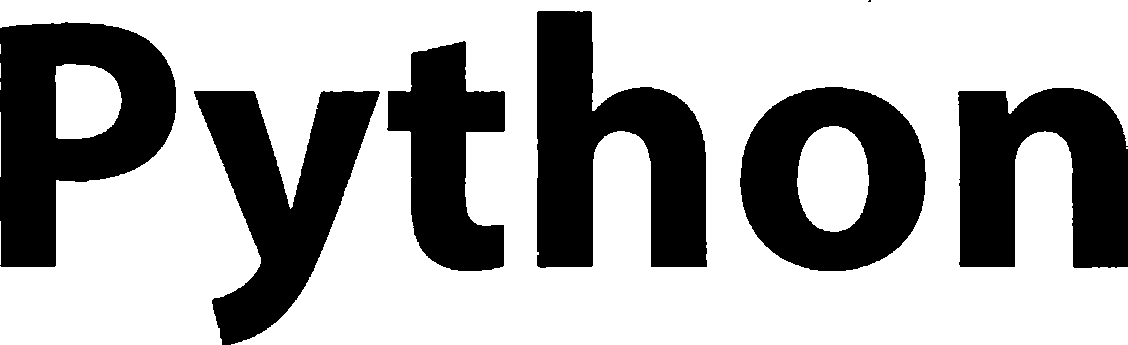




Уэс Маккинли



**и анализ данных**



**Python for Data Analysis**

*Wes McKinney*



**O'REILLY®**

Beijing • Cambridge • Farnham • Kбln • Sebastopol • Tokyo

**УДК 004.438Python:004.6 ББК 32.973.22**

**М15**

М15 Уэс Маккинли

Python и анализ данных/ Пер. с англ. Слинкин А. А. - М.: ДМК Пресс, 2015. - 482 с.: ил.

**ISBN 978-5-97060-315-4**

Книгу можно рассматривать как современное практическое введение в разработку научных приложений на Python, ориентированных па обработку данных. Описаны те части языка Python и библиотеки для него, которые не­ обходимы для эффективного решения широкого круга аналитических задач: интерактивная оболочка IPython, библиотеки NumPy и pandas, библиотека для визуализации данных matplotlib и др.

Издание идеально подойдет как аналитикам, только начинающим осва­ ивать обработку данных, так и опытным программистам па Python, еще не знакомым с научными приложениями.

УДК 004.438Python:004.6 ББК 32.973.22

Original English language edition published bу O'Reilly Media, Inc., 1005 Gravenstein Нighway North, Sebastopol, СА 95472. Copyright © 2013 O'Reilly Mcdia, Inc. Russian­ language edition copyright © 2015 Ьу DMK Press. All rights reserved.

Все права защищены. Любая часть этой книги может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

Материал, изложенный в данной книге, многократно проверен. Но, поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 978-1-449-31979-3 (англ.)

ISBN 978-5-97060-315-4 (рус.)

Copyright © 2013 Wes McKinney.

© Оформление, перевод на русский язык, издание, ДМК Пресс

Предисловие 12

[Графические выделения 12](#_TOC_250191)

Глава 1. Предварительные сведения 13

[О чем эта книга? 13](#_TOC_250190)

[Почему именно Python? 14](#_TOC_250189)

[Python как клей 14](#_TOC_250188)

Решение проблемы «двух языков» 15

[Недостатки Python 15](#_TOC_250187)

[Необходимые библиотеки для Python 16](#_TOC_250186)

[NumPy 16](#_TOC_250185)

[pandas 16](#_TOC_250184)

[matplotlib 17](#_TOC_250183)

IPython 17

[SciPy 18](#_TOC_250182)

[Установка и настройка 18](#_TOC_250181)

[Windows 19](#_TOC_250180)

Apple 05 Х 21

[GNU/Linux 22](#_TOC_250179)

Python 2 и Python 3 23

[Интегрированные среды разработки (IDE) 24](#_TOC_250178)

[Сообщество и конференции 24](#_TOC_250177)

[Структура книги 25](#_TOC_250176)

[Примеры кода 25](#_TOC_250175)

[Данные для примеров 25](#_TOC_250174)

[Соглашения об импорте 25](#_TOC_250173)

[Жаргон 26](#_TOC_250172)

[Благодарности 26](#_TOC_250171)

Глава 2. Первые примеры 28

Набор данных 1.usa.gov с сайта bit.ly 28

[Подсчет часовых поясов на чистом Python 30](#_TOC_250170)

[Подсчет часовых поясов с помощью pandas 32](#_TOC_250169)

[Набор данных Movielens 1 М 38](#_TOC_250168)

[Измерение несогласия в оценках 42](#_TOC_250167)

