УДК 004.424
ББК 32.372
У97

Уэйд Р.
У97 Аналитика в Power BI с помощью R и Python / пер. с англ. А. Ю. Гинько. – М.: ДМК Пресс, 2021. – 338 с.: ил.

ISBN 978-5-97060-923-1

В данной книге подробно рассказывается, как использовать на практике языки программирования R и Python для визуализации данных, загрузки в модель, преобразования и выполнения других задач с помощью аналитического инструмента Power BI. Вы узнаете, как создавать пользовательские элементы визуализации, реализовывать методы машинного обучения и искусственного интеллекта, применять продвинутые методы обработки текстовой информации с использованием техник, недоступных в Power Query и DAX, обеспечивать взаимодействие со службами Microsoft Cognitive Services без необходимости приобретать дорогостоящую подписку на Power BI Premium. В заключение рассказывается, как можно воспользоваться языками программирования R и Python в корпоративных решениях, внедренных в Power BI.

Для выполнения практических упражнений понадобится облачная платформа Microsoft Azure. Также для работы с примерами из данной книги рекомендуется настроить виртуальную машину для анализа данных (Data Science Virtual Machine – DSVM).

Издание адресовано читателям, которые работают с большими объемами данных и хотят эффективно применять инструменты бизнес-аналитики

УДК 004.424
ББК 32.372

ISBN 978-1-4842-5828-6 (англ.)
ISBN 978-5-97060-923-1 (рус.)

Содержание

Об авторе

Райан Уэйд (Ryan Wade) является профессиональным аналитиком данных с более чем 20-летним стажем. Своему образованию и опыту работы Райан обязан тем, что приобрел целостное понимание аналитических процессов как с технической, так и с организационной точки зрения. Является обладателем сертификата MCSE в области бизнес-аналитики и Microsoft R и профессионально программирует на R, Python, DAX, T-SQL, M и VBA применительно к локальным и облачным решениям в аналитике на *платформе данных Microsoft* (Microsoft Data Platform).

Райан принимает активное участие в открытых мероприятиях по R и Python, а также выступает на конференциях SQLSaturdays, TDWI, BDPA и PASS Summit с лекциями по анализу данных. Разработал полноценный онлайн-курс для ExcelTv, в рамках которого демонстрирует применение языка R в Power BI для проведения углубленного анализа данных и их визуализации.

Введение

Microsoft Power BI является одним из наиболее популярных инструментов в области бизнес-аналитики. За последние годы этот программный комплекс опередил своих прямых конкурентов QlikView и Tableau и прочно занял лидирующее положение на рынке. Одним из главных преимуществ Power BI является то, что он представляет собой нечто гораздо большее, чем просто инструмент визуализации данных. Вот лишь несколько явных преимуществ Power BI:

- встроенный язык запросов *DAX*, позволяющий крайне эффективно извлекать информацию из модели данных с применением сложной бизнес-логики;
- интегрированный инструмент подготовки и преобразования данных *Power Query*, при помощи которого можно легко извлекать и трансформировать исходную информацию в вид, пригодный для анализа;
- движок *Vertipaq*, позволяющий хранить данные в оптимальном для формирования отчетов виде и быстро и эффективно обрабатывающий сложные вычисления;
- заранее подготовленные пакеты интерактивных элементов визуализации, помогающие представлять данные в понятной и четкой форме.

Глядя на этот список преимуществ, вы вполне можете задаться вопросом, зачем же столь мощному инструменту понадобилась помощь языков программирования R и Python. Ответ прост – чтобы заполнить области, в которых встроенные средства недостаточно хороши. Вот лишь несколько примеров применения этих языков программирования в рамках Power BI:

- создание пользовательских элементов визуализации без особых усилий;
- реализация интеллектуальной обработки данных, методов машинного обучения и искусственного интеллекта без необходимости приобретать дорогостоящую подписку на *Power BI Premium*;
- применение продвинутых методов обработки текстовой информации с использованием техник, недоступных в *Power Query* и *DAX*;
- взаимодействие со службами *Microsoft Cognitive Services* без необходимости приобретать подписку на *Power BI Premium*;
- взаимодействие со сторонними интерфейсами API с целью эффективного обогащения моделей данных Power BI;
- и многое другое…

В данной книге мы подробно расскажем о том, как использовать на практике языки программирования R и Python для обеспечения всей перечисленной выше функциональности в Power BI. Язык R идеально подходит для Power BI по причине того, что он был создан специально для анализа данных. Уже долгие годы аналитики активно используют R для преобразования и визуализации информации. Так что то немногое, на что не способна программная среда Power BI, может быть с лихвой компенсировано при помощи языка R.

Что касается Python, то этот язык программирования приобрел чрезвычайную популярность в области анализа данных в последнее десятилетие. Одним

из главных преимуществ Python является то, что он подходит для решения не только аналитических задач, но и общих задач программирования. К примеру, взаимодействие с интерфейсами API довольно легко осуществить при помощи Python, тогда как посредством Power Query это будет сделать не так-то просто.

Все эти особенности делают языки R и Python идеально подходящими для такого мощного аналитического инструмента, как Power BI. И в книге, которую вы держите в руках, мы проиллюстрируем на примерах все перечисленные выше возможности и техники. При этом все решения и сценарии будут сопровождаться подробными описаниями, чтобы вы досконально поняли используемую реализацию.

Но перед тем как приступать к конкретным примерам, необходимо для начала настроить среду выполнения. Давайте пройдемся по соответствующим пунктам и подготовимся к работе.

Настройка вашего окружения Azure

В данной книге мы будем часто упоминать и пользоваться различными составляющими облачной платформы *Microsoft Azure*. В частности, для внедрения механизмов искусственного интеллекта в модели данных Power BI мы будем использовать набор служб *Microsoft Cognitive Services*. Также для работы с примерами из данной книги рекомендуется настроить *виртуальную машину для анализа данных* (Data Science Virtual Machine – DSVM). Это не обязательное условие, но желательное. Кроме того, для комфортной работы с тем, что мы будем обсуждать, вам необходимо настроить окружение со следующими инструментами:

- SQL Server 2017 или выше;
- SQL Server Machine Learning Services 2017 или выше с поддержкой R и Python;
- дистрибутив Python из Anaconda;
- R;
- R Studio;
- VS Code;
- Power BI Desktop.

Сконфигурировать такое окружение вручную – задача не из легких, но если вы установите виртуальную машину для анализа данных, большинство настроек будет выполнено за вас автоматически. В следующих разделах мы поговорим о том, как настроить *Azure*, чтобы можно было пользоваться службами *Microsoft Cognitive Services*. Кроме того, вы узнаете, как развернуть *DSVM*.

Подписка на Azure

Оформить подписку на *Azure* можно по адресу https://azure.microsoft.com/en-us/free. В результате вы получите 12 месяцев на пользование выборочными службами плюс кредит на сумму $200 на первый месяц.

Подписка на Microsoft Cognitive Services

Мы будем использовать службу *Microsoft Cognitive Services* для проведения *анализа тональности текста* (sentiment analysis) в Power BI с помощью Python. Для начала вам необходимо будет настроить службу *Microsoft Cognitive Services* в *Azure*, после чего можно будет обращаться к ней из Power BI. Для настройки службы нужно выполнить следующие шаги.

1. Войдите на портал Azure.
2. Введите в строке поиска *Cognitive Services* и нажмите **Enter**.
3. Вы должны оказаться на странице *Cognitive Services*. Нажмите кнопку создания для запуска процесса регистрации.
4. Введите следующую информацию:
   - имя;
   - подписка;
   - расположение;
   - ценовая категория;
   - группа ресурсов.
5. Установите флажок, оповещающий о том, что вы прочитали и согласны с условиями.
6. Нажмите кнопку создания.

Учтите, что использование службы *Microsoft Cognitive Services* является платным. Чтобы получить информацию о ценах, перейдите к ресурсу *Microsoft Cognitive Services*, введите в окно поиска текст *Ценовая категория* и откройте результаты поиска. Выберите подходящую вам категорию. Вы будете перенаправлены на страницу с информацией о ценах согласно выбранному вами региону. На примеры, показанные в данной книге, вам с лихвой хватит выданного вам на первый месяц кредита.

Создание виртуальной машины для анализа данных (DSVM)

Предпочтительным вариантом будет создание виртуальной машины и добавление ресурсов, не установленных в ней по умолчанию. Я рекомендую идти этим путем, поскольку полностью ручная настройка окружения для выполнения примеров из этой книги может занять уйму времени. Использование виртуальной машины позволит сконфигурировать необходимое окружение за несколько минут – притом что в ручном режиме у вас бы это могло отнять много дней проб и ошибок. Если вы решите пойти длинным путем, позже я опишу примерный план действий. Сейчас же приведу инструкции по настройке виртуальной машины.

Шаги создания виртуальной машины в Azure

1. Перейдите на портал https://portal.azure.com. Если система попросит, введите данные учетной записи, созданной ранее.

