ОГЛАВЛЕНИЕ

Предисловие **12**
Графические выделения 12

Глава 1. Предварительные сведения **13**
О чем эта книга? 13
Почему именно Python? 14
Python как клей 14
Решение проблемы «двух языков» 15
Недостатки Python 15
Необходимые библиотеки для Python 16
NumPy 16
pandas 16
matplotlib 17
IPython 17
SciPy 18
Установка и настройка 18
Windows 19
Apple OS X 21
GNU/Linux 22
Python 2 и Python 3 23
Интегрированные среды разработки (IDE) 24
Сообщество и конференции 24
Структура книги 25
Примеры кода 25
Данные для примеров 25
Соглашения об импорте 25
Жаргон 26
Благодарности 26

Глава 2. Первые примеры **28**
Набор данных 1.usa.gov с сайта bit.ly 28
Подсчет часовых поясов на чистом Python 30
Подсчет часовых поясов с помощью pandas 32
Набор данных MovieLens 1M 38
Измерение несогласия в оценках 42
Имена, которые давали детям в США за период с 1880 по 2010 год 43
Анализ тенденций в выборе имен 48
Выводы и перспективы 56

Глава 3. IPython: интерактивные вычисления и среда разработки ... 57

Основы IPython ... 58
Завершение по нажатию клавиши Tab ... 59
Интроспекция ... 60
Команда %run ... 61
Исполнение кода из буфера обмена ... 63
Комбинации клавиш ... 64
Исключения и обратная трассировка ... 65
Магические команды ... 66
Графическая консоль на базе Qt ... 68
Интеграция с matplotlib и режим pylab ... 68
История команд ... 70
Поиск в истории команд и повторное выполнение ... 70
Входные и выходные переменные ... 71
Протоколирование ввода-вывода ... 72
Взаимодействие с операционной системой ... 73
Команды оболочки и псевдонимы ... 73
Система закладок на каталоги ... 75
Средства разработки программ ... 75
Интерактивный отладчик ... 75
Хронометраж программы: %time и %timeit ... 80
Простейшее профилирование: %prun и %run -p ... 82
Построчное профилирование функции ... 83
HTML-блокнот в IPython ... 86
Советы по продуктивной разработке кода с использованием IPython ... 86
Перезагрузка зависимостей модуля ... 87
Советы по проектированию программ ... 88
Дополнительные возможности IPython ... 90
Делайте классы дружественными к IPython ... 90
Профили и конфигурирование ... 90
Благодарности ... 92

Глава 4. Основы NumPy: массивы и векторные вычисления ... 93

NumPy ndarray: объект многомерного массива ... 94
Создание ndarray ... 95
Тип данных для ndarray ... 97
Операции между массивами и скалярами ... 100
Индексирование и вырезание ... 100
Булево индексирование ... 104
Прихотливое индексирование ... 107
Транспонирование массивов и перестановка осей ... 108
Универсальные функции: быстрые поэлементные операции над массивами ... 109
Обработка данных с применением массивов ... 112
Запись логических условий в виде операций с массивами ... 113

Математические и статистические операции 115
Методы булевых массивов 116
Сортировка 117
Устранение дубликатов и другие теоретико-множественные операции....118
Файловый ввод-вывод массивов 119
Хранение массивов на диске в двоичном формате....119
Сохранение и загрузка текстовых файлов 120
Линейная алгебра 121
Генерация случайных чисел 122
Пример: случайное блуждание....123
Моделирование сразу нескольких случайных блужданий 125

Глава 5. Первое знакомство с pandas 127

Введение в структуры данных pandas 128
Объект Series 128
Объект DataFrame 131
Индексные объекты....137
Базовая функциональность.... 139
Переиндексация 139
Удаление элементов из оси....142
Доступ по индексу, выборка и фильтрация 143
Арифметические операции и выравнивание данных 146
Применение функций и отображение....150
Сортировка и ранжирование 151
Индексы по осям с повторяющимися значениями....154
Редукция и вычисление описательных статистик 155
Корреляция и ковариация 158
Уникальные значения, счетчики значений и членство 160
Обработка отсутствующих данных 162
Фильтрация отсутствующих данных 163
Иерархическое индексирование.... 166
Уровни переупорядочения и сортировки....169
Сводная статистика по уровню....170
Работа со столбцами DataFrame....170
Другие возможности pandas.... 172
Доступ по целочисленному индексу 172
Структура данных Panel....173

Глава 6. Чтение и запись данных, форматы файлов 175

Чтение и запись данных в текстовом формате 175
Чтение текстовых файлов порциями 181
Вывод данных в текстовом формате....182
Ручная обработка данных в формате с разделителями....184
Данные в формате JSON 186
XML и HTML: разбор веб-страниц....188
Разбор XML с помощью lxml.objectify....190
Двоичные форматы данных....192