[Имена, которые давали детям в США за период с 1880 по 2010 год 43](#_TOC_250166)

Анализ тенденций в выборе имен 48

[Выводы и перспективы 56](#_TOC_250165)

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

[Основы IPython 58](#_TOC_250164)

[Завершение по нажатию клавиши Таb 59](#_TOC_250163)

[Интроспекция 60](#_TOC_250162)

[Команда ·%run 61](#_TOC_250161)

[Исполнение кода из буфера обмена 63](#_TOC_250160)

[Комбинации клавиш 64](#_TOC_250159)

[Исключения и обратная трассировка 65](#_TOC_250158)

[Магические команды 66](#_TOC_250157)

[Графическая консоль на базе Qt 68](#_TOC_250156)

Интеграция с matplotlib и режим pylab 68

[История команд 70](#_TOC_250155)

[Поиск в истории команд и повторное выполнение **70**](#_TOC_250154)

Входные и выходные переменные **71**

[Протоколирование ввода-вывода 72](#_TOC_250153)

[Взаимодействие с операционной системой 73](#_TOC_250152)

[Команды оболочки и псевдонимы 73](#_TOC_250151)

[Система закладок на каталоги 75](#_TOC_250150)

[Средства разработки программ ... 75](#_TOC_250149)

[Интерактивный отладчик 75](#_TOC_250148)

[Хронометраж программы: %time и %timeit 80](#_TOC_250147)

[Простейшее профилирование: %prun и %run -р 82](#_TOC_250146)

[Построчное профилирование функции 83](#_TOC_250145)

[НТМL-блокнот в IPython 86](#_TOC_250144)

[Советы по продуктивной разработке кода с использованием IPython 86](#_TOC_250143)

[Перезагрузка зависимостей модуля 87](#_TOC_250142)

[Советы по проектированию программ 88](#_TOC_250141)

[Дополнительные возможности IPython 90](#_TOC_250140)

Делайте классь1 дружественными к IPython 90

[Профили и конфигурирование 90](#_TOC_250139)

[Благодарности 92](#_TOC_250138)

Глава 4. Основы NumPy: массивы и векторные вычисления 93

[NumPy ndarray: объект многомерного массива 94](#_TOC_250137)

[Создание ndarray 95](#_TOC_250136)

[Тип данных для ndarray 97](#_TOC_250135)

[Операции между массивами и скалярами 100](#_TOC_250134)

[Индексирование и вырезание 100](#_TOC_250133)

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](#_TOC_250132)

[Запись логических условий в виде операций с массивами 113](#_TOC_250131)

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](#_TOC_250130)

[Чтение и сохранение данных в MongoDB 198](#_TOC_250129)

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

[Комбинирование и слияние наборов данных 199](#_TOC_250128)

[Слияние объектов DataFrame как в базах данных 200](#_TOC_250127)

[Слияние по индексу 204](#_TOC_250126)

[Конкатенация вдоль оси 207](#_TOC_250125)

[Комбинирование перекрывающихся данных 211](#_TOC_250124)

[Изменение формы и поворот.......................•. 212](#_TOC_250123)

[Изменение формы с помощью иерархического индексирования 213](#_TOC_250122)

[Поворот из «длинного» в «широкий» формат 215](#_TOC_250121)

[Преобразование данных 217](#_TOC_250120)

[Устранение дубликатов 217](#_TOC_250119)

[Преобразование данных с помощью функции или отображения 218](#_TOC_250118)

[Замена значений 220](#_TOC_250117)

[Переименование индексов осей 221](#_TOC_250116)

[Дискретизация и раскладывание 222](#_TOC_250115)

[Обнаружение и фильтрация выбросов 224](#_TOC_250114)

[Перестановки и случайная выборка 226](#_TOC_250113)

Вычисление индикаторных переменных 227

[Манипуляции со строками 229](#_TOC_250112)

[Методы строковых объектов 230](#_TOC_250111)

[Регулярные выражения 232](#_TOC_250110)

[Векторные строковые функции в pandas 235](#_TOC_250109)

[Пример: база данных о продуктах питания министерства сельского хозяйства США 237](#_TOC_250108)

**Глава 8. Построение графиков и визуализация 244**

Краткое введение в API библиотеки matplotlib 245

[Рисунки и подграфики 246](#_TOC_250107)

[Цвета, маркеры и стили линий 249](#_TOC_250106)

[Риски, метки и надписи 251](#_TOC_250105)

[Аннотации и рисование в подграфике 254](#_TOC_250104)

[Сохранение графиков в файле 256](#_TOC_250103)

[Конфигурирование matplotlib 257](#_TOC_250102)