2. Нажмите на кнопку **Создать ресурс** (Create a resource) в левом верхнем углу.
3. В строке поиска введите **Data Science Virtual Machine – Windows 2019**.
4. Нажмите на кнопку **Создать** (Create). Появится форма для ввода информации о конфигурации виртуальной машины, в которой будет открыта вкладка **Основные** (Basics). В следующих шагах вы узнаете, как заполнить эту форму.
5. **Подписка** (Subscription): выберите подписку, которую собираетесь использовать. По умолчанию будет выбрана подписка, настроенная ранее.
6. **Группа ресурсов** (Resource group): если у вас уже есть группа ресурсов, которую вы желаете использовать, выберите ее. В противном случае создайте новую для своей виртуальной машины.
7. **Имя виртуальной машины** (Virtual machine name): название, которое вы хотите присвоить виртуальной машине.
8. **Регион** (Region): ближайший к вам географический регион Azure.
9. **Image**: убедитесь, что в данном списке выбран пункт **Data Science Virtual Machine – Windows 2016**.
10. **Размер** (Size): я использую B4ms, поскольку это самый дешевый вариант для 16 Гб оперативной памяти. Этот параметр очень важен для R, Python и Power BI.
11. **Имя пользователя** (Username): придумайте имя пользователя.
12. **Пароль** (Password): введите пароль.
13. **Подтвердите пароль** (Confirm password): введите пароль еще раз.
14. Перейдите на вкладку **Диски** (Disks).
15. **Тип диска ОС** (OS disk type): укажите **Стандартный SSD** (Standard SSD) – это будет оптимальный для нашего случая.
16. Перейдите на вкладку **Сетевые подключения** (Networking).
17. Убедитесь, что все нужные поля заполнены. Обязательные к заполнению поля помечены звездочкой. В эти поля должны быть введены значения по умолчанию. Если таких значений нет, нажмите на ссылку **Создать** (Create new) под соответствующим полем и введите недостающий элемент.
18. Перейдите на вкладку **Управление** (Management).
19. Оставьте все по умолчанию и перейдите на вкладку **Дополнительно** (Advanced).
20. Примите значения по умолчанию и перейдите на вкладку **Теги** (Tags).
21. Откройте вкладку **Просмотр и создание** (Review + create).
22. Вы увидите сводную информацию о создаваемой виртуальной машине. Кроме того, для вас будет рассчитана стоимость ее использования. Если вы согласны с введенными данными, нажмите кнопку **Создать** (Create).

Не забывайте останавливать виртуальную машину после каждого использования, чтобы не платить лишние деньги. На всякий случай вы можете настроить автоматическое отключение машины в определенное время. Для этого необходимо сделать следующее.

1. Перейдите к своей виртуальной машине на портале Azure.
2. Введите текст **Автозавершение работы** (auto-shutdown) в строку поиска и перейдите в соответствующий раздел.
3. Переведите переключатель **Включено** (Enabled) в положение **Вкл** (On).
4. Выберите время, в которое машина будет выключаться, в поле **Запланированное завершение работы** (Scheduled shutdown).
5. В выпадающем списке **Часовой пояс** (Time zone) выберите нужную зону.
6. Если хотите, чтобы система отправляла вам уведомление о выключении виртуальной машины, установите переключатель в разделе **Отправлять уведомление перед автоматическим завершением работы** (Send notification before auto-shutdown) в положение **Да** (Yes). Уведомление будет отправлено на адрес, введенный в поле **Адрес электронной почты** (Email address).

Настройка R на виртуальной машине

Мы будем использовать другой дистрибутив языка R по сравнению с установленным. Нашим выбором будет дистрибутив *Microsoft R Open* (*MRO*). Он полностью совместим с дистрибутивом, распространяемым через центральную систему хранения пакетов CRAN, но при этом существенно улучшен в отношении определенных типов вычислений, а также снабжен дополнительными полезными инструментами. Выполните следующие шаги, чтобы загрузить дистрибутив *MRO* в виртуальную машину.

1. Узнайте версию R, используемую в Power BI. Соответствующую информацию можно найти на сайте Microsoft по адресу https://docs.microsoft.com/en-us/power-bi/visuals/service-r-visuals.
2. Откройте браузер в виртуальной машине и перейдите по следующей ссылке: https://mran.microsoft.com/open. В виртуальной машине по умолчанию установлены два браузера: *Microsoft Edge* и *Firefox*.
3. Нажмите на кнопку **Download** справа, и вы будете перенаправлены на страницу загрузок. Здесь щелкните по ссылке **Past Releases** справа, которая откроет страницу со всеми предыдущими версиями *Microsoft R Open*. Выберите версию, которую использует Power BI.
4. Нажмите на кнопку **Download** напротив версии для *Windows*.
5. Выполните установку загруженного дистрибутива.
6. Откройте R Studio на виртуальной машине.
7. Откройте меню **Tools** => **Global Options** и убедитесь, что выбрана версия MRO, которую вы только что установили. Если это не так, нажмите на кнопку **Change…** и выберите нужный дистрибутив из списка, после чего нажмите на кнопку **OK**.

Настройка Python на виртуальной машине

Одним из преимуществ использования виртуальной машины является то, что вы получаете предустановленный дистрибутив Python, идеально под-

ходящий для анализа данных. Этот дистрибутив называется *Anaconda*. Он поставляется с более чем 1500 библиотек, популярных в среде анализа данных. Также вместе с ним идет *диспетчер пакетов* (package manager) и *система управления окружением* (environment management system) под названием *conda*. Инсталлировать пакеты предпочтительно именно посредством *conda* по причине установки правильных зависимостей между ними. Система управления окружением *conda* значительно облегчает задачу создания изолированной копии Python с предустановленными библиотеками нужных версий.

Давайте для примеров из этой книги создадим отдельное окружение Python с именем *pbi*. Для этого необходимо выполнить следующие действия.

1. Подключитесь к виртуальной машине.
2. Откройте командную строку, нажав на значок поиска рядом с иконкой *Windows* и введя команду *cmd*.
3. Введите следующую команду для создания окружения *conda* с именем *pbi* на базе Python 3.7:

```
conda create -n pbi python=3.7
```

Решение использовать Python версии 3.7 основывается на информации, полученной из инструкции от Microsoft по адресу https://docs.microsoft.com/en-us/business-applications-release-notes/october18/intelligence-platform/power-bi-service/pervasive-artificial-intelligence-bi/python-service. Согласно документации, службы Power BI совместимы с Python 3.x, так что текущая версия 3.x должна подойти.

Теперь у нас есть окружение Python, которое мы можем использовать для примеров из данной книги.

Настройка SQL Server Machine Learning Services на виртуальной машине

В нескольких примерах из этой книги вам потребуется наличие служб машинного обучения SQL Server (*SQL Server Machine Learning Services – SSMLS*). *SSMLS* предоставляет вам инструменты, позволяющие проводить углубленную аналитику в базах данных с использованием R и Python, а также средства, облегчающие работу с большими данными. Еще в составе *SSMLS* есть несколько предварительно обученных моделей от Microsoft, которыми вы можете пользоваться в процессе учебы и работы. В виртуальной машине по умолчанию запущены службы *SSMLS*. Если вы не используете виртуальную машину, вам необходимо будет вручную запустить эти службы, воспользовавшись подробной инструкцией по адресу https://docs.microsoft.com/en-us/sql/machine-learning/install/sql-machine-learning-services-windows-install?view=sql-server-ver15. Предварительно обученные модели, которые мы будем использовать в этой книге, могут быть добавлены в ваш экземпляр SQL Server поверх базовой установки. Перейдите по следующей ссылке и следуйте инструкциям: https://docs.microsoft.com/en-us/sql/machine-learning/install/sql-pretrained-models-install?view=sql-server-ver15.

Установка пакетов R

Некоторые скрипты на языке R из этой книги могут ссылаться на пакеты, которые у вас изначально могут быть не установлены. Исправить это очень просто. Следующая инструкция в консоли R позволит установить популярный пакет с названием *data.table*:

```
install.packages("data.table")
```

Бывает, что вам требуется установить сразу несколько пакетов за раз. Например, вы хотите одновременно инсталлировать пакеты *data.table* и *dplyr*. Для этого достаточно объединить названия этих пакетов в вектор и присвоить результат переменной *pkgs*. Затем можно передать эту переменную на вход функции *install.packages()*, как показано ниже:

```
pkgs <- c("data.table", "dplyr")
install.packages(pkgs)
```

> **Примечание.** Символьный вектор представляет собой тип данных в языке R для хранения текстовой информации в виде одномерного массива. Вы узнаете больше об этом и других типах данных в R в процессе чтения книги.

При создании визуальных элементов в Power BI при помощи R вам необходимо знать, какую версию пакета использует служба Power BI. Вы можете получить список всех доступных пакетов R в Power BI вместе с их версиями по ссылке https://docs.microsoft.com/en-us/power-bi/service-r-packages-support.

При помощи функции *install.packages()* можно установить последнюю версию пакета из репозитория, который вы используете, если у вас дистрибутив R из *CRAN*. При использовании дистрибутива *Microsoft R Open* будет установлена последняя версия пакета, основываясь на *дате снимка* (snapshot date). В обоих случаях может получиться так, что будет установлен пакет не той версии, с которой работает служба. Чтобы устранить это неудобство, необходимо сначала узнать требуемую версию пакета по ссылке выше, после чего загрузить ее при помощи пакета *devtools*. Ниже приведен пример использования пакета *devtools* для установки пакета *ggplot2* версии 0.9.1 из *CRAN*:

```
library(devtools)
install_version(
    "ggplot2",
    version = "0.9.1",
    repos = "http://cran.us.r-project.org")
```

Установка библиотек Python

Установить библиотеки Python можно разными способами. В данной книге мы будем использовать два метода: при помощи *conda* и при помощи *pip*.

Стоит отметить, что установка библиотек в Python выполняется не так просто, как в R. В книге мы будем в основном использовать командную строку *conda* для установки библиотек Python. Для этого необходимо выполнить следующие действия.

1. Откройте строку поиска, иконка которой расположена рядом со значком *Windows*.
2. Введите слово *Anaconda*, после чего в окне выбора появится вариант *Anaconda Prompt*. Щелкните по нему.
3. Активируйте окружение, которое используете для Power BI, введя следующую команду в командную строку:

   ```
   conda activate "<environment name>"
   ```

   Рекомендуется использовать окружение для разработки на Python, ассоциированное с этой книгой.
4. Установите пакет при помощи *conda*, используя следующий шаблон:

   ```
   conda install <"package name">
   ```

   Например, если вы устанавливаете пакет *pandas*, команда должна иметь следующий вид:

   ```
   conda install pandas
   ```

   Если вам необходимо установить пакет *pandas* версии 1.0.4, используйте следующую команду:

   ```
   conda install pandas=1.0.4
   ```

   Не все пакеты доступны для установки при помощи *conda*. Обратитесь к следующей ссылке для получения списка пакетов, которые могут быть установлены с использованием *conda* в Python 3.6: https://docs.anaconda.com/anaconda/packages/py3.6_win-64. Один из пакетов, который мы будем использовать в этой книге, недоступен в *conda*, но может быть установлен при помощи *PyPI*. Имя этого пакета *CensusData*. Вам необходимо использовать систему управления пакетами *pip* для установки библиотеки *CensusData*, как показано ниже:

   ```
   pip install CensusData
   ```

Настройка Power BI на виртуальной машине

В *Power BI Desktop* необходимо выполнить ряд изменений в настройках, чтобы можно было работать с R и Python. Подробно эти изменения описаны в репозитории кода книги на *GitHub*. Здесь вы также найдете инструкции по созданию и использованию окружения *conda* в Python. Ссылка на репозиторий кода: https://github.com/Apress/adv-analytics-in-power-bi-w-r-and-python.

