< 更多IPython资源 | 目录 | 理解Python中的数据类型 >

Introduction to NumPy

NumPy 介绍

> This chapter, along with chapter 3, outlines techniques for effectively loading, storing, and manipulating in-memory data in Python. The topic is very broad: datasets can come from a wide range of sources and a wide range of formats, including be collections of documents, collections of images, collections of sound clips, collections of numerical measurements, or nearly anything else. Despite this apparent heterogeneity, it will help us to think of all data fundamentally as arrays of numbers.

下面我们将开启新的一章，本章连同第三章一起，会介绍和讨论高效的装载，存储和处理Python中内存数据的技巧。这个主题非常广泛：数据集可能来自非常不同的来源和非常不同的格式，包括文档的集合，图像的集合，声音片段的集合，数值测量的集合，甚至其他任何东西的集合。尽管数据集有着超出想象的异质性，我们还是可以将所有的数据抽象成为数值组成的数组。

> For example, images–particularly digital images–can be thought of as simply two-dimensional arrays of numbers representing pixel brightness across the area. Sound clips can be thought of as one-dimensional arrays of intensity versus time. Text can be converted in various ways into numerical representations, perhaps binary digits representing the frequency of certain words or pairs of words. No matter what the data are, the first step in making it analyzable will be to transform them into arrays of numbers. (We will discuss some specific examples of this process later in Feature Engineering)

例如图像，这里我们特指数字图像，可以被认为是简单的二维数组，包含着代表这区域内每个像素亮度的数值。声音片段可以被认为是一维的数组，包含着时间范围内声音强度的数值。文本可以使用各种方法转换成为数值方式表示，比方说使用二进制数字表示某个单词或短语的出现频率。无论数据是哪种类型，我们对它们进行处理的时候，第一步总是设计将它们转换为数值。（参见特征工程）

> For this reason, efficient storage and manipulation of numerical arrays is absolutely fundamental to the process of doing data science. We'll now take a look at the specialized tools that Python has for handling such numerical arrays: the NumPy package, and the Pandas package (discussed in Chapter 3).

因此，有效的存储和处理数值数组对于数据科学来说是最根本的能力。我们接下来会讨论Python中具备这样强大功能的特殊工具：NumPy和Pandas（将在第三章讨论）。

> This chapter will cover NumPy in detail. NumPy (short for *Numerical Python*) provides an efficient interface to store and operate on dense data buffers. In some ways, NumPy arrays are like Python's built-in `list` type, but NumPy arrays provide much more efficient storage and data operations as the arrays grow larger in size. NumPy arrays form the core of nearly the entire ecosystem of data science tools in Python, so time spent learning to use effectively will be valuable no matter what aspect of data science interests you.

本章会详细介绍NumPy（*Numerical Python* 数值Python的缩写），它提供了强大的接口供我们存储和操作非稀疏数据集合。在某些情况下，NumPy的数组表现得就像Python内建的 `列表` ，但是NumPy数组在存储和操作大量数据集合的时候提供了有效得多的功能和性能。NumPy数组是Python的数据科学领域工具链的核心，很多其他的工具都是在它的基础上构建的，因此无论你感兴趣的是数据科学的哪个领域，NumPy都值得你花时间进行钻研。

> If you followed the advice outlined in the Preface and installed the Anaconda stack, you already have NumPy installed and ready to go. If you're more the do-it-yourself type, you can go to http://www.numpy.org/ and follow the installation instructions found there. Once you do, you can import NumPy and double-check the version:

如果你遵从这本书序言的内容安装的Anaconda，那么NumPy已经自动安装好了，你可以继续往下阅读。如果你喜欢DIY，你可以到NumPy官网，然后按照提示自行安装。当你完成之后，你就可以在你的脚本中载入NumPy模块了，然后输出NumPy的版本号验证安装结果：

```
In [1]: import numpy
        numpy.__version__
Out[1]: '1.16.4'
```

> For the pieces of the package discussed here, I'd recommend NumPy version 1.8 or later. By convention, you'll find that most people in the SciPy/PyData world will import NumPy using `np` as an alias:

对于本书中的例子来说，作者推荐安装NumPy 1.8或以上版本。习惯上，大多数人都会使用 `np` 作为别名来载入NumPy模块：

```
In [2]: import numpy as np
```

> Throughout this chapter, and indeed the rest of the book, you'll find that this is the way we will import and use NumPy.

本章以及本书后续内容，这都是我们载入NumPy模块的标准方式。

Reminder about Built In Documentation

内建帮助和文档

> As you read through this chapter, don't forget that IPython gives you the ability to quickly explore the contents of a package (by using the tab-completion feature), as well as the documentation of various functions (using the `?` character – Refer back to Help and Documentation in IPython).

在你阅读本章的过程中，请不要忘记了IPython提供的内建帮助工具 `?` 以及使用制表符自动补全的功能。（参见：IPython帮助和文档。

> For example, to display all the contents of the numpy namespace, you can type this:

例如，要查看numpy模块中的所有内容（属性和方法），你可以输入：

```
In [3]: np.<TAB>
```

> And to display NumPy's built-in documentation, you can use this:

如果想查看numpy的内建文档，你可以输入：

```
In [4]: np?
```

> More detailed documentation, along with tutorials and other resources, can be found at http://www.numpy.org.

需要更加详尽的文档、教程或其他资源，你可以访问NumPy官网。

< 更多IPython资源 | 目录 | 理解Python中的数据类型 >