[Функции построения графиков в pandas 258](#_TOC_250101)

[Линейные графики 258](#_TOC_250100)

Столбчатые диаграммы 260

[Гистограммы и графики плотности 264](#_TOC_250099)

[Диаграммы рассеяния 266](#_TOC_250098)

Нанесение данных на карту: визуализация данных о землетрясении

на Гаити 267

Инструментальная экосистема визуализации для Python 273 [Chaco 274](#_TOC_250097)

[mayavi 274](#_TOC_250096)

[Прочие пакеты 275](#_TOC_250095)

[Будущее средств визуализации 275](#_TOC_250094)

**Глава 9. Агрегирование данных и групповые операции 276**

[Механизм GroupBy 277](#_TOC_250093)

[Обход групп 280](#_TOC_250092)

Выборка столбца или подмножества столбцов 281

[Группировка с помощью словарей и объектов Series 282](#_TOC_250091)

[Г руппировка с помощью функций 284](#_TOC_250090)

[Группировка по уровням индекса 284](#_TOC_250089)

[Агрегирование данных 285](#_TOC_250088)

[Применение функций, зависящих от столбца, и нескольких функций 287](#_TOC_250087)

[Возврат агрегированных данных в «неиндексированном» виде 289](#_TOC_250086)

[Групповые операции и преобразования 290](#_TOC_250085)

Метод apply: часть общего принципа разделения-применения- объединения 292

[Квантильный и интервальный анализ 294](#_TOC_250084)

[Пример: подстановка зависящих от группы значений вместо отсутствующих 296](#_TOC_250083)

Пример: случайная выборка и перестановка 297

Пример: групповое взвешенное среднее и корреляция 299

[Пример: групповая линейная регрессия 301](#_TOC_250082)

[Сводные таблицы и кросс-табуляция 302](#_TOC_250081)

Таблицы сопряженности 304

[Пример: база данных федеральной избирательной комиссии за 2012 год 305](#_TOC_250080)

[Статистика пожертвований по роду занятий и месту работы 308](#_TOC_250079)

[Распределение суммы пожертвований по интервалам 311](#_TOC_250078)

[Статистика пожертвований по штатам 313](#_TOC_250077)

**Глава 1 0. Временные ряды З 16**

[Типы данных и инструменты, относящиеся к дате и времени 317](#_TOC_250076)

[Преобразование между строкой и datetime 318](#_TOC_250075)

[Основы работы с временными рядами 321](#_TOC_250074)

[Индексирование, выборка, подмножества 322](#_TOC_250073)

[Временные ряды с неуникальными индексами 324](#_TOC_250072)

[Диапазоны дат, частоты и сдвиг 325](#_TOC_250071)

[Генерация диапазонов дат 325](#_TOC_250070)

[Частоты и смещения дат 326](#_TOC_250069)

[Сдвиг данных (с опережением и с запаздыванием) 329](#_TOC_250068)

[Часовые пояса 331](#_TOC_250067)

[Локализация и преобразование 332](#_TOC_250066)

[Операции над объектами Timestamp с учетом часового пояса 333](#_TOC_250065)

[Операции между датами из разных часовых поясов 334](#_TOC_250064)

Периоды и арифметика периодов 335

[Квартальная частота периода 337](#_TOC_250062)

Преобразование временных меток в периоды и обратно 339

[Создание Periodlndex из массивов 340](#_TOC_250061)

[Передискретизация и преобразование частоты 341](#_TOC_250060)

[Понижающая передискретизация 342](#_TOC_250059)

Повышающая передискретизация и интерполяция 345

Передискретизация периодов 346

[Графики временных рядов 348](#_TOC_250058)

[Скользящие оконные функции 350](#_TOC_250057)

[Экспоненциально взвешенные функции 353](#_TOC_250056)

[Бинарные скользящие оконные функции 353](#_TOC_250055)

[Скользящие оконные функции, определенные пользователем 355](#_TOC_250054)

Замечания о быстродействии и потреблении памяти 356

**Глава 11. Финансовые и экономические приложения 358**

[О переформатировании данных 358](#_TOC_250053)

[Временные ряды и выравнивание срезов 358](#_TOC_250052)

[Операции над временными рядами с различной частотой 361](#_TOC_250051)

[Время суток и выборка данных «по состоянию на» 364](#_TOC_250050)

[Сращивание источников данных 366](#_TOC_250049)

Индексы доходности и кумулятивная доходность…………………………… ……368

[Групповые преобразования и анализ 370](#_TOC_250048)

[Оценка воздействия групповых факторов 372](#_TOC_250047)

[Децильный и квартильный анализ 373](#_TOC_250046)