Альтернативная настройка

Использование виртуальной машины – предпочтительная, но вовсе не обязательная опция для работы с примерами из этой книги. Если вы хотите пойти другим путем, вам придется устанавливать все программное обеспечение вручную. Приводим ссылки на все необходимые установки:

- Power BI: http://www.microsoft.com/en-us/download/details.aspx?id=58494;
- R Studio: https://rstudio.com/products/rstudio/download;
- Microsoft R Open: https://mran.microsoft.com/download;
- Anaconda: http://www.anaconda.com/products/individual;
- VS Code: https://code.visualstudio.com/download;
- SQL Server 2019 Developer: http://www.microsoft.com/en-us/sql-server/sql-server-downloads;
- SQL Server Machine Learning Services: https://docs.microsoft.com/en-us/sql/machine-learning/install/sql-machine-learning-services-windows-install?view=sql-server-ver15.

Если вы остановите выбор на этом варианте, я очень рекомендую использовать виртуальную машину на базе *Windows Server 2016* или выше. Предлагаю ссылку YouTube на пошаговую инструкцию по установке *Windows Server 2019* на *VirtualBox*: http://www.youtube.com/watch?v=ZjQSuyuN0nA&t=8s.

Загрузка пакетов R в SSMLS

В главе 10 вы научитесь работать с моделями машинного обучения посредством служб *SQL Server Machine Learning Services 2019* с использованием языка R. И для этого вам понадобится, чтобы необходимые пакеты были загружены в *SSMLS 2019*. Ниже представлен скрипт на языке T-SQL, который вы можете использовать для вывода информации о том, какие пакеты в данный момент загружены в ваш экземпляр *SSMLS 2019*:

```
EXECUTE sp_execute_external_script
    @language=N'R',
    @script = N'
packagematrix <- installed.packages();
Name <- packagematrix[,1];
Version <- packagematrix[,3];
OutputDataSet <- data.frame(Name, Version);'

WITH RESULT SETS ((PackageName nvarchar(250), PackageVersion nvarchar(max) ))
```

Если пакета, который вам нужен, нет в списке, вам необходимо будет загрузить его вручную. Для этого нужно выполнить следующую пошаговую инструкцию.

Шаг 1. Загрузите пакет sqlmlutils в папку Documents

Пакет *sqlmlutils* можно загрузить по адресу https://github.com/Microsoft/sqlmlutils/tree/master/R/dist.

Скачайте файл zip с репозитория GitHub и сохраните в папке *Documents*.

Шаг 2. Запустите следующий код из командной строки

Откройте командную строку под администратором и запустите следующий код на выполнение:

```
R -e "install.packages('RODBCext', repos='https://cran.microsoft.com')"
R CMD INSTALL %UserProfile%\Documents\sqlmlutils_0.7.1.zip
```

Этот код сработает, если вы предварительно положили архив *sqlmlutils* в папку *Documents* под вашим профилем. Если файл располагается в другом месте, вам необходимо откорректировать путь.

Шаг 3. Загрузите необходимые пакеты

После выполнения второго шага инструкции вы будете готовы к загрузке пакетов в *SSMLS 2019* из скрипта R в R Studio. Ниже приведем фрагмент кода для загрузки пакета *dplyr* в *SSMLS 2019*:

```
library(sqlmlutils)
connection <- connectionInfo(
    server   = "server",
    database = "database",
    uid      = "username",
    pwd      = "password")

sql_install.packages(connectionString = connection,
pkgs = "dplyr", verbose = TRUE, scope = "PUBLIC")
```

Вы можете загружать несколько пакетов одновременно. Скажем, вам необходимо загрузить пакеты *dplyr* и *data.table*. Это можно сделать путем создания символьного вектора, содержащего оба пакета, и передачи его параметру *pkgs*, как показано ниже:

```
library(sqlmlutils)
connection <- connectionInfo(
    server   = "<server>",
    database = "<database>",
    uid      = "<username>",
    pwd      = "<password>")

pkgList <- c("dplyr","data.table")
sql_install.packages(connectionString = connection,
pkgs = pkgList, verbose = TRUE, scope = "PUBLIC")
```

Загрузка необходимых библиотек Python в SSMLS

Как и в случае с загрузкой пакетов R в SQL Server Machine Learning Services 2019, вы должны знать, как установить необходимые пакеты Python при помощи *sqlmlutils* для чтения заключительных глав этой книги. Для этого вам нужно выполнить следующие шаги.

Шаг 1. Скачайте пакет sqlmlutils на свой компьютер в папку Documents

Загрузить пакет *sqlmlutils* можно по ссылке https://github.com/Microsoft/sqlmlutils/tree/master/Python/dist.

Скачайте архив zip и сохраните его в папке *Documents*.

Шаг 2. Откройте командную строку и введите следующие инструкции

```
pip install "pymssql<3.0"
pip install --upgrade --upgrade-strategy only-if-needed c:\temp\sqlmlutils-0.7.2.zip
```

Шаг 3. Загрузите необходимые пакеты

После выполнения шага 2 вы сможете загружать нужные вам пакеты в *SSMLS 2019* посредством запуска скрипта на Python в *VS Code*. Ниже представлен фрагмент кода, позволяющий загрузить библиотеку *pandas* в *SSMLS 2019*:

```
import sqlmlutils
connection = sqlmlutils.ConnectionInfo(
    server="<имя сервера>", database="<база данных>",
    uid="<имя пользователя>", pwd="<пароль>"))
sqlmlutils.SQLPackageManager(connection).install("pandas")
```

Локальный шлюз данных

Локальный шлюз данных (on-premises data gateway) представляет собой инструмент для обеспечения безопасной передачи данных между локальными источниками данных и облаком Azure. Локальный шлюз может быть запущен в двух режимах: персональном и стандартном. Если вы захотите развернуть решения, рассматриваемые в главах с третьей по девятую, в службе Power BI и запускать их с определенной периодичностью, вам понадобится установить локальный шлюз данных в персональном режиме. На момент написания книги скрипты на R и Python, используемые в Power Query, могут быть запущены через локальный шлюз только в персональном режиме. Недостатком

использования шлюза в таком режиме является то, что ваше решение нельзя будет назвать корпоративным, поскольку доступ к нему будет, как ясно из названия режима, только у вас.

Но в главе 10 вы узнаете, как можно использовать R и Python в корпоративном решении на базе Power BI через *SQL Server Machine Learning Services* (*SSMLS*). При использовании служб *SSMLS 2019* ваш код на R и Python будет заключен в специальную хранимую процедуру на языке T-SQL. А хранимые процедуры допустимо использовать в стандартном режиме локального шлюза. В главе 10 вы также познакомитесь с основами рефакторинга кода на R и Python, описанного в предыдущих главах, под специальные хранимые процедуры, используемые в службах SSMLS 2019. Заметьте, что визуальные элементы R не полагаются на локальный шлюз, поскольку они обрабатываются при помощи экземпляра R непосредственно в службе Power BI.

Источники информации

В данной книге описываются разнообразные технологии, и было бы невозможно досконально рассказать о них всех. Понимая это, я решил предоставить вам наиболее полный список литературы для самостоятельного изучения тем, которые вам пока незнакомы. Также я включил ссылку на репозиторий с исходными кодами, используемыми в книге, и дополнил список ссылками на полезные ресурсы.

Репозиторий книги

Исходный код для всех упражнений из книги собран в едином репозитории по адресу https://github.com/Apress/adv-analytics-in-power-bi-w-r-and-python. Полные скрипты на R и Python сгруппированы в хранилище по главам и темам. В репозитории также представлены актуальные источники данных, используемые в примерах, или информация о получении доступа к ним.

Ресурсы по R

Книги:

- *R for Data Science*: прекрасная книга, в которой о языке R рассказывает один из самых плодовитых создателей пакетов Хэдли Уикхэм (Hadley Wickham). В ней в том числе описываются полезные пакеты из авторской библиотеки *tidyverse*. Книга доступна бесплатно по адресу https://r4ds.had.co.nz;
- *An Introduction to Statistical Learning: With Applications in R*: эта книга описывает базовые принципы статистики, без которых не обойтись при изучении машинного обучения с R. Книге уже несколько лет, но она не утратила популярности в сообществе языка R. Часто материалы

из этой книги используются в качестве учебных пособий для курсов во многих колледжах и университетах.

Веб-сайты:

- *RStudio*: на данном портале собраны разнообразные обучающие ресурсы по языку R. Для получения доступа к ним перейдите в меню на вкладку **Resources**, после чего вам будет предложен выбор бесплатных вебинаров, инструкций и книг. Также на этом сайте вы можете скачать последнюю версию рекомендованной среды для работы с языком R – *R Studio*. Адрес сайта: https://rstudio.com;
- *The R Graph Gallery*: на этом сайте представлены визуальные элементы, созданные при помощи R, с полными исходными кодами. Вы можете использовать эти наработки для собственных идей. Адрес сайта: www.r-graph-gallery.com;
- *R Bloggers*: прекрасный сайт для общения в сообществе по языку R. Адрес сайта: www.r-bloggers.com.

Обучение:

- *dplyr tutorial Part 1*: первая из двух частей обучающего видео по пакету *dplyr* от его автора Хэдли Уикхэма. Материал был записан в 2014 году, но до сих пор не утратил своей актуальности. Ссылка на первую часть видео: www.youtube.com/watch?v=8SGif63VW6E;
- *dplyr tutorial Part 2*: вторая часть обучающего видео от Хэдли Уикхэма: www.youtube.com/watch?v=Ue08LVuk790.

