**NOTES**

**Why do we use python?**

It can be used to write quick-and-dirty small programs,or scripts. I don’t like the term “scripting language” as it carries a connotation that theycannot be used for building mission-critical software. For data analysis and interactive, exploratory computing and data visualization, Pythonwill inevitably draw comparisons with the many other domain-specific open sourceand commercial programming languages and tools in wide use, such as R, MATLAB,SAS, Stata, and others. Python is easy to use, powerful, and versatile, making it a great choice for beginners and experts alike. Python's readability makes it a great first programming language — it allows you to think like a programmer and not waste time with confusing syntax. Python is a suitable language not only for doingresearch and prototyping but also building the production systems, too.

**Why not Python?**

As Python is an interpreted programming language, in general most Python code willrun substantially slower than code written in a compiled language like Java or C++. Asprogrammer time is typically more valuable than CPU time, many are happy to makethis tradeoff.Python is not an ideal language for highly concurrent, multithreaded applications, par-ticularly applications with many CPU-bound threads. The reason for this is that it haswhat is known as the global interpreter lock (GIL), a mechanism which prevents theinterpreter from executing more than one Python bytecode instruction at a time. Thetechnical reasons for why the GIL exists are beyond the scope of this book, but as ofthis writing it does not seem likely that the GIL will disappear anytime soon.

**NumPy:**

NumPy, short for Numerical Python, is the foundational package for scientific com-puting in Python. The majority of this book will be based on NumPy and libraries builton top of NumPy. It provides, among other things:• A fast and efficient multidimensional array object ndarray• Functions for performing element-wise computations with arrays or mathematicaloperations between arrays• Tools for reading and writing array-based data sets to disk• Linear algebra operations, Fourier transform, and random number generation• Tools for integrating connecting C, C++, and Fortran code to Python

**Pandas**:

Pandas provides rich data structures and functions designed to make working withstructured data fast, easy, and expressive. It is, as you will see, one of the critical in-gredients enabling Python to be a powerful and productive data analysis environment.The primary object in pandas that will be used in this book is the DataFrame, a two-dimensional tabular, column-oriented data structure with both row and column labels.pandas combines the high performance array-computing features of NumPy with theflexible data manipulation capabilities of spreadsheets and relational databases (suchas SQL). It provides sophisticated indexing functionality to make it easy to reshape,slice and dice, perform aggregations, and select subsets of data. pandas is the primarytool that we will use in this book.

**Matplotlib**:

Matplotlib is the most popular Python library for producing plots and other 2D datavisualizations. It was originally created by John D. Hunter (JDH) and is now maintainedby a large team of developers. It is well-suited for creating plots suitable for publication.It integrates well with IPython (see below), thus providing a comfortable interactiveenvironment for plotting and exploring data. The plots are also interactive; you canzoom in on a section of the plot and pan around the plot using the toolbar in the plotwindow

**SciPy**:

SciPySciPy is a collection of packages addressing a number of different standard problemdomains in scientific computing. Here is a sampling of the packages included:•scipy.integrate: numerical integration routines and differential equation solvers•scipy.linalg: linear algebra routines and matrix decompositions extending be-yond those provided in numpy.linalg.•scipy.optimize: function optimizers (minimizers) and root finding algorithms•scipy.signal: signal processing tools•scipy.sparse: sparse matrices and sparse linear system solvers•scipy.special: wrapper around SPECFUN, a Fortran library implementing manycommon mathematical functions, such as the gamma function•scipy.stats: standard continuous and discrete probability distributions (densityfunctions, samplers, continuous distribution functions), various statistical tests,and more descriptive statistics•scipy.weave: tool for using inline C++ code to accelerate array computation.

**scikit-learn:**

Since the project’s inception in 2010, scikit-learn has become the premier general-purpose machine learning toolkit for Python programmers. In just seven years, it hashad over 1,500 contributors from around the world. It includes submodules for suchmodels as:•Classification: SVM, nearest neighbors, random forest, logistic regression, etc.•Regression: Lasso, ridge regression, etc.•Clustering: k-means, spectral clustering, etc.•Dimensionality reduction: PCA, feature selection, matrix factorization, etc.•Model selection: Grid search, cross-validation, metrics•Preprocessing: Feature extraction, normalization

**Munge/munging/wrangling**: Describes the overall process of manipulating unstructured and/or messy datainto a structured or clean form. The word has snuck its way into the jargon ofmany modern-day data hackers. “Munge” rhymes with “grunge.”