[Другие примеры приложений 375](#_TOC_250045)

[Стохастический граничный анализ 375](#_TOC_250044)

Роллинг фьючерсных контрактов 377

[Скользящая корреляция и линейная регрессия 380](#_TOC_250043)

**Глава 12. Дополнительные сведения о библиотеке NumPy 383**

Иерархия типов данных в NumPy 384

[Дополнительные манипуляции с массивами 385](#_TOC_250042)

[Изменение формы массива 385](#_TOC_250041)

[Упорядочение элементов массива в С и в Fortran 387](#_TOC_250040)

[Конкатенация и разбиение массива 388](#_TOC_250039)

[Повторение элементов: функции tile и repeat 390](#_TOC_250038)

[Эквиваленты прихотливого индексирования: функции take и put 391](#_TOC_250037)

[Укладывание 393](#_TOC_250036)

[Укладывание по другим осям 394](#_TOC_250035)

[Установка элементов массива с помощью укладывания 397](#_TOC_250034)

[Дополнительные способы использования универсальных функций 398](#_TOC_250033)

Методы экземпляра u-функций 398

Пользовательские u-функции 400

[Структурные массивы.........................................................' 401](#_TOC_250032)

[Вложенные типы данных и многомерные поля 402](#_TOC_250031)

[Зачем нужны структурные массивы? 403](#_TOC_250030)

Манипуляции со структурными массивами: numpy.lib.recfunctions 403

[Еще о сортировке 403](#_TOC_250029)

[Косвенная сортировка: методы argsort и lexsort 405](#_TOC_250028)

[Альтернативные алгоритмы сортировки 406](#_TOC_250027)

[Метод numpy.searchsorted: поиск элементов в отсортированном массиве 407](#_TOC_250026)

[Класс matrix в NumPy...................................\_. 408](#_TOC_250025)

Дополнительные сведения о вводе-выводе массивов 410

[Файлы, спроецированные на память 410](#_TOC_250024)

HDF5 и другие варианты хранения массива 412

[Замечание о производительности 412](#_TOC_250023)

[Важность непрерывной памяти 412](#_TOC_250022)

[Другие возможности ускорения: Cython, f2py, С 414](#_TOC_250021)

**Приложение. Основы языка Python 415**

[Интерпретатор Python ....................., 416](#_TOC_250020)

[Основы 417](#_TOC_250019)

[Семантика языка 417](#_TOC_250018)

[Скалярные типы 425](#_TOC_250017)

[Поток управления 431](#_TOC_250016)

[Структуры данных и последовательности 437](#_TOC_250015)

[Список 439](#_TOC_250014)

[Встроенные функции последовательностей 443](#_TOC_250013)

[Словарь 445](#_TOC_250012)

[Множество 448](#_TOC_250011)

[Списковое, словарное и множественное включение 450](#_TOC_250010)

[Функции 452](#_TOC_250009)

[Пространства имен, области видимости и локальные функции 453](#_TOC_250008)

Возврат нескольких значений 454

[Функции являются объектами 455](#_TOC_250007)

[Анонимные (лямбда) функции 456](#_TOC_250006)

[Замыкания: функции, возвращающие функции 457](#_TOC_250005)

Расширенный синтаксис вызова с помощью \*args и \*\*kwargs 459

[Каррирование: частичное фиксирование аргументов 459](#_TOC_250004)

[Генераторы 460](#_TOC_250003)

[Генераторные выражения 462](#_TOC_250002)

[Модуль itertools 462](#_TOC_250001)

[Файлы и операционная система 463](#_TOC_250000)

**Предметный указатель 46**

**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 сло жилось 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-aдpeca, 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, ориентиро­ ванное 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 пре­ образовать 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 рас­

четов с визуализацией результатов Python неизбежно приходится сравнивать со многими предметно-ориентированными языками программирования и ин­ струментами - 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 С, С++ и FORTRAN. Во многих современных вычислительных средах применяется общий набор унаследованных библиотек, написанных на FORTRAN и С, 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 С.