Взаимодействие с HTML и Web API 194
Взаимодействие с базами данных 196
Чтение и сохранение данных в MongoDB 198

Глава 7. Переформатирование данных: очистка, преобразование, слияние, изменение формы 199

Комбинирование и слияние наборов данных 199
Слияние объектов DataFrame как в базах данных 200
Слияние по индексу 204
Конкатенация вдоль оси 207
Комбинирование перекрывающихся данных 211
Изменение формы и поворот 212
Изменение формы с помощью иерархического индексирования 213
Поворот из «длинного» в «широкий» формат 215
Преобразование данных 217
Устранение дубликатов 217
Преобразование данных с помощью функции или отображения 218
Замена значений 220
Переименование индексов осей 221
Дискретизация и раскладывание 222
Обнаружение и фильтрация выбросов 224
Перестановки и случайная выборка 226
Вычисление индикаторных переменных 227
Манипуляции со строками 229
Методы строковых объектов 230
Регулярные выражения 232
Векторные строковые функции в pandas 235
Пример: база данных о продуктах питания министерства сельского хозяйства США 237

Глава 8. Построение графиков и визуализация 244

Краткое введение в API библиотеки matplotlib 245
Рисунки и подграфики 246
Цвета, маркеры и стили линий 249
Риски, метки и надписи 251
Аннотации и рисование в подграфике 254
Сохранение графиков в файле 256
Конфигурирование matplotlib 257
Функции построения графиков в pandas 258
Линейные графики 258
Столбчатые диаграммы 260
Гистограммы и графики плотности 264
Диаграммы рассеяния 266
Нанесение данных на карту: визуализация данных о землетрясении на Гаити 267
Инструментальная экосистема визуализации для Python 273

Chaco 274
mayavi 274
Прочие пакеты 275
Будущее средств визуализации 275

Глава 9. Агрегирование данных и групповые операции 276

Механизм GroupBy 277
Обход групп 280
Выборка столбца или подмножества столбцов 281
Группировка с помощью словарей и объектов Series 282
Группировка с помощью функций 284
Группировка по уровням индекса 284

Агрегирование данных 285
Применение функций, зависящих от столбца, и нескольких функций 287
Возврат агрегированных данных в «неиндексированном» виде 289

Групповые операции и преобразования 290
Метод apply: часть общего принципа разделения–применения–объединения 292
Квантильный и интервальный анализ 294
Пример: подстановка зависящих от группы значений вместо отсутствующих 296
Пример: случайная выборка и перестановка 297
Пример: групповое взвешенное среднее и корреляция 299
Пример: групповая линейная регрессия 301

Сводные таблицы и кросс-табуляция 302
Таблицы сопряженности 304
Пример: база данных федеральной избирательной комиссии за 2012 год 305
Статистика пожертвований по роду занятий и месту работы 308
Распределение суммы пожертвований по интервалам 311
Статистика пожертвований по штатам 313

Глава 10. Временные ряды 316

Типы данных и инструменты, относящиеся к дате и времени 317
Преобразование между строкой и datetime 318

Основы работы с временными рядами 321
Индексирование, выборка, подмножества 322
Временные ряды с неуникальными индексами 324

Диапазоны дат, частоты и сдвиг 325
Генерация диапазонов дат 325
Частоты и смещения дат 326
Сдвиг данных (с опережением и с запаздыванием) 329

Часовые пояса 331
Локализация и преобразование 332
Операции над объектами Timestamp с учетом часового пояса 333
Операции между датами из разных часовых поясов 334
Периоды и арифметика периодов 335
Преобразование частоты периода 336

Квартальная частота периода 337
Преобразование временных меток в периоды и обратно 339
Создание PeriodIndex из массивов 340
Передискретизация и преобразование частоты 341
Понижающая передискретизация 342
Повышающая передискретизация и интерполяция 345
Передискретизация периодов 346
Графики временных рядов 348
Скользящие оконные функции 350
Экспоненциально взвешенные функции 353
Бинарные скользящие оконные функции 353
Скользящие оконные функции, определенные пользователем 355
Замечания о быстродействии и потреблении памяти 356

Глава 11. Финансовые и экономические приложения 358
О переформатировании данных 358
Временные ряды и выравнивание срезов 358
Операции над временными рядами с различной частотой 361
Время суток и выборка данных «по состоянию на» 364
Сращивание источников данных 366
Индексы доходности и кумулятивная доходность 368
Групповые преобразования и анализ 370
Оценка воздействия групповых факторов 372
Децильный и квартильный анализ 373
Другие примеры приложений 375
Стохастический граничный анализ 375
Роллинг фьючерсных контрактов 377
Скользящая корреляция и линейная регрессия 380