Ресурсы по Python

Книги:

- *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*: превосходная книга, освещающая применение принципов машинного обучения и искусственного интеллекта к вашим данным с помощью инструментов Python. Книга доступна для покупки на большинстве ресурсов;
- *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*: эта книга написана автором библиотеки *pandas* Уэсом Маккинни (Wes McKinney). *Pandas* по сей день является наиболее популярной библиотекой для работы с данными в Python. Книга доступна для покупки на большинстве ресурсов.

Видеоблоги, подкасты и курсы:

- *Data School*: это канал в YouTube от Кевина Маркхэма (Kevin Markham), на котором публикуется масса обучающих видео на тему анализа данных в Python при помощи библиотек *pandas*, *matplotlib*, *scikit-learn* и др. Кевин прекрасно справляется с задачей донесения до слушателей сложных тем довольно простым и понятным языком. Адрес канала: www.youtube.com/channel/UCnVzApLJE2ljPZSeQylSEyg;
- *Google's Python Class*: это достаточно старый, но все еще актуальный ресурс введения в Python от Google. В данном обучающем материале прекрасно представлено введение в структуры данных языка и другие

базовые концепции, лежащие в основе любого проекта. Также на сайте присутствует весьма информативный раздел, посвященный регулярным выражениям. Адрес двухдневного курса по Python: https://developers.google.com/edu/python;

- *Talk Python to Me*: очень информативный подкаст, в котором обсуждаются разные области применительно к Python, в том числе и анализ данных. Адрес подкаста: https://talkpython.fm.

Веб-сайты:

- *PEP 8*: одним из главных преимуществ Python в сравнении с другими языками программирования является его доступность и легкость чтения исходного кода. В отличие от большинства языков, Python жестко регламентирует стиль написания кода для облегчения его восприятия в дальнейшем. И именно стилю программирования на Python посвящен этот великолепный сайт, находящийся по адресу https://pep8.org.

Ресурсы по Power BI

Видеоблоги:

- *Guy in a Cube*: однозначно лучший видеоблог по Power BI. На этом канале YouTube освещаются все важнейшие аспекты Power BI, включая администрирование, моделирование данных и визуализацию. Ссылка на канал: www.youtube.com/channel/UCFp1vaKzpfvoGai0vE5VJ0w.

Веб-сайты:

- *SQLBI*: создатели этого сайта (Марко Руссо и Альберто Феррари) являются очень авторитетными авторами книг по языку запросов DAX применительно к Power BI. На их сайте содержится очень много статей, видеоуроков и обучающих материалов по DAX, а также полезные инструменты вроде *DAX Studio*;
- *Tabular Editor*: Tabular Editor – это инструмент с открытым кодом, который должен быть в арсенале любого серьезного разработчика Power BI. В скором времени он должен быть интегрирован в Power BI. Чтобы лучше изучить этот полезный инструмент, перейдите по ссылке, ведущей на его страницу: https://tabulareditor.com.

Книги:

- *The Definitive Guide to DAX* («Подробное руководство по DAX»): эта книга – настоящая библия для тех, кто хочет освоить язык запросов DAX, используемый в Power BI. Книга доступна для покупки в большинстве магазинов (https://dmkpress.com/catalog/computer/data/978-5-97060-859-3/);
- *M Is for Data Monkeys*: в этой книге дано введение в функциональный язык программирования M, используемый в инструменте Power Query. Авторы проделали хорошую работу, рассказав обо всех основных принципах и возможностях преобразования данных доступным языком. Эта книга может быть использована как основа для перехода к более сложным ресурсам по языку M. Книга доступна для покупки в большинстве магазинов.

Подкасты:

- *BIFOCAL*: прекрасный подкаст, позволяющий быть в курсе всего, что происходит в мире Power BI. Найти его можно на популярных платформах подкастинга.

Общие ресурсы

Книги:

- *Data Science for Business*: книга от Фостера Провоста (Foster Provost) и Тома Фосетта (Tom Fawcett), в которой представлены основные принципы науки о данных (data science) строгим языком программиста. Это очень популярная книга в сообществе бизнес-аналитики и может заинтересовать тех, кто ищет способы реализации методов науки о данных в области бизнес-приложений. Книга доступна для покупки во многих магазинах.

Веб-сайты:

- *Data Science Central*: это один из самых популярных сайтов, посвященных науке о данных. Адрес сайта: www.datasciencecentral.com;
- *Kaggle*: сайт изначально задумывался как место для соревнований в области программирования, но сейчас перерос в нечто большее. На этом сайте можно найти большое количество наборов данных, которые удобно использовать при работе с примерами по машинному обучению и искусственному интеллекту. Адрес сайта: www.kaggle.com;
- *ExcelTv*: Microsoft Excel был и в обозримом будущем останется одним из основных инструментов в аналитике данных. Авторы сайта ExcelTv подготовили большое количество обучающих материалов, которые помогут вам стать настоящим профессионалом в Excel. Адрес сайта: https://excel.tv.

Видеоблоги и обучающие сайты:

- *Excel on Fire*: Oz – создатель и ведущий этого видеоблога, – пожалуй, самый яркий и необычный преподаватель Excel и Power Query. За последние несколько лет он записал большое количество полезных видео по искусному преобразованию данных при помощи инструмента Power Query. При этом сами ролики записаны на высшем уровне, а способность ведущего доносить сложные вещи простым и понятным языком просто поражает. Ссылка на канал: www.youtube.com/user/WalrusCandy/featured;
- *Regular Expression Tutorial* от Кори Шафера (Corey Schafer): отличный канал с вводными и продвинутыми уроками по использованию регулярных выражений. Поверьте, после просмотра нескольких видео на этом канале вы не только поймете, что означают все эти мудреные символы в регулярных выражениях, но и поразитесь, насколько полезными они могут быть при решении самых разных задач. Ссылка на видеоблог: www.youtube.com/watch?v=sa-TUpSx1JA.

Подкасты:

- *Analytics on Fire podcast*: это универсальный источник еженедельных мастер-классов по бизнес-аналитике от ведущих специалистов в дан-

ной области. Настраивайтесь на волну каждую неделю, и вы будете получать наслаждение от этого микса из образования и развлечения. Найти подкаст можно на популярных платформах подкастинга;

- *Data Skeptic*: в этом подкасте вопросы из области науки о данных повышенной сложности преподносятся в простой и легкой для усвоения форме с очень увлекательными примерами. Найти подкаст можно на популярных платформах подкастинга;
- *Freakonomics*: изучить технические аспекты науки о данных – это лишь полдела. Очень важно также развить в себе аналитические способности. И этот подкаст, безусловно, поможет вам в этом. Найти подкаст можно на популярных платформах подкастинга;
- *SQL Data Partners*: это очень познавательный и нескучный подкаст, освещающий вопросы, касающиеся *платформы данных Microsoft* (Microsoft Data Platform). Ведущие подкаста – признанные специалисты в этой области, но при этом они много шутят и смеются. Возможно, они будущие комики. :) Их подкаст можно найти на популярных платформах подкастинга;
- *Storytelling with Data*: этот прекрасный подкаст от Коул Нафлик (Cole Knaflic) посвящен различным техникам визуализации данных. Коул также является автором одноименной книги. Найти ее подкаст можно на популярных платформах подкастинга.

Описание глав

- Глава 1 «Грамматика графиков». Вероятно, одним из главных превосходств языка R над Python является его богатая оснащенность средствами визуализации данных. Ведущим пакетом R в области визуализации является *ggplot2*, базирующийся на концепции, известной как *грамматика графиков* (grammar of graphics). В данной главе мы изучим основы графического пакета *ggplot2* и рассмотрим его применение на практике совместно с Power BI.
- Глава 2 «Создание пользовательских визуализаций на R в Power BI при помощи ggplot2». Одно из явных преимуществ пакета *ggplot2* состоит в выразительности, с которой вы можете создавать свои собственные визуализации. В данной главе мы на нескольких примерах продемонстрируем идею выбора типа визуализации на языке R из всех доступных в Power BI при помощи пакета *ggplot2*.
- Глава 3 «Чтение файлов CSV». В этой главе мы рассмотрим концепции применительно к языкам R и Python, позволяющие динамически комбинировать файлы CSV, что с использованием инструмента Power Query было бы достаточно затруднительно.
- Глава 4 «Чтение данных из Microsoft Excel». Здесь мы научимся при помощи R и Python динамически сочетать рабочие листы из нескольких рабочих книг Excel, что в Power Query реализовать бывает непросто.
- Глава 5 «Чтение данных из SQL Server». Из данной главы вы узнаете, как посредством R и Python загружать данные из SQL Server в модель

данных Power BI. Одним из преимуществ такого метода загрузки информации – и мы покажем это в наших примерах – является возможность осуществления логирования.

- Глава 6 «Чтение в модель данных Power BI посредством API». В данной главе вы познакомитесь со способами извлечения данных в Power BI при помощи API с использованием языков R и Python. Посредством Power Query реализовать подобные методы бывает довольно сложно, а иногда просто невозможно.
- Глава 7 «Продвинутые строковые операции и распознавание шаблонов». Из этой главы вы узнаете, как при помощи регулярных выражений в R и Python осуществлять сложные манипуляции со строками. Регулярные выражения – очень мощный инструмент для работы с текстом, и если в R и Python он присутствует по умолчанию, то в Power Query пока нативно не реализован.
- Глава 8 «Вычисляемые столбцы с помощью R и Python». Здесь мы рассмотрим технику создания сложных математических выражений при помощи R и Python. Вы познакомитесь с основами написания математических формул и узнаете, как использовать готовые функции, скрывающие от вас истинную сложность вычислений. В качестве примера мы используем формулу гаверсинуса.
- Глава 9 «Применение методов машинного обучения и искусственного интеллекта к моделям данных Power BI». В девятой главе книги мы обсудим множество тем, включая использование в бизнес-аналитике методов машинного обучения и искусственного интеллекта. Начнем мы с примеров того, как использовать пользовательские модели машинного обучения, реализованные на R и Python, к модели данных Power BI. Затем посмотрим, как можно улучшить модели данных Power BI с применением службы *Microsoft Cognitive Services* без необходимости оформления дорогостоящей подписки на Power BI Premium. При этом мы не будем ограничиваться лишь инструментами от Microsoft и покажем, как можно воспользоваться службой *IBM Watson Natural Language Understanding* для выполнения специфического анализа текста, недоступного в рамках *Microsoft Cognitive Services*.
- Глава 10 «Создание моделей анализа данных и скриптов для обработки информации». В заключительной главе мы посмотрим, как можно воспользоваться языками программирования R и Python в корпоративных решениях, внедренных в Power BI. Показанные методы ориентированы на бесплатные решения, доступные пользователям, у которых уже установлена локальная версия SQL Server версии 2017 и выше.