**Pseudocode** :A description of an algorithm or process that takes a code-like form while likelynot being actual valid source code.

Syntactic sugar Programming syntax that does not add new features, but makes something moreconvenient or easier to type

**How to import Libraries** :

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt import urllib

**How to read txt file:** path = 'ch02/usagov\_bitly\_data2012-03-16-1331923249.txt' and print open(path).readline()

**Basic python commands :** print "Hello, Python!"

**Writing comments** : #!/usr/bin/python # First comment print "Hello, Python!" # second comment

**Python string commands**: str = 'Hello World!'

print str # Prints complete string

print str[0] # Prints first character of the string

print str[2:5] # Prints characters starting from 3rd to 5th

print str[2:] # Prints string starting from 3rd character

print str \* 2 # Prints string two times

print str + "TEST" # Prints concatenated strin

Interacting with the outside world Reading and writing with a variety of file formats and databases.Preparation

Cleaning, munging, combining, normalizing, reshaping, slicing and dicing, andt ransforming data for analysis.

Transformation :A pplying mathematical and statistical operations to groups of data sets to derivenew data sets. For example, aggregating a large table by group variables. Modeling and computation

Connecting your data to statistical models, machine learning algorithms, or othercomputational toolsPresentationCreating interactive or static graphical visualizations or textual summaries

The %run CommandYou can run any file as a Python program inside the environment of your IPythonsession using the %run command. Suppose you had the following simple script storedin ipython\_script\_test.py:def f(x, y, z):return (x + y) / za = 5b = 6c = 7.5result = f(a, b, c)You can execute this by passing the filename to %run:In [14]: %run ipython\_script\_test.py

**About Magic Commands:**

Python’s special commands (which are not built into Python itself ) are known as“magic” commands. These are designed to facilitate common tasks and enable you toeasily control the behavior of the IPython system. A magic command is any com‐mand prefixed by the percent symbol %. For example, you can check the executiontime of any Python statement, such as a matrix multiplication, using the %timeitmagic function (which will be discussed in more detail later):In [20]: a = np.random.randn(100, 100)In [20]: %timeit np.dot(a, a)10000 loops, best of 3: 20.9 μs per loopMagic commands can be viewed as command-line programs to be run within theIPython system. Many of them have additional “command-line” options, which canall be viewed

Matplotlib IntegrationOne reason for IPython’s popularity in analytical computing is that it integrates wellwith data visualization and other user interface libraries like matplotlib. Don’t worryif you have never used matplotlib before; it will be discussed in more detail later in this book. The %matplotlib magic function configures its integration with the IPy‐thon shell or Jupyter notebook. This is important, as otherwise plots you create willeither not appear (notebook) or take control of the session until closed (shell).

**Imports :**

In Python a module is simply a file with the .py extension containing Python code

Most objects in Python, such as lists, dicts, NumPy arrays, and most user-definedtypes (classes), are mutable. This means that the object or values that they contain canbe modified

The primary Python types for numbers are int and float. An int can store arbitrar‐ily large numbers. Floating-point numbers are represented with the Python float type. Under the hoodeach one is a double-precision (64-bit) value. They can also be expressed with scien‐tific notation.

**Strings :** Many people use Python for its powerful and flexible built-in string processing capa‐bilities. You can write string literals using either single quotes ' or double quotes ":a = 'one way of writing a string'b = "another way"For multiline strings with line breaks, you can use triple quotes, either ''' or """

Bytes and UnicodeIn modern Python (i.e., Python 3.0 and up), Unicode has become the first-class stringtype to enable more consistent handling of ASCII and non-ASCII text. In older ver‐sions of Python, strings were all bytes without any explicit Unicode encoding. Youcould convert to Unicode assuming you knew the character encoding.

**Booleans:**

The two boolean values in Python are written as True and False. Comparisons andother conditional expressions evaluate to either True or False. Boolean values arecombined with the and and or keywords.

**Type casting:** The str, bool, int, and float types are also functions that can be used to cast valuesto those types

**Dates and times :** The built-in Python datetime module provides datetime, date, and time types. Thedatetime type, as you may imagine, combines the information stored in date andtime and is the most commonly used

while loopsA while loop specifies a condition and a block of code that is to be executed until thecondition evaluates to False or the loop is explicitly ended with break.

passpass is the “no-op” statement in Python. It can be used in blocks where no action is tobe taken (or as a placeholder for code not yet implemented); it is only requiredbecause Python uses whitespace to delimit blocks

**range** :

The range function returns an iterator that yields a sequence of evenly spacedintegers