Глава 12. Дополнительные сведения о библиотеке NumPy ... 383
Иерархия типов данных в NumPy 384
Дополнительные манипуляции с массивами 385
Изменение формы массива 385
Упорядочение элементов массива в C и в Fortran 387
Конкатенация и разбиение массива 388
Повторение элементов: функции tile и repeat 390
Эквиваленты прихотливого индексирования: функции take и put 391
Укладывание 393
Укладывание по другим осям 394
Установка элементов массива с помощью укладывания 397
Дополнительные способы использования универсальных функций 398
Методы экземпляра u-функций 398
Пользовательские u-функции 400
Структурные массивы 401
Вложенные типы данных и многомерные поля 402
Зачем нужны структурные массивы? 403

Манипуляции со структурными массивами: numpy.lib.recfunctions 403
Еще о сортировке 403
Косвенная сортировка: методы argsort и lexsort 405
Альтернативные алгоритмы сортировки 406
Метод numpy.searchsorted: поиск элементов в отсортированном массиве 407
Класс matrix в NumPy 408
Дополнительные сведения о вводе-выводе массивов 410
Файлы, спроецированные на память 410
HDF5 и другие варианты хранения массива 412
Замечание о производительности 412
Важность непрерывной памяти 412
Другие возможности ускорения: Cython, f2py, C 414

Приложение. Основы языка Python 415
Интерпретатор Python 416
Основы 417
Семантика языка 417
Скалярные типы 425
Поток управления 431
Структуры данных и последовательности 437
Список 439
Встроенные функции последовательностей 443
Словарь 445
Множество 448
Списковое, словарное и множественное включение 450
Функции 452
Пространства имен, области видимости и локальные функции 453
Возврат нескольких значений 454
Функции являются объектами 455
Анонимные (лямбда) функции 456
Замыкания: функции, возвращающие функции 457
Расширенный синтаксис вызова с помощью *args и **kwargs 459
Каррирование: частичное фиксирование аргументов 459
Генераторы 460
Генераторные выражения 462
Модуль itertools 462
Файлы и операционная система 463

Предметный указатель 466

ПРЕДИСЛОВИЕ

За последние 10 лет вокруг языка Python образовалась и активно развивается целая экосистема библиотек с открытым исходным кодом. К началу 2011 года у меня сложилось стойкое ощущение, что нехватка централизованных источников учебных материалов по анализу данных и математической статистике становится камнем преткновения на пути молодых программистов на Python, которым такие приложения нужны по работе. Основные проекты, связанные с анализом данных (в особенности NumPy, IPython, matplotlib и pandas), к тому времени стали уже достаточно зрелыми, чтобы про них можно было написать книгу, которая не устареет сразу после выхода. Поэтому я набрался смелости заняться этим делом. Я был бы очень рад, если бы такая книга существовала в 2007 году, когда я приступал к использованию Python для анализа данных. Надеюсь, вам она окажется полезной, и вы сумеете с успехом воспользоваться описываемыми инструментами в собственной работе.

Графические выделения

В книге применяются следующие графические выделения:

Курсив

Новые термины, URL-адреса, адреса электронной почты, имена и расширения имен файлов.

`Моноширинный`

Листинги программ, а также элементы кода в основном тексте: имена переменных и функций, базы данных, типы данных, переменные окружения, предложения и ключевые слова языка.

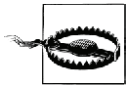
`Моноширинный полужирный`

Команды или иной текст, который должен быть введен пользователем буквально.

`Моноширинный курсив`

Текст, вместо которого следует подставить значения, заданные пользователем или определяемые контекстом.

Так обозначается совет, рекомендация или замечание общего характера.



Так обозначается предупреждение или предостережение.

ГЛАВА 1. Предварительные сведения

О чем эта книга?

Эта книга посвящена вопросам преобразования, обработки, очистки данных и вычислениям на языке Python. Кроме того, она представляет собой современное практическое введение в научные и инженерные расчеты на Python, ориентированное на приложения для обработки больших объемов данных. Это книга о тех частях языка Python и написанных для него библиотек, которые необходимы для эффективного решения широкого круга задач анализа данных. Но в ней вы *не* найдете объяснений аналитических методов с привлечением Python в качестве языка реализации.

Говоря «данные», я имею в виду, прежде всего, *структурированные данные*; это намеренно расплывчатый термин, охватывающий различные часто встречающиеся виды данных, как то:

- многомерные списки (матрицы);
- табличные данные, когда данные в разных столбцах могут иметь разный тип (строки, числа, даты или еще что-то). Сюда относятся данные, которые обычно хранятся в реляционных базах или в файлах с запятой в качестве разделителя;
- данные, представленные в виде нескольких таблиц, связанных между собой по ключевым столбцам (то, что в SQL называется первичными и внешними ключами);
- равноотстоящие и неравноотстоящие временные ряды.