Теперь, когда все подготовительные мероприятия завершены, вы можете приступать к чтению книги. Инструмент Power BI славится своими богатыми возможностями в области визуализации данных. И начнем мы с того, как можно значительно расширить их при помощи языка R и пакета *ggplot2*.

Часть I

СОЗДАНИЕ ПОЛЬЗОВАТЕЛЬСКОЙ ВИЗУАЛИЗАЦИИ ПРИ ПОМОЩИ R

Глава 1

Грамматика графиков

Визуализация данных во все времена была одной из важнейших составляющих статистики как науки. Именно посредством визуализаций специалисты в этой области могут делиться с окружающими своими изысканиями и при этом не нагружать их сухими цифрами, а говорить на понятном им языке. С учетом того, что язык программирования R был создан аналитиками и для аналитиков, участники сообщества приложили немало усилий, чтобы разработчики могли создавать богатые визуализации, способные помочь им доходчиво рассказывать свои истории о данных.

Наиболее популярным пакетом визуализации данных в R, безусловно, является *ggplot2*. Он был разработан Хэдли Уикхэмом и базируется на концепции, получившей название *многослойная грамматика графиков* (layered grammar of graphics). В рамках этой концепции любой график определяется следующими характеристиками:

- набором данных по умолчанию с привязками переменных к *эстетикам* (aesthetics);
- одним или более слоями, состоящими из геометрического объекта, статистической трансформации, настройки позиции, а также в качестве необязательных элементов из набора данных и привязки эстетик;
- одной *шкалой* (scale) для каждой привязки эстетики;
- системой координат;
- параметрами фасетирования (faceting).

Это определение в полной мере описывает принципы многослойной грамматики графиков. Скоро вы узнаете, как использовать описанную выше концепцию для построения прекрасных визуализаций в Power BI при помощи пакета *ggplot2*.

Несмотря на то что Power BI предоставляет возможности создания визуальных элементов с использованием самых разных библиотек и пакетов Python и R, в данной книге мы главным образом сосредоточимся на применении пакета *ggplot2* и нескольких вспомогательных пакетов. Причина в том, что пакет *ggplot2* находится вне конкуренции в сравнении с другими инструментами из арсенала R и Python. Принципы многослойной грамматики графиков, на которых базируется пакет *ggplot2*, позволяют значительно сократить время от рождения идеи определенной визуализации данных до ее воплощения на практике. Этот пакет обладает богатейшим набором возможностей, так что подавляющее большинство ваших идей в области визуализации может быть легко реализовано при помощи него.

Пошаговое создание визуализации в Power BI при помощи R

В данном разделе будет подробно рассказано о процессе создания пользовательского элемента визуализации в Power BI посредством языка R. Следующие действия вам необходимо будет выполнить вне зависимости от того, какой тип визуализации вы собираетесь использовать.

Шаг 1. Настройте Power BI

Во вводной части книги мы уже говорили о том, как настроить Power BI для работы с R. Убедитесь, что вы выполнили все инструкции и рекомендации по подготовке к работе.

Шаг 2. Перенесите визуальный элемент R в рабочую область Power BI

На вашей панели выбора элементов визуализации справа есть кнопка с латинской буквой «R», показанная на рис. 1.1, при наведении на которую мышью появляется подсказка **Визуальный элемент скрипта R** (R custom visual). Для создания визуализации при помощи этого элемента необходимо нажать на иконку R, в результате чего будет создана заготовка в рабочей области.

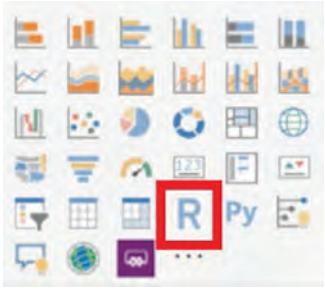

Рис. 1.1 ❖ Иконка создания визуального элемента R в Power BI

Элемент визуализации R предустановлен в Power BI по умолчанию. В рабочей области вы можете изменить его размер и позицию, как и у любого другого визуального элемента в Power BI.

Шаг 3. Определитесь с набором данных

Как и с любым элементом визуализации, в данном случае вам также необходимо определить источник данных. Вы можете собрать набор данных путем переноса полей из вашей модели данных и необязательных мер на панель **Значения** (Values). Полученный в результате набор данных будет доступен в R в виде объекта с именем *data frame*. Вы можете воспринимать этот объект как обычную таблицу Excel с расширенными возможностями для анализа данных.

Одной из особенностей такого объекта, о которой не стоит забывать при создании визуальных элементов R, является обязательная уникальность

строк. При создании объекта *data frame* R автоматически будет проверять его на уникальность строк, и вам необходимо позаботиться о том, чтобы это условие выполнялось в исходном наборе данных.

Шаг 4. Спроектируйте визуальный элемент в среде разработки R

Теперь, когда вы определились с источником данных для своей визуализации, можно приступать к ее проектированию в вашей любимой среде разработки. Когда вы настраивали Power BI для работы со скриптами R, на одном из шагов вы, если помните, указывали предпочтительную среду разработки. Это делалось для того, чтобы вы могли создавать свой элемент визуализации в привычном для вас инструменте вроде *R Studio* вместо того, чтобы пользоваться встроенным в Power BI *редактором скриптов R* (R script editor).

В конечном счете код для вашего визуального элемента должен быть вставлен в редактор скриптов R, показанный на рис. 1.2. Этот редактор открывается в нижней части рабочей области в Power BI при нажатии на кнопку создания визуального элемента скрипта R.

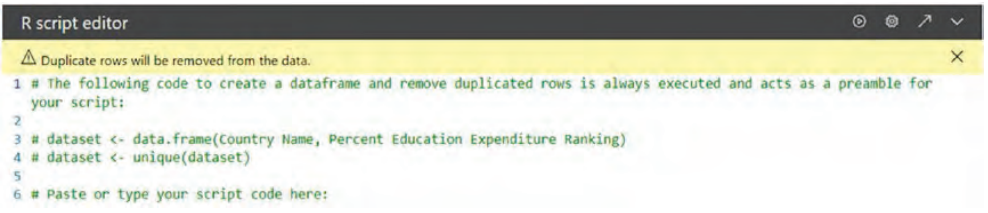

Рис. 1.2 ❖ Редактор скриптов R в Power BI

Как видите, встроенный редактор – не идеальное место для написания скриптов на языке R. Здесь нет *автоматического дополнения ввода* (Intellisense), нет консоли, а пространство для кода весьма и весьма ограничено. К счастью, Power BI позволяет вам портировать набор данных, который вы определили для своего визуального элемента, в вашу среду разработки. Рекомендованной средой является *R Studio*, и в данной книге мы будем обсуждать работу именно в ней.

Чтобы портировать набор данных для вашего пользовательского визуального элемента в *R Studio*, необходимо выбрать элемент в рабочей области и нажать в заголовке редактора скриптов на стрелку, направленную вверх и направо. Это приведет к запуску оболочки *R Studio* с базовым шаблоном скрипта, включающим в себя код с созданием переменной для *датафрейма* (data frame) с именем *dataset* на основе данных, переданных элементу визуализации в Power BI. Также в оболочку будет перенесен весь код, ассоциированный с вашим визуальным элементом. В общем, в Microsoft решили

не изобретать велосипед, а дать нам возможность разрабатывать свои элементы там, где нам удобно. Лично я рекомендую вам проектировать все свои разработки на языке R именно в *R Studio*, а не пользоваться для этой цели встроенным редактором в Power BI.

Шаг 5. Используйте следующий шаблон для разработки элемента на R

Представленный ниже шаблон очень удобно использовать при разработке визуального элемента скрипта R в Power BI:

```
if (<"data test">) {
    #<"Code for R Visual">
} else {
    plot.new()
    title(main = "<predefined message>")
}
```

В первой строке шаблона выполняется проверка набора данных на соответствие всем требованиям визуализации на языке R. Если все в порядке, визуализация создается успешно, в противном случае будет сгенерирован пустой элемент с заранее определенным текстом. За создание пустого элемента отвечает следующий фрагмент кода:

```
plot.new()
title(main = " <predefined message> ")
```

В первой строке создается пустая диаграмма, а во второй к ней добавляется заголовок с заранее определенным сообщением, которое будет показано пользователю.

Шаг 6. Добавьте скрипту функциональности

После написания скрипта для вашего элемента визуализации в сторонней среде разработки перенесите его в редактор скриптов R в Power BI – целиком, за исключением строки с созданием переменной набора данных с именем *dataset*. Этот набор будет автоматически создан в Power BI. Заметьте, что созданный вами элемент визуализации на языке R будет обладать полной интерактивностью и реагировать на изменение фильтров и других визуальных элементов. В следующей главе мы рассмотрим несколько примеров. Единственное, чего вы лишитесь при создании элемента скрипта, – это *двунаправленной фильтрации* (bidirectional filtering). Таким образом, изменение внешних элементов визуализации в Power BI будет распространять свое действие на ваш элемент R, но фильтровать другие элементы непосредственно из созданного вами не получится.

Рекомендованные шаги по созданию визуального элемента на R при помощи ggplot2

В предыдущем разделе мы рассмотрели базовый шаблон, с которого стоит начинать разработку собственного элемента визуализации на языке R для Power BI. По сути, он состоит из проверки набора данных на удовлетворение всем необходимым требованиям и перехвата ошибок. В данной секции мы сосредоточимся на фрагменте кода для создания самого элемента, который в представленном листинге обозначен как `#<»Code for R Visual»>`.