За последние несколько лет на одно из первых мест в области создания быстрых компилируемых расширений Python и организации интерфейса с кодом на С и С++ вышел проект Cython *(http.j/cython.01-g).*

***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, С# или С++. 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 С++. Но поскольку *время програмиста* 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, по­ траченное на программирование на низкоуровневом и не обеспечивающем максимальную продуктивность языке типа С++, 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 многопо­ точного параллельного кода; просто такой код нельзя выполнять в одном процессе Python. Например, в проекте Cython реализована простая интеграция с OpenMP, написанной на С 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 *пdттау;*
* 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 случай­ ных чисел;

•

средства для интеграции с кодом, написанным на С, С++ 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, би­ блиотеки, написанные на низкоуровневом языке типа С или Fortran, могут рабо­ тать 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 вырази­ тельным 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:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | total bill | tip | sex | smoker | day | time | size |
| 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 данными, свойственными электронным таблицам и реляционным базам данных (на­ пример, на основе SQL). Она предоставляет развитые средства индексирования, позволяющие без труда изменять форму наборов данных, формировать продоль­ ные 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 вы­ сокопроизводительных средств анализа временных рядов, специально ориентиро­ ванных 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, н 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 *рапе/ data* (панельные данные), при­ меняемого в экономстрике термина для обозначения многомерных структуриро­ ванных 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 Qohn 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:

* НТМL-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 се при чтении этой книги.

***SciPy***

SciPy - собрание пакетов, предназначенных для решения различных стандарт­ ных вычислительных задач. Вот несколько из них:

* + scipy. integrate: подпрограммы численного интегрирования и решения дифференциальных уравнений;
  + scipy. linalg: подпрограммы линейной алгебры и разложения матриц, до­

полняющие те, что включены в numpy. l inalg;

* + scipy. optimize: алгоритмы оптимизации функций (нахождения миниму­ мов) 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 ве­ роятностей (функции плотности вероятности, формирования выборки,

функции непрерывного распределения вероятности), различные статисти­ ческие критерии и дополнительные описательные статистики;

* + scipy. weave: средство для встраивания кода на С++ 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 единственноверной 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 дис­ трибутив от компании Enthought (*http.www.enthought.com*).

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 ис­ пользования, а прочим пользователям предлагается платная годовая подписка.

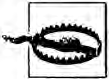
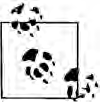
Python(x,y) *(*[*http://pythonxy.google.com):*](http://pythonxy.google.com):)бесплатный ориентированный на научные применения дистрибутив для Windows.

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, РуТаЫеs, PyQt (или эквивалентный ему PySide), xml, lml, basemap, pymongo и requests. Они используются в разных примерах. Устанавливать их не­ обязательно, и я рекомендую не торопиться с этим до момента, когда они действи­ тельно понадобятся. Например, сборка PyQt или РуТаЫеs из исходных кодов в OS Х или Linux - довольно муторное дело. А пока нам важно получить минимальную работоспособную среду: EPDFree и pandas.

Сведения обо всех Руthоn-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 Руthоn-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 реали­ зации Python, например lronPython, Jython или РуРу. Но для работы с ин­ струментами, представленными в этой книге, в настоящее время необхо­ дим стандартный написанный на С интерпретатор Python, известный под названием CPython.

***Windows***

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

рекомендую предварительно удалить его вручную (или с помощью средства “Установка и удаление проrрамм:”).

Затем нужно убедиться, что Python прописался в системной переменной РАТН и что не возникло конфликтов с ранее установленными версиями Python. Откройте консоль (выберите из меню Пуск)> пункт выполнить и наберите cmd.ехе). Попробуйте запустить интерпретатор Python, введя команду python. Должно по­ явиться сообщение, в котором указана установленная версия EPDFree:

*C:\Users\Wes>python*

*Python 2.7.3 IEPD\_free 7.3-1 (32-bit) 1 (default, Apr 12 2012, 14:30:37) on win32 Туре "credits", "demo" or "enthought'; for more information.*

*>>>*

Если в сообщении указана другая версия EPD или вообще ничего не запуска­ ется, то нужно привести в порядок переменные cpeдыWindows. В Windows 7 нач­ ните вводить фразу <<environment variaЬles:1> в поле поиска программ и выберите раздел Edit environment varibles for your account. В Windows ХР нужно перейти в Панель управления > Система > Дополнительно > Переменные сре­ ды. В появляющемся окне найдите переменную Path. В ней должны присутство­ вать 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.*](http://pypi.python.org/pypi/pandas)Для EPDFree это будет файл pandas-0.9.0.win32-py2.7.ехе. 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 IEPD\_free 7.3-1 (32-bit) 1*

*Туре "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.*

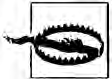
*help -> Python's own help system.*

*object? -> Details about 'object', use 'object??" for extra details.*

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

*In [1]: import pandas*

*In [2]: plot(arange(lO))*



**Установка н настройка**

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

$ ipython notebook --pylab=inline

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

Дистрибутив EPDFree для Windows содержит только 32-разрядные исполняе­ мые 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 Гальке (Christoph Gohlke) из Калифорнийского университета в Ирвайне опубликовал неофициальные двоичные 32- и 64-разрядные установщики для всех используемых в книге пакетов [*(http.j/www.lfd.uci.edu/-gohlke/*](http://www.lfd.uci.edu/-gohlke/) *pythonlibs/).*