Этот список далеко не полный. Значительную часть наборов данных можно преобразовать к структурированному виду, более подходящему для анализа и моделирования, хотя сразу не всегда очевидно, как это сделать. В тех случаях, когда это не удается, иногда есть возможность извлечь из набора данных структурированное множество признаков. Например, подборку новостных статей можно преобразовать в таблицу частот слов, к которой затем применить анализ эмоциональной окраски.

Большинству пользователей электронных таблиц типа Microsoft Excel, пожалуй, самого широко распространенного средства анализа данных, такие виды данных хорошо знакомы.

Почему именно Python?

Для многих людей (и меня в том числе) Python — язык, в который нельзя не влюбиться. С момента своего появления в 1991 году Python стал одним из самых популярных динамических языков программирования наряду с Perl, Ruby и другими. Относительно недавно Python и Ruby приобрели особую популярность как средства создания веб-сайтов в многочисленных каркасах, например Rails (Ruby) и Django (Python). Такие языки часто называют *скриптовыми*, потому что они используются для быстрого написания небольших программ – *скриптов*. Лично мне термин «скриптовый язык» не нравится, потому что он наводит на мысль, будто для создания ответственного программного обеспечения язык не годится. Из всех интерпретируемых языков Python выделяется большим и активным сообществом *научных расчетов*. Применение Python для этой цели в промышленных и академических кругах значительно расширилось с начала 2000-х годов.

В области анализа данных и интерактивных научно-исследовательских расчетов с визуализацией результатов Python неизбежно приходится сравнивать со многими предметно-ориентированными языками программирования и инструментами – с открытым исходным кодом и коммерческими – такими, как R, MATLAB, SAS, Stata и другими. Сравнительно недавнее появление улучшенных библиотек для Python (прежде всего, pandas) сделало его серьезным конкурентом в решении задач манипулирования данными. В сочетании с достоинствами Python как универсального языка программирования это делает его отличным выбором для создания приложений обработки данных.

Python как клей

Своим успехом в качестве платформы для научных расчетов Python отчасти обязан простоте интеграции с кодом на C, C++ и FORTRAN. Во многих современных вычислительных средах применяется общий набор унаследованных библиотек, написанных на FORTRAN и C, содержащих реализации алгоритмов линейной алгебры, оптимизации, интегрирования, быстрого преобразования Фурье и других. Поэтому многочисленные компании и национальные лаборатории используют Python как «клей» для объединения написанных за 30 лет программ.

Многие программы содержат небольшие участки кода, на выполнение которых уходит большая часть времени, и большие куски «склеивающего кода», который выполняется нечасто. Во многих случаях время выполнения склеивающего кода несущественно, реальную отдачу дает оптимизация узких мест, которые иногда имеет смысл переписать на низкоуровневом языке типа C.

За последние несколько лет на одно из первых мест в области создания быстрых компилируемых расширений Python и организации интерфейса с кодом на C и C++ вышел проект Cython (*http://cython.org*).

Решение проблемы «двух языков»

Во многих организациях принято для научных исследований, создания опытных образцов и проверки новых идей использовать предметно-ориентированные языки типа MATLAB или R, а затем переносить удачные разработки в производственную систему, написанную на Java, C# или C++. Но все чаще люди приходят к выводу, что Python подходит не только для стадий исследования и создания прототипа, но и для построения самих производственных систем. Я полагаю, что компании все чаще будут выбирать этот путь, потому что использование одного и того же набора программных средств учеными и технологами несет несомненные выгоды организации.

Недостатки Python

Python – великолепная среда для создания приложений для научных расчетов и большинства систем общего назначения, но тем не менее существуют задачи, для которых Python не очень подходит.

Поскольку Python – интерпретируемый язык программирования, в общем случае написанный на нем код работает значительно медленнее, чем эквивалентный код на компилируемом языке типа Java или C++. Но поскольку *время программиста* обычно стоит гораздо дороже *времени процессора*, многих такой компромисс устраивает. Однако в приложениях, где задержка должна быть очень мала (например, в торговых системах с большим количеством транзакций), время, потраченное на программирование на низкоуровневом и не обеспечивающем максимальную продуктивность языке типа C++, во имя достижения максимальной производительности, будет потрачено не зря.

Python – не идеальный язык для программирования многопоточных приложений с высокой степенью параллелизма, особенно при наличии многих потоков, активно использующих процессор. Проблема связана с наличием *глобальной блокировки интерпретатора* (GIL) – механизма, который не дает интерпретатору исполнять более одной команды байт-кода Python в каждый момент времени. Объяснение технических причин существования GIL выходит за рамки этой книги, но на данный момент представляется, что GIL вряд ли скоро исчезнет. И хотя во многих приложениях обработки больших объектов данных для обеспечения приемлемого времени приходится организовывать кластер машин, встречаются все же ситуации, когда более желательна однопроцессная многопоточная система.