Далее мы рассмотрим, как можно изменить базовый шаблон для построения диаграммы, показанной на рис. 1.3.

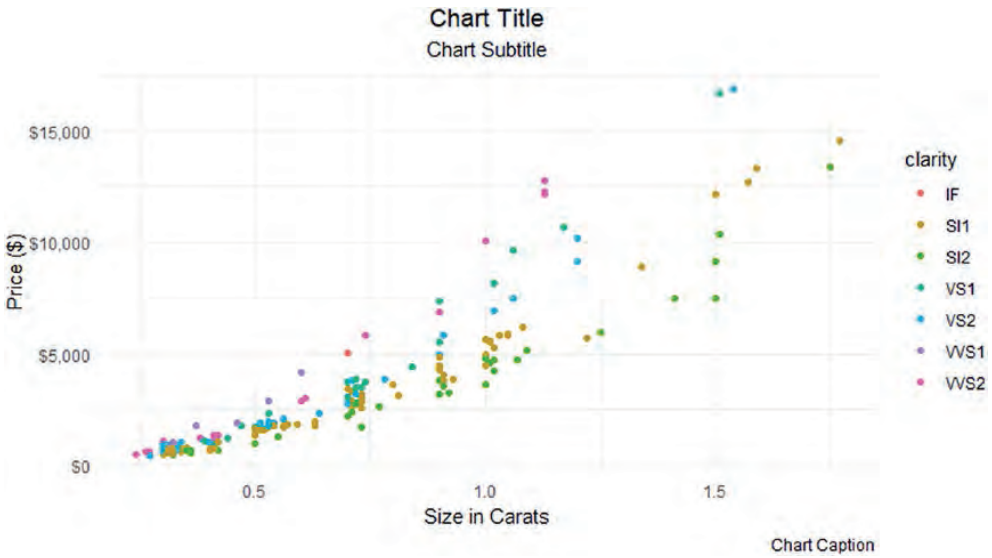

Рис. 1.3 ❖ Визуальный элемент для демонстрации в Power BI

В качестве источника для этой диаграммы используется набор данных об алмазах в части зависимости цены от веса камней в каратах, при этом анализ проводится только по алмазам с цветовой классификацией D. Это довольно подходящий пример для нас, поскольку он позволяет пройти по порядку все шаги при создании элемента визуализации скрипта R. Давайте начнем с самого начала.

Шаг 1. Импортируйте нужные для скрипта пакеты

Для данного примера мы импортируем два пакета, как показано в представленном ниже листинге:

```
library(tidyverse)
library(scales)
```

Мы будем использовать два пакета *ggplot2* и *dplyr* из набора пакетов *tidyverse*.

> **Примечание.** Набор пакетов *tidyverse* объединяет в себе несколько библиотек, значительно облегчающих процесс анализа данных в языке R. Указание на весь набор пакетов (или метапакет) *tidyverse* позволяет загрузить все нужные вам ключевые пакеты одной строкой, вместо того чтобы подгружать каждый пакет по отдельности.

Пакет *ggplot2* мы будем использовать для построения визуализаций, тогда как пакет *dplyr* поможет нам при обработке исходных данных. Загруженный также пакет *scales* поможет нам с форматированием числовых значений.

Шаг 2. Выполните необходимое преобразование исходных данных

Зачастую, перед тем как «скормить» исходные данные пакету *ggplot2*, требуется привести их в удобный для анализа вид. Один из примеров такой трансформации данных приведен ниже:

```
plot.data <-
    diamonds %>%
    filter(color == "D") %>%
    sample_n(200)
```

Набор данных, переданный на вход элементу визуализации R, помимо полезных показателей содержит информацию, не предназначенную для вывода. Так что мы воспользовались пакетом *dplyr*, чтобы очистить исходные данные от лишнего наполнения. Мы знаем, что наша визуализация должна показывать информацию только по алмазам цветовой классификации D. Таким образом, нам необходимо отфильтровать оригинальный датафрейм *diamonds*, чтобы он включал только нужный нам цветовой диапазон. В листинге выше мы делаем именно это в третьей строке кода.

После фильтрации в наборе данных остается информация только об алмазах нужной нам цветовой классификации. Всего о таких алмазах насчитывается 6775 записей, что довольно много для нашей *диаграммы рассеяния* (scatter plot). Вывод на график чересчур большого количества исследований может привести к тому, что визуализация станет совершенно нечитаемой. Чтобы избежать этого, можно случайным образом выбрать из всей информации какое-то ее подмножество и отобразить только его. Подмножество данных будет обладать тем же распределением, что и исходный набор, что позволит отобразить на графике правильные тенденции и при этом сделать его легким для восприятия. В представленном примере мы воспользовались функцией *sample_n()* из пакета *dplyr*, оставив в итоговом наборе всего 200 записей, которые вполне приемлемо смотрятся на диаграмме рассеяния.

Шаг 3. Создайте визуализацию при помощи функции ggplot()

Создание диаграммы при использовании пакета *ggplot2* осуществляется путем вызова функции *ggplot()*. На вход функции подаются параметры, которые будут доступны всем без исключения слоям нашей визуализации. В данном случае мы передали в качестве параметра *data* содержимое датафрейма *plot.data*, а значения переменных, хранящих информацию о весе алмазов и их цене, передали аргументам *x* и *y* функции aes(), которая также поступила на вход функции *ggplot()*, как показано ниже:

```
ggplot(data = plot.data, aes(x = carat, y = price))
```

Очень важно понимать, что собой представляет функция *aes()* и как ее следует использовать. Функция *aes()* описывает правила привязки ваших данных к визуальным атрибутам, или *эстетикам* (aesthetics). Среди распространенных эстетик можно выделить следующие:

- координаты *x* и *y* на выбранном виде диаграммы;
- цвет выбранных вами геометрических фигур;
- размер геометрических фигур;
- цветовое заполнение фигур на диаграмме.

В нашем простом примере функция *aes()* используется для описания того, какие показатели должны быть отмечены на горизонтальной и вертикальной осях. При этом функция *aes()* может быть объявлена как в функции *ggplot()*, как в нашем случае, так и в функции *geom()*, о которой вы узнаете на следующем шаге. Разница в том, что если функция *aes()* объявлена внутри функции *ggplot()*, ее действие будет распространяться на все слои диаграммы, а если указана в функции *geom()*, действие будет ограничиваться конкретным слоем. В следующей главе мы рассмотрим множество примеров применения функции *aes()* на практике.

Шаг 4. Добавьте нужные геометрии

Пример добавления *геометрии* (geom) к визуализации показан в следующем фрагменте кода:

```
ggplot(plot.data, aes(x = carat, y = price)) +
geom_point()
```

При инициализации функции *ggplot()* на предыдущем шаге наша визуализация не содержала ни одного слоя, а значит, и визуализировать ничего не могла. Механизм добавления слоев в функции *ggplot()* реализуется через указание соответствующих функций *geom*. Геометрия, которую мы используем для диаграммы рассеяния, добавляется в функцию *ggplot()* посредством вы-

зова функции *geom_point()*. В представленном выше фрагменте кода функция *geom_point()* наследуется от предшествующей ей функции *ggplot()*, поэтому источник данных и координаты *x* и *y*, обычно обязательные для указания, можно явно не определять. Результирующий график показан на рис. 1.4.

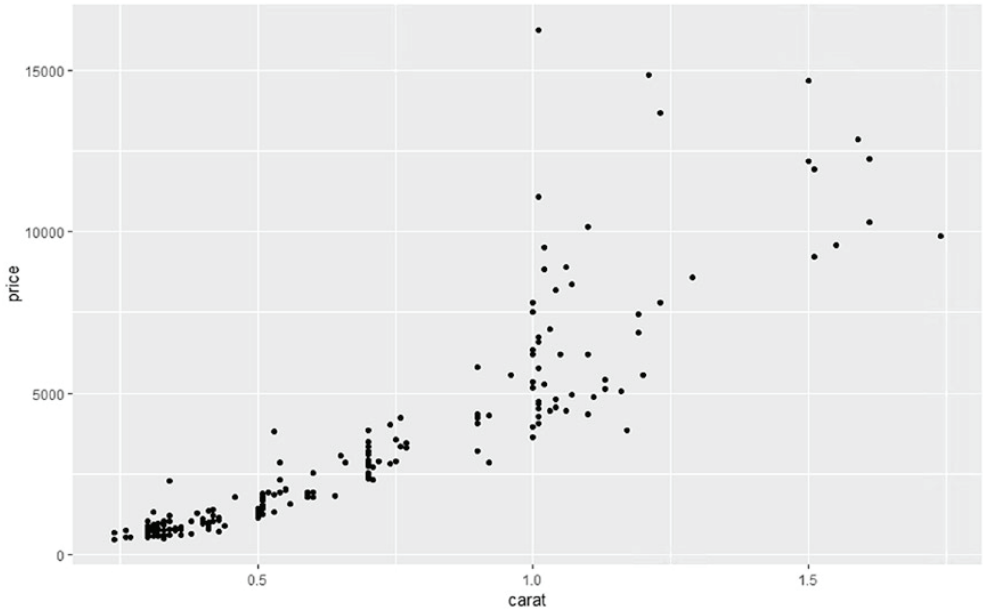

Рис. 1.4 ❖ Базовая диаграмма рассеяния

Теперь нам необходимо добавить еще один слой детализации, а именно мы добавим цветовую дифференциацию по *чистоте* (clarity) алмаза. Для этого необходимо присвоить эстетике *color* значение переменной *clarity* в функции *geom_point()*, как показано ниже:

```
ggplot(plot.data, aes(x = carat, y = price)) +
geom_point(aes(color = clarity))
```

Как видите, мы установили эстетику *color* только в созданной нами геометрии, поскольку она будет использоваться лишь в этом слое.

> **Примечание.** Очень важно помнить, что, когда вы устанавливаете источники данных или эстетики в функции *ggplot()*, они будут распространяться на все слои в вашей визуализации, а когда делаете это в конкретной геометрии, их область доступа будет ограничена только ассоциированным с этой геометрией слоем.