***AppleOSX***

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

После установки Xcode запустите терминал (Terminal.app), перейдя в меню Ap­ plications > Utilities. Введите gcc и нажмите клавишу Enter. Должно по­ явиться такое сообщение:

*$ gcc*

*i686-apple-darwin10-gcc-4.2.1: по input files*

Теперь необходимо установить EPDFree. Скачайте установщик, который дол­ жен представлять собой образ диска с именем вида epd\_free-7.3-1-macosx­i386.dmg. Дважды щелкните мышью по файлу dmg-файлу, чтобы смонтировать его, а затем дважды щелкните по mрkg-файлу, чтобы запустить установщик.

Установщик автоматически добавит путь к исполняемому файлу EPDFree в ваш файл .bash\_profile, его полный путь /Usеrs/ваше\_имя/ .bash\_profile:

# Setting РАТН for EPD\_free-7.3-1 PATH="/Library/Frameworks/Python.framework/Versions/Current/Ьin:${PATH}" export РАТН

Если на последующих шагах возникнут проблемы, проверьте файл . bash\_ profile - быть может, придется добавить указанный каталог в переменную РАТН вручную.

Пришло время установить pandas. Выполните в терминале такую команду:

*$ sudo easy\_install pandas Searching for pandas*

*Reading* [*http://pypi.python.org/simple/pandas/*](http://pypi.python.org/simple/pandas/) *Reading* [*http://pandas.pydata.org*](http://pandas.pydata.org/)

*Reading* [*http://pandas.sourceforge.net*](http://pandas.sourceforge.net/) *Best match: pandas 0.9.0*

*Downloading* [*http://pypi.python.org/packages/source/p/pandas/pandas-0.9.0.zip*](http://pypi.python.org/packages/source/p/pandas/pandas-0.9.0.zip) *Processing pandas-0.9.0.zip*

*Writing /tmp/easy\_install-HSmIXб/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-RhLGOz*

*Adding pandas 0.9.0 to easy-install.pth file*

*Installed /Library/Frameworks/Python.framework/Versions/7.3/liЬ/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 rрафика:

*$ ipython --pylab*

*22:29 -/VirtualBox VМs/WindowsXP $ ipython*

*Python 2.7.3 IEPD\_free 7.3-1 (32-Ьit)I (default, Apr 12 2012, 11:28:34) Туре "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, а matplotlib-based Python environment [backend: WXAgg]. For more information, type 'help(pylaЫ '.*

*In [1]: import pandas*

*In [2]: plot(arange(lO))*

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

|  |  |
| --- | --- |
|  | В некоторые, но не во все дистрибутивы Linux включены достаточно актуальные версии всех необходимых Python-naкeтoв, 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 ориенти­ роваться на дистрибутив Deblan, на базе котороrо построены такие системы, как Ubuntu и Mint. Установка в основных чертах производится так же, как для OS Х, отличается только порядок установки EPDFree. Установщик представляет собой скрипт оболочки, запускаемый из терминала. В зависимости от разрядности си­ стемы нужно выбрать установщик типа хВб (32-разрядный) или х86\_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 ме­ сто установки файлов EPDFree. Я рекомендую устанавливать их в свой домашний каталог, например /home/wesm/epd (вместо wesm подставьте свое имя пользовате­ ля).

После того как установщик отработает, добавьте в свою переменную $РАТН под­ каталог bin EPDFree. Если вы работаете с оболочкой bash (в Ubuntu она под­ разумевается 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 С, например gcc; во многие дистрибутивы Linux gcc включен, но это необязательно. В системах на базе Deblan установка gcc производится командой:

*sudo apt-get install gcc*

Если набрать в командной строке слово gcc, то должно быть напечатано со­ общение:

*$ 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 командьi слово sudo и ввести пароль. Проверка рабо­ тоспособности производится так же, как в случае OS Х.

***Python 2 и Python* З**

Сообщество Python в настоящее время переживает затянувшийся переход от семейства версий Pytl1011 2 к семейству Python 3. До появления Python 3.0 весь код на Python был обратно совместимым. Сообщество решило, что ради прогресса языка необходимо внести некоторые изменения, которые лишат код этого свой­ ства.

При написании этой книги я взял за основу Python 2.7, потому что большая часть научного сообщества Python еще не перешла на Python 3. Впрочем, если не считать немногих исключений, у вас не возникнет трудностей с исполнением приведенного в книге кода, даже если работать с Python 3.2.

