

访问和删除 ndarray 中的元素及向其中插入元素

The screenshot shows a Jupyter Notebook window titled 'NumPy'. The browser address bar indicates the URL is 'localhost:8888/notebooks/numpy_sc/NumPy.ipynb'. The notebook interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar with various icons. The code area contains two input cells: the first cell has the code 'import numpy as np', and the second cell is empty. A blue circular cursor is positioned over the second cell. At the bottom of the notebook, there is a text overlay that reads 'Now that you know how to create a variety of NumPy arrays,'. The status bar at the very bottom shows a timer '00:00 / 05:37' and a zoom level '1x' followed by 'CC'.

你已经知道如何创建各种 ndarray，现在将学习 NumPy 使我们如何有效地操纵 ndarray 中的数据。NumPy ndarray 是可变的，意味着 ndarray 中的元素在 ndarray 创建之后可以更改。NumPy ndarray 还可以切片，因此可以通过多种方式拆分 ndarray。例如，我们可以从 ndarray 中获取想要的任何子集。通常，在机器学习中，你需要使用切片拆分数据，例如将数据集拆分为训练集、交叉验证集和测试集。

我们首先将了解如何通过索引访问或修改 ndarray 中的元素。可以在方括号 [] 中添加索引来访问元素。在 NumPy 中，你可以使用正索引和负索引访问 ndarray 中的元素。正索引表示从数组的开头访问元素，负索引表示从数组的末尾访问元素。我们来看看如何访问秩为 1 的 ndarray 中的元素：

```
# We print x
print()
print('x = ', x)
print()

# Let's access some elements with positive indices
print('This is First Element in x:', x[0])
print('This is Second Element in x:', x[1])
print('This is Fifth (Last) Element in x:', x[4])
print()

# Let's access the same elements with negative indices
print('This is First Element in x:', x[-5])
print('This is Second Element in x:', x[-4])
print('This is Fifth (Last) Element in x:', x[-1])
```

```
x = [1 2 3 4 5]
```

```
This is First Element in x: 1
This is Second Element in x: 2
This is Fifth (Last) Element in x: 5
```

```
This is First Element in x: 1
This is Second Element in x: 2
This is Fifth (Last) Element in x: 5
```

注意，要访问 ndarray 中的第一个元素，我们需要使用索引 0，而不是 1。此外注意，可以同时使用正索引和负索引访问同一个元素。正如之前提到的，正索引用于从数组的开头访问元素，负索引用于从数组的末尾访问元素。

现在我们看看如何更改秩为 1 的 ndarray 中的元素。方法是访问要更改的元素，然后使用 `=` 符号分配新的值：



```
# We print the original x
print()
print('Original:\n x = ', x)
print()

# We change the fourth element in x from 4 to 20
x[3] = 20

# We print x after it was modified
print('Modified:\n x = ', x)
```

Original: x = [1 2 3 4 5]

Modified: x = [1 2 3 20 5]

同样，我们可以访问和修改秩为 2 的 ndarray 中的特定元素。要访问秩为 2 的 ndarray 中的元素，我们需要提供两个索引，格式为 `[row, column]`。我们来看一些示例：

```
# We create a 3 x 3 rank 2 ndarray that contains integers from 1 to 9
X = np.array([[1,2,3],[4,5,6],[7,8,9]])

# We print X
print()
print('X = \n', X)
print()

# Let's access some elements in X
print('This is (0,0) Element in X:', X[0,0])
print('This is (0,1) Element in X:', X[0,1])
print('This is (2,2) Element in X:', X[2,2])
```

X =
[[1 2 3]
 [4 5 6]
 [7 8 9]]

This is (0,0) Element in X: 1
This is (0,1) Element in X: 2
This is (2,2) Element in X: 9



可以像针对秩为 1 的 ndarray 一样修改秩为 2 的 ndarray 中的元素。我们来看一个示例：

```
# We create a 3 x 3 rank 2 ndarray that contains integers from 1 to 9
X = np.array([[1,2,3],[4,5,6],[7,8,9]])

# We print the original x
print()
print('Original:\n X = \n', X)
print()

# We change the (0,0) element in X from 1 to 20
X[0,0] = 20

# We print X after it was modified
print('Modified:\n X = \n', X)
```

Original:

X =

[[1 2 3]

[4 5 6]

[7 8 9]]

Modified:

X =

[[20 2 3]

[4 5 6]

[7 8 9]]