Я не хочу сказать, что Python вообще непригоден для исполнения многопоточного параллельного кода; просто такой код нельзя выполнять в одном процессе Python. Например, в проекте Cython реализована простая интеграция с OpenMP, написанной на C библиотеке параллельных вычислений, позволяющая распараллеливать циклы и тем самым значительно ускорять работу численных алгоритмов.

Необходимые библиотеки для Python

Для читателей, плохо знакомых с экосистемой Python и используемыми в книге библиотеками, я приведу краткий обзор библиотек.

NumPy

NumPy, сокращение от «Numerical Python», – основной пакет для выполнения научных расчетов на Python. Большая часть этой книги базируется на NumPy и построенных поверх него библиотек. В числе прочего он предоставляет:

- быстрый и эффективный объект многомерного массива *ndarray*;
- функции для выполнения вычислений над элементами одного массива или математических операций с несколькими массивами;
- средства для чтения и записи на диски наборов данных, представленных в виде массивов;
- операции линейной алгебры, преобразование Фурье и генератор случайных чисел;
- средства для интеграции с кодом, написанным на C, C++ или Fortran.

Помимо быстрых средств работы с массивами, одной из основных целей NumPy в части анализа данных является организация контейнера для передачи данных между алгоритмами. Как средство хранения и манипуляции данными массивы NumPy куда эффективнее встроенных в Python структур данных. Кроме того, библиотеки, написанные на низкоуровневом языке типа C или Fortran, могут работать с данными, хранящимися в массиве NumPy, вообще без копирования.

pandas

Библиотека pandas предоставляет структуры данных и функции, призванные сделать работу со структурированными данными простым, быстрым и выразительным делом. Как вы вскоре убедитесь, это один из основных компонентов, превращающих Python в мощный инструмент продуктивного анализа данных. Основной объект pandas, используемый в этой книге, – `DataFrame` – двумерная таблица, в которой строки и столбцы имеют метки:

```
>>> frame
     total_bill   tip     sex      smoker day    time          size
1    16.99        1.01    Female   No     Sun    Dinner        2
2    10.34        1.66    Male     No     Sun    Dinner        3
3    21.01        3.5     Male     No     Sun    Dinner        3
4    23.68        3.31    Male     No     Sun    Dinner        2
5    24.59        3.61    Female   No     Sun    Dinner        4
6    25.29        4.71    Male     No     Sun    Dinner        4
7    8.77         2       Male     No     Sun    Dinner        2
8    26.88        3.12    Male     No     Sun    Dinner        4
9    15.04        1.96    Male     No     Sun    Dinner        2
10   14.78        3.23    Male     No     Sun    Dinner        2
```

Библиотека pandas сочетает высокую производительность средств работы с массивами, присущую NumPy, с гибкими возможностями манипулирования дан-

ными, свойственными электронным таблицам и реляционным базам данных (например, на основе SQL). Она предоставляет развитые средства индексирования, позволяющие без труда изменять форму наборов данных, формировать продольные и поперечные срезы, выполнять агрегирование и выбирать подмножества. В этой книге библиотека pandas будет нашим основным инструментом.

Для разработки финансовых приложений pandas предлагает богатый набор высокопроизводительных средств анализа временных рядов, специально ориентированных на финансовые данные. На самом деле, я изначально проектировал pandas как удобный инструмент анализа именно финансовых данных.

Пользователям языка статистических расчетов R название DataFrame покажется знакомым, потому что оно выбрано по аналогии с объектом `data.frame` в R. Однако они не идентичны: функциональность `data.frame` является собственным подмножеством той, что предлагает `DataFrame`. Хотя эта книга посвящена Python, я время от времени будут проводить сравнения с R, потому что это одна из самых распространенных сред анализа данных с открытым исходным кодом, знакомая многим читателям.

Само название pandas образовано как от *panel data* (панельные данные), применяемого в эконометрике термина для обозначения многомерных структурированных наборов данных, так и от фразы *Python data analysis*.

matplotlib

Библиотека matplotlib – самый популярный в Python инструмент для создания графиков и других способов визуализации двумерных данных. Первоначально она была написана Джоном Д. Хантером (John D. Hunter, JDH), а теперь сопровождается большой группой разработчиков. Она отлично подходит для создания графиков, пригодных для публикации. Интегрирована с IPython (см. ниже), что позволяет организовать удобное интерактивное окружение для визуализации и исследования данных. Графики *интерактивны* – можно увеличить масштаб какого-то участка графика и выполнять панорамирование с помощью панели инструментов в окне графика.