***Интегрированные среды разработки (IDE)***

Когда меня спрашивают о том, какой средой разработки я пользуюсь, я почти всегда отвечаю: «IPython плюс текстовый редактор». Обычно я пишу программу и итеративно тестирую и отлаживаю ее по частям в IPython. Полезно также иметь возможность интерактивно экспериментировать с данными и визуально проверять, что в результате определенных манипуляций получается ожидаемый результат. Библиотеки pandas и NumPy спроектированы с учетом простоты использования в оболочке.

Но кто-то по-прежнему предпочитает работать в IDE, а не в текстовом редак­ торе. Интегрированные среды действительно предлагают много полезных «фенечек» типа автоматического завершения или быстрого доступа к документации по функциям и классам. Вот некоторые доступные варианты:

* + Eclipse с подключаемым модулем PyDev;
  + Python Tools для Visual Studio (для работающих в Windows);
  + PyCharm;
  + Spyder;
  + Komodo IDE.

Сообщество и конференции

Помимо поиска в Интернете, существуют полезные списки рассылки, посвящен­ ные использованию Python в научных расчетах. Их участники быстро отвечают на вопросы. Вот некоторые из таких ресурсов:

* + pydata: группа Google по вопросам, относящимся к использованию Python для анализа данных и pandas;
  + pystatsmodels: вопросы, касающиеся statsmodels и pandas;
  + numpy-discussion: вопросы, касающиеся NumPy;
  + scipy-user: общие вопросы использования SciPy и Python для научных рас­ четов.

Я сознательно не публикую URL-aдpeca, потому что они часто меняются. Поиск в Интернете вам в помощь.

Ежегодно в разных странах проводят конференции для программистов на Python. PyCon и EuroPython - две самых крупных, проходящие соответственно в США и в Европе. SciPy и EuroSciPy - конференции, ориентированные па научные применения Python, где вы найдете немало «собратьев», если, прочитав эту книгу, захотите более плотно заняться анализом данных с помощью Python.

**Структура книги**

Если вы раньше никогда не программировали на Python, то имеет смысл начать с конца книги, где я поместил очень краткое руководство по синтаксису Python, языковым средствам и встроенным структурам данных: кортежам, спискам и сло­ варям. Эти знания необходимы для чтения книги.

Книга начинается знакомством со средой IPython. Затем следует краткое вве­ дение в основные возможности NumPy, а более продвинутые рассматриваются в другой главе ближе к концу книги. Далее я знакомлю читателей с библиотекой pandas, а все остальные rлавы посвящены анализу данных с помощью pandas, NumPy и matplotlib (для визуализации). Я старался располагать материал последовательно, но иногда главы все же немного перекрываются.

Файлы с данными и материалы, относящиеся к каждой главе, размещаются в репозитории git на сайте GitHub:

[*http://github.com/pydata/pydata-book*](http://github.com/pydata/pydata-book)

Призываю вас скачать данные и пользоваться ими при воспроизведении при­ меров кода и экспериментах с описанными в книге инструментами. Я буду благодарен за любые добавления, скрипты, блокноты IPython и прочие материалы, которые вы захотите поместить в репозиторий книги для всеобщей пользы.

***Примеры кода***

Примеры кода в большинстве случаев показаны так, как выглядят в оболочке IPython: ввод и вывод.

*In[5]:код*

*Out[5]: результат*

Иногда для большей ясности несколько примеров кода показаны рядом. Их следует читать слева направо и исполнять по отдельности.

|  |  |
| --- | --- |
| *In [5]: код*  *Out[5]: результат* | *In [6]:код2*  *Out[6]: результат2* |

***Данные для примеров***

Наборы данных для примеров из каждой главы находятся в репозитории на сайте GitHub: *http.//{github.com/pydata/pydata-book.* Вы можете получить их либо с помощью командной утилиты системы управления версиями git, либо скачав ziр-файл репозитория с сайта.

Я стремился, чтобы в репозиторий попало все необходимое для воспроизведения примеров, но мог где-то ошибиться или что-то пропустить. В таком случае пишите мне на адрес wesmckinn@gmail.com

***Соглашения об импорте***

В сообществе Python принят ряд соглашений об именовании наиболее употре­ бительных модулей:

*import numpy as np import pandas as pd*

*import matplotlib.pyplot as plt*

Это означает, что arange - ссылка на функцию arange в пакете NumPy. Так делается, потому что импорт всех имен из большого пакета, каким: является NumPy (from numpy import \*), считается среди разработчиков на Python дурным тоном.

**Жаргон**

Я употребляю некоторые термины, встречающиеся как в программировании, так и в науке о данных, с которыми вы, возможно, незнакомы. Поэтому приведу краткие определения.