现在看看如何向 ndarray 中添加元素及删除其中的元素。我们可以使用

`np.delete(ndarray, elements, axis)` 函数删除元素。此函数会沿着指定的轴从给定 ndarray 中删除给定的元素列表。对于秩为 1 的 ndarray，不需要使用关键字 `axis`。对于秩为 2 的 ndarray，`axis = 0` 表示选择行，`axis = 1` 表示选择列。我们来看一些示例：



```
# We create a rank 2 ndarray
Y = np.array([[1,2,3],[4,5,6],[7,8,9]])

# We print x
print()
print('Original x = ', x)

# We delete the first and last element of x
x = np.delete(x, [0,4])

# We print x with the first and last element deleted
print()
print('Modified x = ', x)

# We print Y
print()
print('Original Y = \n', Y)

# We delete the first row of y
w = np.delete(Y, 0, axis=0)

# We delete the first and last column of y
v = np.delete(Y, [0,2], axis=1)

# We print w
print()
print('w = \n', w)

# We print v
print()
print('v = \n', v)
```

Original x = [1 2 3 4 5]

Modified x = [2 3 4]

Original Y =

[[1 2 3]

[4 5 6]

[7 8 9]]

w =

[[4 5 6]

[7 8 9]]



```
[[2]
```

```
[5]
```

```
[8]]
```

现在来看看如何向 ndarray 中附加值。我们可以使用

`np.append(ndarray, elements, axis)` 函数向 ndarray 中附加值。该函数会将给定的元素列表沿着指定的轴附加到 ndarray 中。我们来看一些示例：

```
# We create a rank 2 ndarray
Y = np.array([[1,2,3],[4,5,6]])

# We print x
print()
print('Original x = ', x)

# We append the integer 6 to x
x = np.append(x, 6)

# We print x
print()
print('x = ', x)

# We append the integer 7 and 8 to x
x = np.append(x, [7,8])

# We print x
print()
print('x = ', x)

# We print Y
print()
print('Original Y = \n', Y)

# We append a new row containing 7,8,9 to y
v = np.append(Y, [[7,8,9]], axis=0)

# We append a new column containing 9 and 10 to y
q = np.append(Y, [[9],[10]], axis=1)

# We print v
print()
print('v = \n', v)

# We print q
print()
print('q = \n', q)
```

Original x = [1 2 3 4 5]

x = [1 2 3 4 5 6]

x = [1 2 3 4 5 6 7 8]



```
[4 5 6]]
```

```
v =
```

```
[[1 2 3]
```

```
 [4 5 6]
```

```
 [7 8 9]]
```

```
q =
```

```
[[ 1 2 3 9]
```

```
 [ 4 5 6 10]]
```

注意，当我们将行或列附加到秩为 2 的 ndarray 中时，行或列的形状必须正确，以与秩为 2 的 ndarray 的形状相符。

现在来看看如何向 ndarray 中插入值。我们可以使用

`np.insert(ndarray, index, elements, axis)` 函数向 ndarray 中插入值。此函数会将给定的 **元素** 列表沿着指定的 **轴** 插入到 ndarray 中，并放在给定的 **索引** 前面。我们来看一些示例：


```
# We create a rank 2 ndarray
Y = np.array([[1,2,3],[7,8,9]])

# We print x
print()
print('Original x = ', x)

# We insert the integer 3 and 4 between 2 and 5 in x.
x = np.insert(x,2,[3,4])

# We print x with the inserted elements
print()
print('x = ', x)

# We print Y
print()
print('Original Y = \n', Y)

# We insert a row between the first and last row of y
w = np.insert(Y,1,[4,5,6],axis=0)

# We insert a column full of 5s between the first and second column of y
v = np.insert(Y,1,5, axis=1)

# We print w
print()
print('w = \n', w)

# We print v
print()
print('v = \n', v)
```

Original x = [1 2 5 6 7]

x = [1 2 3 4 5 6 7]

Original Y =

[[1 2 3]

[7 8 9]]

w =

[[1 2 3]

[4 5 6]

[7 8 9]]

```
[[1 5 2 3]
 [7 5 8 9]]
```

NumPy 还允许我们将 ndarray 上下堆叠起来，或者左右堆叠。可以使用 `np.vstack()` 函数进行垂直堆叠，或使用 `np.hstack()` 函数进行水平堆叠。请务必注意，为了堆叠 ndarray，ndarray 的形状必须相符。我们来看一些示例：

```
# We create a rank 1 ndarray
x = np.array([1,2])

# We create a rank 2 ndarray
Y = np.array([[3,4],[5,6]])

# We print x
print()
print('x = ', x)

# We print Y
print()
print('Y = \n', Y)

# We stack x on top of Y
z = np.vstack((x,Y))

# We stack x on the right of Y. We need to reshape x in order to stack it on
the right of Y.
w = np.hstack((Y,x.reshape(2,1)))

# We print z
print()
print('z = \n', z)

# We print w
print()
print('w = \n', w)
```

```
x = [1 2]
```

```
Y =
[[3 4]
 [5 6]]
```



```
[3 4]
```

```
[5 6]]
```

```
w =
```

```
[[3 4 1]
```

```
[5 6 2]]
```

下一项