IPython

IPython – компонент стандартного набора инструментов научных расчетов на Python, который связывает все воедино. Он обеспечивает надежную высокопродуктивную среду для интерактивных и исследовательских расчетов. Это оболочка Python с дополнительными возможностями, имеющая целью ускорить написание, тестирование и отладку кода на Python. Особенно она полезна для работы с данными и их визуализации с помощью matplotlib. Я почти всегда использую IPython в собственной работе для прогона, отладки и тестирования кода.

Помимо стандартных средств консольной оболочки, IPython предоставляет:

- HTML-блокнот в духе программы Mathematica для подключения к IPython с помощью веб-браузера (подробнее об этом ниже);

- консоль с графическим интерфейсом пользователя на базе библиотеки Qt, включающую средства построения графиков, многострочный редактор и подсветку синтаксиса;
- инфраструктуру для интерактивных параллельных и распределенных вычислений.

Я посвятил отдельную главу оболочке IPython и способам оптимальной работы с ней. Настоятельно рекомендую использовать ее при чтении этой книги.

SciPy

SciPy – собрание пакетов, предназначенных для решения различных стандартных вычислительных задач. Вот несколько из них:

- `scipy.integrate`: подпрограммы численного интегрирования и решения дифференциальных уравнений;
- `scipy.linalg`: подпрограммы линейной алгебры и разложения матриц, дополняющие те, что включены в `numpy.linalg`;
- `scipy.optimize`: алгоритмы оптимизации функций (нахождения минимумов) и поиска корней;
- `scipy.signal`: средства обработки сигналов;
- `scipy.sparse`: алгоритмы работы с разреженными матрицами и решения разреженных систем линейных уравнений;
- `scipy.special`: обертка вокруг SPECFUN, написанной на Fortran библиотеке, содержащей реализации многих стандартных математических функций, в том числе гамма-функции;
- `scipy.stats`: стандартные непрерывные и дискретные распределения вероятностей (функции плотности вероятности, формирования выборки, функции непрерывного распределения вероятности), различные статистические критерии и дополнительные описательные статистики;
- `scipy.weave`: средство для встраивания кода на C++ с целью ускорения вычислений с массивами.

Совместно NumPy и SciPy образуют достаточно полную замену значительной части системы MATLAB и многочисленных дополнений к ней.

Установка и настройка

Поскольку Python используется в самых разных приложениях, не существует единственно верной процедуры установки Python и необходимых дополнительных пакетов. У многих читателей, скорее всего, нет среды, подходящей для научных применений Python и проработки этой книги, поэтому я приведу подробные инструкции для разных операционных систем. Я рекомендую использовать следующие базовые дистрибутивы Python:

Enthought Python Distribution: ориентированный на научные применения дистрибутив от компании Enthought (*http://www.enthought.com*). Включает EPDFree – бесплатный дистрибутив (содержит NumPy, SciPy, matplotlib, Chaco и IPython), и

EPD Full — полный комплект, содержащий более 100 научных пакетов для разных предметных областей. EPD Full бесплатно поставляется для академического использования, а прочим пользователям предлагается платная годовая подписка.

Python(x,y) (*http://pythonxy.googlecode.com*): бесплатный ориентированный на научные применения дистрибутив для Windows.

В инструкциях ниже подразумевается EPDFree, но вы можете выбрать любой другой подход, все зависит от ваших потребностей. На момент написания этой книги EPD включает версию Python 2.7, хотя в будущем это может измениться. После установки на вашей машине появятся следующие готовые к импорту пакеты:

- базовые пакеты для научных расчетов: NumPy, SciPy, matplotlib и IPython (входят в EPDFree);
- зависимости для IPython Notebook: tornado и pyzmq (также входят в EPDFree);
- pandas (версии 0.8.2 или выше).

По ходу чтения книги вам могут понадобиться также следующие пакеты: statsmodels, PyTables, PyQt (или эквивалентный ему PySide), xlrd, lxml, basemap, pymongo и requests. Они используются в разных примерах. Устанавливать их необязательно, и я рекомендую не торопиться с этим до момента, когда они действительно понадобятся. Например, сборка PyQt или PyTables из исходных кодов в OS X или Linux – довольно муторное дело. А пока нам важно получить минимальную работоспособную среду: EPDFree и pandas.

Сведения обо всех Python-пакетах, ссылки на установщики двоичных версий и другую справочную информацию можно найти в указателе пакетов Python (Python Package Index – PyPI, *http://pypi.python.org*). Заодно это отличный источник для поиска новых Python-пакетов.

Во избежание путаницы я не стану обсуждать более сложные средства управления окружением такие, как pip и virtualenv. В Интернете можно найти немало отличных руководств по ним.

