


Python Data Science Handbook

Essential Tools for Working with Data

Stephen CUI 

March 8, 2023

Chapter 1

Understanding Data Types in Python

1.1 Fixed-Type Arrays in Python

Python offers several different options for storing data in efficient, fixed-type data buffers. The built-in array module (available since Python 3.3) can be used to create dense arrays of a uniform type.

While Python's array object provides efficient storage of array-based data, NumPy adds to this efficient operations on that data.

1.2 Creating Arrays from Python Lists

Remember that unlike Python lists, NumPy arrays can only contain data of the same type. If the types do not match, NumPy will upcast them according to its type promotion rules.

1.3 Creating Arrays from Scratch

Especially for larger arrays, it is more efficient to create arrays from scratch using routines built into NumPy.

1.4 NumPy Standard Data Types

NumPy arrays contain values of a single type, so it is important to have detailed knowledge of those types and their limitations. The standard NumPy data types are listed in [Table 1.1](#). Note that when constructing an array, they can be specified using a string or using the associated NumPy object. NumPy also supports compound data types, which will be covered in ??.

Table 1.1: Standard NumPy data types

Data type	Description
<code>bool_</code>	Boolean (True or False) stored as a byte
<code>int_</code>	Default integer type (same as C long; normally either int64 or int32)
<code>intc</code>	Identical to C int (normally int32 or int64)
<code>intp</code>	Integer used for indexing (same as C <code>ssize_t</code> ; normally either int32 or int64)
<code>int8</code>	Byte (−128 to 127)
<code>int16</code>	Integer (−32768 to 32767)
<code>int32</code>	Integer (−2147483648 to 2147483647)
<code>int64</code>	Integer (−9223372036854775808 to 9223372036854775807)
<code>uint8</code>	Unsigned integer (0 to 255)
<code>uint16</code>	Unsigned integer (0 to 65535)
<code>uint32</code>	Unsigned integer (0 to 4294967295)
<code>uint64</code>	Unsigned integer (0 to 18446744073709551615)
<code>float_</code>	Shorthand for float64
<code>float16</code>	Half-precision float: sign bit, 5 bits exponent, 10 bits mantissa
<code>float32</code>	Single-precision float: sign bit, 8 bits exponent, 23 bits mantissa
<code>float64</code>	Double-precision float: sign bit, 11 bits exponent, 52 bits mantissa
<code>complex_</code>	Shorthand for complex128
<code>complex64</code>	Complex number, represented by two 32-bit floats
<code>complex128</code>	Complex number, represented by two 64-bit floats