Результирующая диаграмма с параметром *clarity* для каждой точки показана на рис. 1.5.

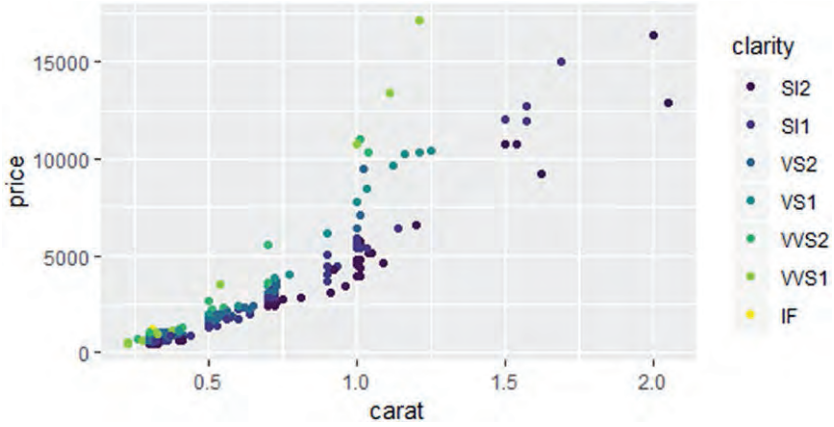

Рис. 1.5 ❖ Базовая диаграмма рассеяния с детализацией

Шаг 5. Определите заголовки, подзаголовки и подписи

Используя следующий код, можно добавить на визуализацию заголовки:

```
ggplot(plot.data, aes(x = carat, y = price)) +
geom_point(aes(color = clarity))+
labs(
    title = "Chart Title",
    subtitle = "Chart Subtitle",
    caption = "Chart Caption"
)
```

Пакет *ggplot2* позволяет очень легко добавлять на диаграммы заголовки, подзаголовки и подписи при помощи функции *labs()*. Название этой функции сокращено от «labels» (метки). При этом титульные надписи на вашей визуализации могут быть как статическими строками, так и динамическими. В следующей главе мы рассмотрим примеры использования динамических заголовков. На рис. 1.6 показано, как будет выглядеть наша диаграмма с заголовками.

Шаг 6. Приведите в порядок оси

Зачастую имена, присвоенные осям *x* и *y* по умолчанию при построении диаграммы, нормально подходят и не требуют каких-то изменений. Но если вам необходимо внести правки в названия осей, придется воспользоваться функциями семейства *scale*.

Давайте вспомним весь код, который мы написали на данный момент:

```
library(tidyverse)
library(scales)
```

```
plot.data <-
    diamonds %>%
    filter(color == "D") %>%
    sample_n(200)

ggplot(plot.data, aes(carat, price)) +
    geom_point(aes(color = clarity)) +
    labs(
        title = "Chart Title",
        subtitle = "Chart Subtitle",
        caption = "Chart Caption"
    )
```

Запуск этого фрагмента кода приведет к построению диаграммы рассеяния, показанной на рис. 1.6. Как видите, на ней осталось несколько надписей, вставленных по умолчанию, которые нам хотелось бы изменить. Список наших будущих изменений:

- название оси *x*;
- название оси *y*;
- способ форматирования чисел на оси *y*.

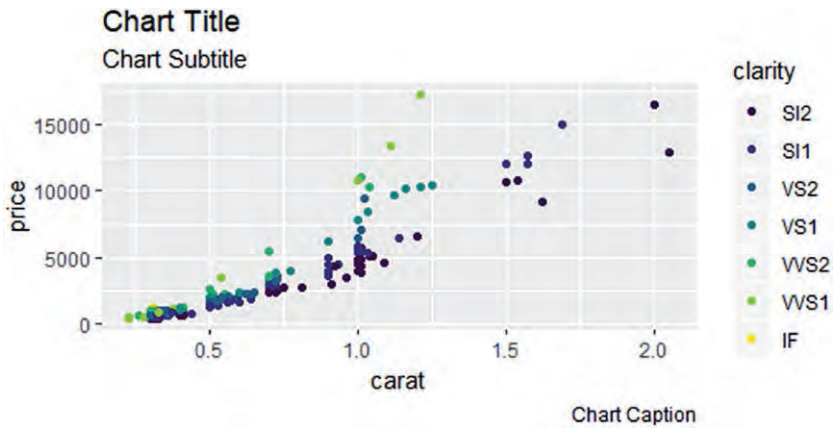

Рис. 1.6 ❖ Визуализация с установленными заголовками и подписями

Эти изменения можно внести, воспользовавшись функциями *scale_x_continuous()* и *scale_y_continuous()*, как показано ниже:

```
library(tidyverse)
library(scales)

plot.data <-
    diamonds %>%
    filter(color == "D") %>%
    sample_n(200)

ggplot(plot.data, aes(x = carat, y = price)) +
    geom_point(aes(color = clarity)) +
```

```
labs(
    title = "Chart Title",
    subtitle = "Chart Subtitle",
    caption = "Chart Caption"
) +
scale_x_continuous(name = "Size in Carats") +
scale_y_continuous(
    name = "Price ($)",
    labels = dollar_format()
)
```

Получившаяся в результате запуска этого кода диаграмма показана на рис. 1.7.

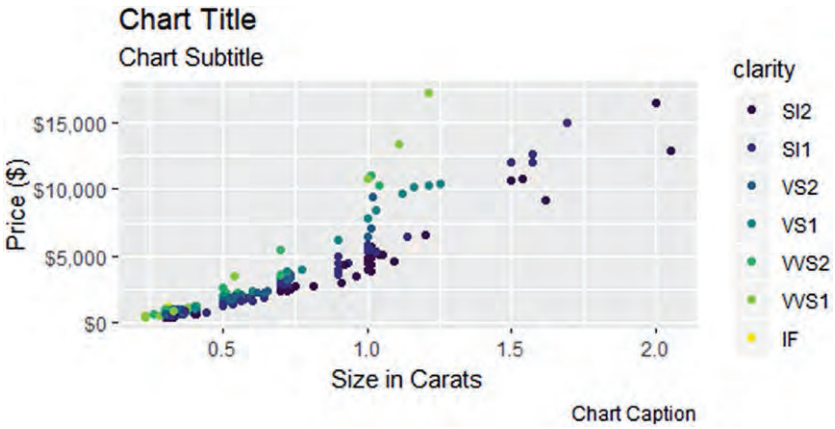

Рис. 1.7 ❖ Диаграмма с настроенными осями

Как видите, при помощи функций *scale_x_continuous()* и *scale_y_continuous()* можно достаточно быстро внести необходимые нам корректировки.

> **Примечание.** Заметьте, что вы добавили шкалы на график так же точно, как до этого добавляли слои. Здесь важно понять, что добавленные шкалы заменяют собой шкалы по умолчанию, но не добавляют к вашей диаграмме слои.

Поскольку оси *x* и *y* базируются на *непрерывных данных* (continuous data), для них будет использоваться именно непрерывная шкала. Единственное, что нам необходимо изменить в отношении оси *x*, – это ее подпись, что, как видите, легко делается путем передачи аргумента *name*. Для оси *y* мы изменим не только подпись, но и способ форматирования числовых меток на оси. Подпись также меняется при помощи аргумента *name*, а для меток предусмотрен свой аргумент с названием *labels*. Мы передали этому аргументу функцию *dollar_format()* из пакета *scales*, что привело к форматированию меток на оси в виде долларов. Функция *dollar_format()* наследует параметр *y* от функции *ggplot()*, которая была определена в начале кода, так что явно объявлять его не нужно.

Шаг 7. Примените тему при необходимости

В пакете *ggplot2* есть предустановленные *темы* (themes), отвечающие за оформление диаграмм в той части, где нет данных. Сюда относится и расположение подписей и заголовков на графике, и цвет фона, и размер шрифтов для заголовков, и многое другое. Применить одну из предустановленных тем к вашему визуальному элементу довольно просто. Темы добавляются так же точно, как устанавливаются слои и модифицируются шкалы. Вы просто ставите знак + (плюс), следом за которым указываете название функции, символизирующей выбранную вами тему, как показано во фрагменте кода ниже:

```
ggplot(plot.data, aes(x = carat, y = price)) +
    geom_point(aes(color = clarity)) +
    labs(
        title = "Chart Title",
        subtitle = "Chart Subtitle",
        caption = "Chart Caption"
    ) +
    scale_x_continuous(name = "Size in Carats") +
    scale_y_continuous(
        name = "Price ($)",
        labels = dollar_format()
    ) +
    theme_minimal()
```

Результат показан на рис. 1.8.

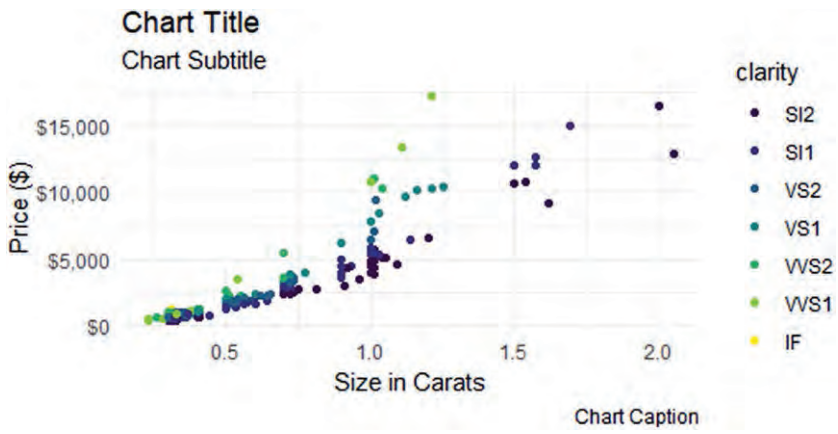

Рис. 1.8 ❖ Установка темы для визуализации

В данном примере мы выбрали предустановленную тему *theme_minimal()*. Считается хорошим тоном снабжать графики дополнительными данными только в случае особой необходимости. Белый фон под диаграммой также выглядит довольно приятно. Тема *theme_minimal()*, как ясно из названия, базируется на основных принципах минимализма. Подробно о темах, предустановленных в пакете *ggplot2*, мы поговорим далее в этой главе.