* Переформатированuе (Munge/Munging/Wгangling)

Процесс приведения неструктурированных и (или) замусоренных

данных к структурированной или чистой форме. Слово вошло в лексикон многих современных специалистов по анализу данных.

* Псевдокод

Описание алгоритма или процесса в форме, напоминающей код, хотя фактически это не есть корректный исходный код на каком-то языке программирования.

* Синтаксическая глазурь (syntactic sugar)

Синтаксическая конструкция, которая не добавляет какую-то новую функциональность, а лишь вносит дополнительное удобство или позволяет сделать код короче.

Благодарности

Я не смог бы написать эту книгу без помощи со стороны многих людей.

Из сотрудников издательства O'Reilly я крайне признателен своим редакторам Меган Бланшетт (Meghan Blanchette) и Джулии Стил Uulie Steele), которые на­ правляли меня на протяжении всей работы. Майк Лукидес (Mike Loukides) работал со мной на этапе подготовки предложения и помог превратить замысел в реальность.

Техническое рецензирование осуществляла большая группа людей. В частно­ сти, Мартин Блез (Martin Blais) и Хью Уайт (Hugh White) оказали неоценимую помощь в подборе примеров, повышении ясности изложения и улучшении струк­ туры книги в целом. Джеймс Лонг Qames Long), Дрю Конвей (Drew Conway), Фернандо Перес (Fernando Perez), Брайан Грейнджер (Brian Granger), Томас Клюйвер (Thomas Kluyver), Адам Клейн (Adam Klein), Джош Клейн Uosh Klein), Чань Ши (Chang She) и Стефан ван дер Вальт (Stefan van der Walt) просмотрели по одной или по несколько глав под разными углами зрения.

Много идей по поводу примеров и наборов данных мне предложили друзья и

коллеги по сообществу анализа данных, в том числе: Майк Дьюар (Mike Dewar)

Джефф Хаммербахер (JeH Hammerbacher), Джеймс Джондроу (JamesJohndrow), Кристиан Лам (Kristian Lum), Адам Клейн, Хилари Мейсон (Hilary Mason), Чань Ши и Эшли Уильямс (Ashley Williaшs).

Разумеется, я в долгу перед многими лидерами сообщества «научного Pythonj,>, которые создали открытый исходный код, легший в основу моих исследований, и оказывали мне поддержку па протяжении всей работы над книгой: группа раз­ работки ядра IPython (Фернандо Перес, Брайан Грейнджер, Мин Рэган-Келли (Min Ragan-Kelly), Томас Клюйвер и другие), Джон Хантер (John Hunter), Скип­ пер Сиболд (Skipper Seabold), Трэвис Олифант (Travis Oliphant), Питер Уонг (Peter Wang), Эрик Джоунз (EricJones), Роберт Керн (Robert Kern), Джозеф Пер­ ктольд (Josef Perktold), Франческ Олтсд (Francesc Alted), Крис Фоннесбек (Chris Fonnesbeck) и многие, многие другие, перечислять которых здесь нет никакой возможности. Меня также поддерживали, подбадривали и делились идеями Дрю Конвей, Шон Тэйлор (Sean Taylor), Джузеппе Палсолого (Giuseppe Paleologo), Джаред Ландер (Jared Lander), Дэвид Эпштейн (David Epstein), Джог Кровас (John Krowas), Джошуа Блум (Joshua Вlоош), Дэн Пилсфорт (Den Pilsworth), Джон Майлз-Уайт (John Myles-Whit.e) и многие другие, имена которых я сейчас не могу

вспомнить.

Я также благодарен многим, кто оказал влияние на мое становление как уче­ ного. В первую очередь, это мои бывшие коллеги по компании AQR, которые поддерживали мою работу над pandas в течение многих лет: Алекс Рейфман (Alex Reyfшan), Майкл Вонг (Michael Wong), Тим Сарджен (Tim Sargen), Октай Курбанов (Oktay Kurbanov), Мэтью Щанц (Matthew Tschantz), Рони Израэлов (Roni Israelov), Майкл Кац (Michael Katz), Крис Уга (Chris Uga), Прасад Раманан (Prasad Ramanan), Тэд Сквэр (Ted Square) и Хун Ким (Hoon Kim). И наконец, благодарю моих университетских наставников Хэйнса Миллера (МТИ) и Майка Уэста (университет Дьюк).

Если говорить о личной жизни, то я благодарен Кэйси Динкин (Casey Dinkin), чью каждодневную поддержку невозможно переоценить, ту, которая терпела пере­ пады моего настроения, когда я пытался собрать окончательный вариант рукописи в дополнение к своему и так уже перегруженному графику. А также моим родителям, Биллу и Ким, которые учили меня никогда не отступать от мечты и не соглашаться на меньшее.