Некоторым пользователям могут быть интересны альтернативные реализации Python, например IronPython, Jython или PyPy. Но для работы с инструментами, представленными в этой книге, в настоящее время необходим стандартный написанный на C интерпретатор Python, известный под названием CPython.

Windows

Для начала установки в Windows скачайте установщик EPDFree с сайта *http://www.enthought.com*; это файл с именем `epd_free-7.3-1-winx86.msi`. Запустите его и согласитесь с предлагаемым по умолчанию установочным каталогом `C:\Python27`. Если в этом каталоге уже был установлен Python, то рекомендую

предварительно удалить его вручную (или с помощью средства «Установка и удаление программ»).

Затем нужно убедиться, что Python прописался в системной переменной PATH и что не возникло конфликтов с ранее установленными версиями Python. Откройте консоль (выберите из меню «Пуск» пункт «Выполнить...» и наберите `cmd.exe`). Попробуйте запустить интерпретатор Python, введя команду `python`. Должно появиться сообщение, в котором указана установленная версия EPDFree:

```
C:\Users\Wes>python
Python 2.7.3 |EPD_free 7.3-1 (32-bit)| (default, Apr 12 2012, 14:30:37) on win32
Type "credits", "demo" or "enthought" for more information.
>>>
```

Если в сообщении указана другая версия EPD или вообще ничего не запускается, то нужно привести в порядок переменные среды Windows. В Windows 7 начните вводить фразу «environment variables» в поле поиска программ и выберите раздел `Edit environment variables for your account`. В Windows XP нужно перейти в `Панель управления > Система > Дополнительно > Переменные среды`. В появляющемся окне найдите переменную `Path`. В ней должны присутствовать следующие два каталога, разделенные точкой с запятой:

```
C:\Python27;C:\Python27\Scripts
```

Если вы ранее устанавливали другие версии Python, удалите относящиеся в Python каталоги из системы и из переменных `Path`. После манипуляций с путями консоль необходимо перезапустить, чтобы изменения вступили в силу.

После того как Python удалось успешно запустить из консоли, необходимо установить pandas. Проще всего для этой цели загрузить подходящий двоичный установщик с сайта *http://pypi.python.org/pypi/pandas*. Для EPDFree это будет файл `pandas-0.9.0.win32-py2.7.exe`. После того как установщик отработает, запустим IPython и проверим, что все установилось правильно, для этого импортируем pandas и построим простой график с помощью matplotlib:

```
C:\Users\Wes>ipython --pylab
Python 2.7.3 |EPD_free 7.3-1 (32-bit)|
Type "copyright", "credits" or "license" for more information.

IPython 0.12.1 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??" for extra details.

Welcome to pylab, a matplotlib-based Python environment [backend: WXAgg].
For more information, type 'help(pylab)'.

In [1]: import pandas

In [2]: plot(arange(10))
```

Если все нормально, то не будет никаких сообщений об ошибках и появится окно с графиком. Можно также проверить, что HTML-блокнот IPython HTML работает правильно:

```
$ ipython notebook --pylab=inline
```

Если в Windows вы обычно используете Internet Explorer, то для работы блокнота IPython, скорее всего, придется установить Mozilla Firefox или Google Chrome.

Дистрибутив EPDFree для Windows содержит только 32-разрядные исполняемые файлы. Если вам необходим 64-разрядный дистрибутив, то проще всего взять EPD Full. Если же вы предпочитаете устанавливать все с нуля и не платить EPD за подписку, то Кристоф Гольке (Christoph Gohlke) из Калифорнийского университета в Ирвайне опубликовал неофициальные двоичные 32- и 64-разрядные установщики для всех используемых в книге пакетов (*http://www.lfd.uci.edu/~gohlke/pythonlibs/*).

Apple OS X

Перед тем как приступать к установке в OS X, необходимо установить Xcode – комплект средств разработки ПО от Apple. Нам понадобится из него gcc – комплект компиляторов для C и C++. Установщик Xcode можно найти на установочном DVD, поставляемом вместе с компьютером, или скачать непосредственно с сайта Apple.

После установки Xcode запустите терминал (Terminal.app), перейдя в меню `Applications > Utilities`. Введите `gcc` и нажмите клавишу **Enter**. Должно появиться такое сообщение:

```
$ gcc
i686-apple-darwin10-gcc-4.2.1: no input files
```

Теперь необходимо установить EPDFree. Скачайте установщик, который должен представлять собой образ диска с именем вида `epd_free-7.3-1-macosx-i386.dmg`. Дважды щелкните мышью по файлу `dmg`-файлу, чтобы смонтировать его, а затем дважды щелкните по `mpkg`-файлу, чтобы запустить установщик.