Шаг 8. Используйте функцию theme() для настройки оформления

Если вы остались не полностью удовлетворены примененной к визуализации предустановленной темой, то можете провести ее тонкую настройку при помощи специальной функции *theme()*. В следующем фрагменте кода мы вызовем эту функцию с целью форматирования заголовков:

```
ggplot(plot.data, aes(carat, price)) +
    geom_point(aes(color = clarity)) +
    labs(
        title = "Chart Title",
        subtitle = "Chart Subtitle",
        caption = "Chart Caption"
    ) +
    scale_x_continuous(name = "Size in Carats") +
    scale_y_continuous(
        name = "Price ($)",
        labels = dollar_format()
    ) +
    theme_minimal() +
    theme(
        plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        plot.caption = element_text(hjust = 1)
    )
```

Результат запуска этого кода показан на рис. 1.9.

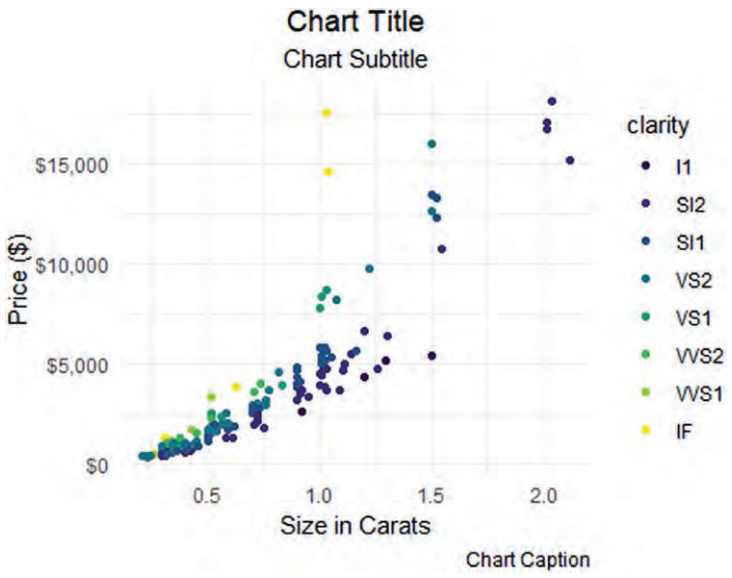

Рис. 1.9 ❖ Смещение заголовков и подписей с помощью функции *theme()*

Функция *theme()* открывает вам доступ к форматированию более чем 80 различных элементов на графике. Для большинства этих элементов существуют функции семейства *element*, предназначенные специально для выполнения подобных изменений.

В представленном выше фрагменте кода мы изменили положение заголовка диаграммы, подзаголовка и подписи. Как видите, элементу, ассоциированному с заголовком графика, соответствует объект *plot.title*, к подзаголовку мы обращаемся как к *plot.subtitle*, а к нижней подписи – как к *plot.caption*. При этом изменение всех трех элементов выполняется при помощи функции *element_text()*. Заголовок и подзаголовок графика были центрированы путем передачи аргументу *hjust* функции *element_text()*, отвечающему за горизонтальное позиционирование элемента, значения 0.5. Этот аргумент может принимать значения в диапазоне от 0 до 1, где 0 означает расположение элемента по левому краю, 0.5 – по центру, а 1 – по правому краю.

Ниже представлены функции семейства *element* со всеми своими аргументами:

```
element_blank()

element_rect(fill = NULL, colour = NULL, size = NULL, linetype = NULL, color = NULL,
inherit.blank = FALSE)

element_line(colour = NULL, size = NULL, linetype = NULL, lineend = NULL, color = NULL,
arrow = NULL, inherit.blank = FALSE)

element_text(family = NULL, face = NULL, colour = NULL, size = NULL, hjust = NULL, vjust =
NULL, angle = NULL, lineheight = NULL, color = NULL, margin = NULL, debug = NULL, inherit.
blank = FALSE)
```

По адресу https://ggplot2.tidyverse.org/reference/element.html можно найти полную документацию по функции *theme()*.

Дополнительный шаг: задайте цвета точек на диаграмме рассеяния

Иногда вам будет необходимо, чтобы все точки на диаграмме рассеяния были одного цвета. Давайте посмотрим, что будет, если присвоить эстетике *color* конкретный цвет, а не привязывать ее к определенному полю в наборе данных. Во фрагменте кода ниже мы присвоим значение *blue* этой эстетике в функции *aes()*:

```
library(tidyverse)
library(scales)

plot.data <-
    diamonds %>%
    filter(color == "D") %>%
    sample_n(200)
```

```
ggplot(plot.data, aes(carat, price)) +
    geom_point(aes(color = "blue")) +
    labs(
        title = "Chart Title",
        subtitle = "Chart Subtitle",
        caption = "Chart Caption"
    ) +
    scale_x_continuous(name = "Size in Carats") +
    scale_y_continuous(
        name = "Price ($)",
        labels = dollar_format()
    ) +
    theme_minimal() +
    theme(
        plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        plot.caption = element_text(hjust = 1)
    )
```

Результирующая диаграмма показана на рис. 1.10.

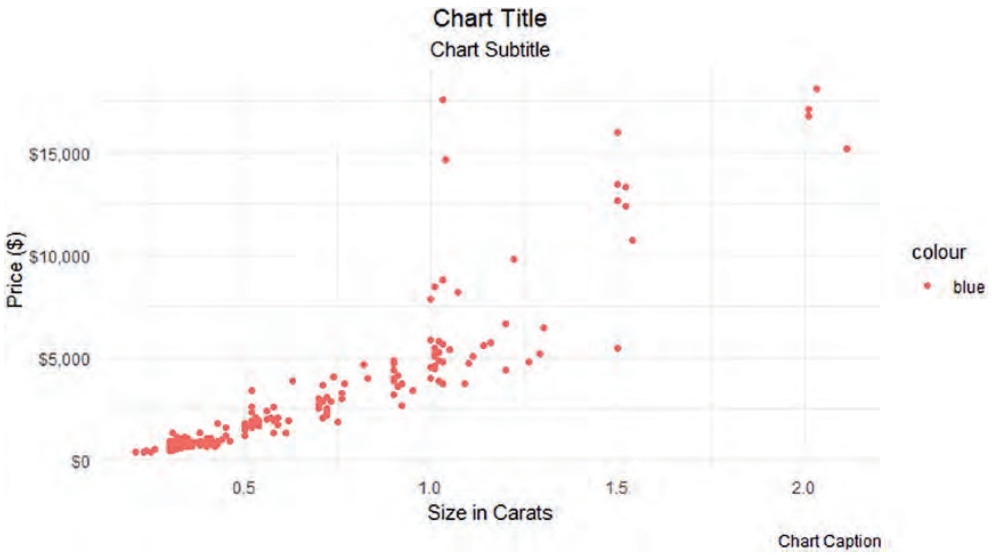

Рис. 1.10 ❖ Установка эстетики *color* в функции *aes()*

Заметьте, что мы не добились желаемого результата. Точки на нашей диаграмме рассеяния получились не синего цвета, как мы планировали, а красного. Причина в том, что мы попытались присвоить цвет точкам внутри функции *aes()*. Если вы хотите присвоить цвету конкретное значение, делать это необходимо за пределами функции *aes()*, как показано в исправленном листинге ниже:

```
library(tidyverse)
library(scales)

plot.data <-
    diamonds %>%
    filter(color == "D") %>%
    sample_n(200)

ggplot(plot.data, aes(carat, price)) +
    geom_point(color = "blue") +
    labs(
        title = "Chart Title",
        subtitle = "Chart Subtitle",
        caption = "Chart Caption"
    ) +
    scale_x_continuous(name = "Size in Carats") +
    scale_y_continuous(
        name = "Price ($)",
        labels = dollar_format()
    ) +
    theme_minimal() +
    theme(
        plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        plot.caption = element_text(hjust = 1)
    )
```

Теперь мы получим ожидаемый результат с синими точками по диаграмме рассеяния, как видно на рис. 1.11.

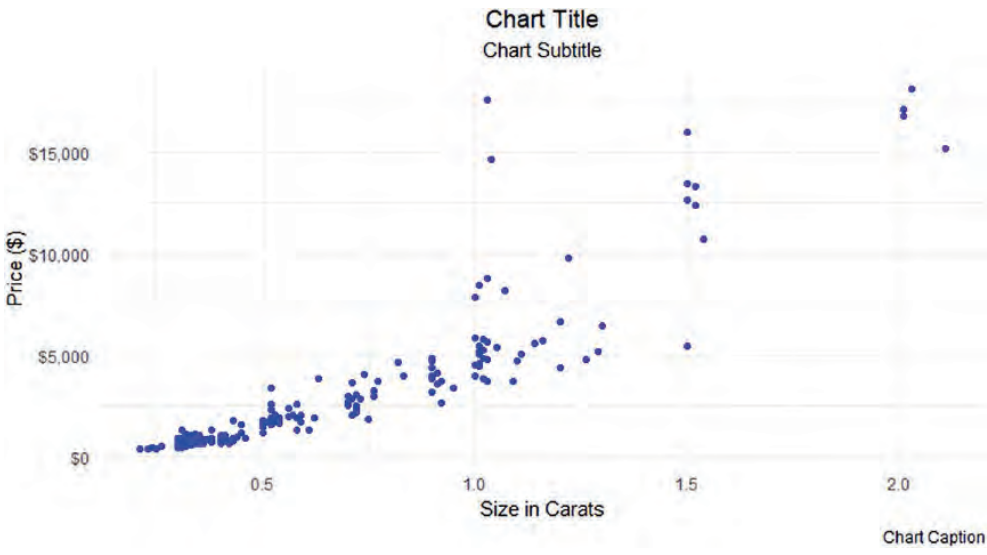

Рис. 1.11 ❖ Определение цвета точек за пределами функции *aes()*