items for Alice
alice_sel_shopping_cart = pd.DataFrame(items, index = ['glasses',

# We display alice_sel_shopping_cart
alice_sel_shopping_cart
```

	Alice
glasses	*110*
bike	*500*

你还可以使用列表（数组）字典手动地创建 DataFrame。流程和之前一样，首先创建一个字典，然后将该字典传递给 `pd.DataFrame()` 函数。但是在这种情况下，字典中的所有

```
# We create a dictionary of lists (arrays)
data = {'Integers' : [1,2,3],
        'Floats' : [4.5, 8.2, 9.6]}

# We create a DataFrame
df = pd.DataFrame(data)

# We display the DataFrame
df
```

	Floats	*Integers*
0	*4.5*	*1*
1	*8.2*	*2*
2	*9.6*	*3*

注意，因为我们创建的 `data` 字典没有标签索引，因此 Pandas 在创建 DataFrame 时自动使用数字行索引。但是，我们可以通过在 `pd.DataFrame()` 函数中使用关键字 `index`，为行索引添加标签。我们来看一个示例：

```
# We create a dictionary of lists (arrays)
data = {'Integers' : [1,2,3],
        'Floats' : [4.5, 8.2, 9.6]}

# We create a DataFrame and provide the row index
df = pd.DataFrame(data, index = ['label 1', 'label 2', 'label 3'])

# We display the DataFrame
df
```

label 1	*4.5*	*1*
label 2	*8.2*	*2*
label 3	*9.6*	*3*

手动创建 Pandas DataFrame 的最后一种方式是使用 Python 字典列表。流程和之前一样，我们先创建字典，然后将该字典传递给 `pd.DataFrame()` 函数。

```
# We create a list of Python dictionaries
items2 = [{'bikes': 20, 'pants': 30, 'watches': 35},
          {'watches': 10, 'glasses': 50, 'bikes': 15, 'pants':5}]

# We create a DataFrame
store_items = pd.DataFrame(items2)

# We display the DataFrame
store_items
```

	bikes	*glasses*	*pants*	*watches*
0	*20*	*NaN*	*30*	*35*
1	*15*	*50.0*	*5*	*10*

同样注意，因为我们创建的 `items2` 字典没有标签索引，因此 Pandas 在创建 DataFrame 时自动使用数字行索引。和之前一样，我们可以通过在 `pd.DataFrame()` 函数中使用关键字 `index`，为行索引添加标签。假设我们将使用该 DataFrame 存储某个商店的商品库存数量。我们将行索引的标签设为 **store 1** 和 **store 2**。

```
# We create a list of Python dictionaries
items2 = [{'bikes': 20, 'pants': 30, 'watches': 35},
          {'watches': 10, 'glasses': 50, 'bikes': 15, 'pants':5}]
```

```
# We display the DataFrame
store_items
```

	bikes	glasses	pants	watches
store 1	20	NaN	30	35
store 2	15	50.0	5	10

Search or ask questions in Knowledge.

Ask peers or mentors for help in Student Hub.

下一项