Установщик автоматически добавит путь к исполняемому файлу EPDFree в ваш файл `.bash_profile`, его полный путь `/Users/ваше_имя/.bash_profile`:

```
# Setting PATH for EPD_free-7.3-1
PATH="/Library/Frameworks/Python.framework/Versions/Current/bin:${PATH}"
export PATH
```

Если на последующих шагах возникнут проблемы, проверьте файл `.bash_profile` – быть может, придется добавить указанный каталог в переменную PATH вручную.

Пришло время установить pandas. Выполните в терминале такую команду:

```
$ sudo easy_install pandas
Searching for pandas
Reading http://pypi.python.org/simple/pandas/
Reading http://pandas.pydata.org
Reading http://pandas.sourceforge.net
Best match: pandas 0.9.0
Downloading http://pypi.python.org/packages/source/p/pandas/pandas-0.9.0.zip
Processing pandas-0.9.0.zip
Writing /tmp/easy_install-H5mIX6/pandas-0.9.0/setup.cfg
Running pandas-0.9.0/setup.py -q bdist_egg --dist-dir /tmp/easy_install-H5mIX6/
pandas-0.9.0/egg-dist-tmp-RhLG0z

Adding pandas 0.9.0 to easy-install.pth file
Installed /Library/Frameworks/Python.framework/Versions/7.3/lib/python2.7/
site-packages/pandas-0.9.0-py2.7-macosx-10.5-i386.egg
Processing dependencies for pandas
Finished processing dependencies for pandas
```

Чтобы убедиться в работоспособности, запустите IPython в режиме Pylab и проверьте импорт pandas и интерактивное построение графика:

```
$ ipython --pylab
22:29 ~/VirtualBox VMs/WindowsXP $ ipython
Python 2.7.3 |EPD_free 7.3-1 (32-bit)| (default, Apr 12 2012, 11:28:34)
Type "copyright", "credits" or "license" for more information.

IPython 0.12.1 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

Welcome to pylab, a matplotlib-based Python environment [backend: WXAgg].
For more information, type 'help(pylab)'.

In [1]: import pandas

In [2]: plot(arange(10))
```

Если все нормально, появится окно графика, содержащее прямую линию.

GNU/Linux

В некоторые, но не во все дистрибутивы Linux включены достаточно актуальные версии всех необходимых Python-пакетов, и их можно установить с помощью встроенного средства управления пакетами, например `apt`. Я продемонстрирую установку на примере EPDFree, потому что она типична для разных дистрибутивов.

Детали варьируются в зависимости от дистрибутива Linux, я буду ориентироваться на дистрибутив Debian, на базе которого построены такие системы, как

Ubuntu и Mint. Установка в основных чертах производится так же, как для OS X, отличается только порядок установки EPDFree. Установщик представляет собой скрипт оболочки, запускаемый из терминала. В зависимости от разрядности системы нужно выбрать установщик типа `x86` (32-разрядный) или `x86_64` (64-разрядный). Имя соответствующего файла имеет вид `epd_free-7.3-1-rh5-x86_64.sh`. Для установки нужно выполнить такую команду:

```
$ bash epd_free-7.3-1-rh5-x86_64.sh
```

После подтверждения согласия с лицензией вам будет предложено указать место установки файлов EPDFree. Я рекомендую устанавливать их в свой домашний каталог, например `/home/wesm/epd` (вместо `wesm` подставьте свое имя пользователя).

После того как установщик отработает, добавьте в свою переменную `$PATH` подкаталог `bin` EPDFree. Если вы работаете с оболочкой bash (в Ubuntu она подразумевается по умолчанию), то нужно будет добавить такую строку в свой файл `.bashrc`:

```
export PATH=/home/wesm/epd/bin:$PATH
```

Естественно, вместо `/home/wesm/epd/` следует подставить свой установочный каталог. Затем запустите новый процесс терминала или повторно выполните файл `.bashrc` командой `source ~/.bashrc`.

Далее понадобится компилятор C, например gcc; во многие дистрибутивы Linux gcc включен, но это необязательно. В системах на базе Debian установка gcc производится командой:

```
sudo apt-get install gcc
```

Если набрать в командной строке слово `gcc`, то должно быть напечатано сообщение:

```
$ gcc
gcc: no input files
```

Теперь можно устанавливать pandas:

```
$ easy_install pandas
```

Если вы устанавливали EPDFree от имени пользователя root, то, возможно, придется добавить в начало команды слово `sudo` и ввести пароль. Проверка работоспособности производится так же, как в случае OS X.

Python 2 и Python 3

Сообщество Python в настоящее время переживает затянувшийся переход от семейства версий Python 2 к семейству Python 3. До появления Python 3.0 весь код на Python был обратно совместимым. Сообщество решило, что ради прогресса

Конец ознакомительного фрагмента.
Приобрести книгу можно
в интернет-магазине «Электронный универс»
(e-Univers.ru)