

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

С. В. Рындина

БИЗНЕС-АНАЛИТИКА: ВИЗУАЛИЗАЦИЯ ДАННЫХ

Учебно-методическое пособие

ПЕНЗА 2018

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Пензенский государственный университет» (ПГУ)

С. В. Рындина

Бизнес-аналитика: визуализация данных

Учебно-методическое пособие

Пенза
Издательство ПГУ
2018

УДК 004.6
Р93

Р е ц е н з е н т
кандидат технических наук, доцент
А. А. Масленников

Рындина, С. В.

Р93 Бизнес-аналитика: визуализация данных : учеб.-метод. пособие / С. В. Рындина. – Пенза : Изд-во ПГУ, 2018. – 44 с.

Рассмотрены возможности языка программирования R для анализа данных, акцент сделан на базовых графических возможностях и пакете ggplot2. Приведены задания для лабораторных работ по визуализации данных. Материал пособия соответствует программе дисциплины «Бизнес-аналитика на основе больших данных», но может быть использован также при изучении дисциплины «Бизнес-прогнозирование» и при написании выпускной работы бакалавра.

Издание подготовлено на кафедре «Экономическая кибернетика» ПГУ и предназначено для обучающихся по направлению подготовки 38.03.05 «Бизнес-информатика».

УДК 004.6

© Пензенский государственный
университет, 2018

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	4
1 РОЛЬ ДАННЫХ В БИЗНЕС-АНАЛИТИКЕ	5
1.1 Business analysis body of knowledge (BABOK).....	5
1.2 Наборы данных для анализа. Открытые данные	6
1.3 Язык программирования R. Среда разработки RStudio	8
1.4 Типы данных в R	18
1.5 Импорт данных в R	21
2 БАЗОВЫЕ ВОЗМОЖНОСТИ ЯЗЫКА R ДЛЯ ВИЗУАЛИЗАЦИИ ДАННЫХ.....	26
2.1 Функция plot().....	26
2.2 Функция boxplot().....	31
2.3 Функция mosaicplot()	34
3 ВОЗМОЖНОСТИ ПАКЕТА ggplot2	36
4 ЛАБОРАТОРНЫЕ РАБОТЫ	41
Лабораторная работа № 1. «Визуализация данных: базовая графика»	41
Лабораторная работа № 2. «Визуализация данных: пакет ggplot2»	41
БИБЛИОГРАФИЧЕСКИЙ СПИСОК.....	43

ВВЕДЕНИЕ

В соответствии с учебным планом студенты третьего курса направления подготовки 38.03.05 «Бизнес-информатика» изучают дисциплину «Бизнес-аналитика на основе больших данных».

Визуализация и разведочный анализ данных – базовые методы бизнес-аналитики, основанной на данных, вне зависимости от их объема.

Целью учебно-методического пособия является рассмотрение графических методов представления данных, их возможностей для анализа данных и извлечения знаний.

1 РОЛЬ ДАННЫХ В БИЗНЕС-АНАЛИТИКЕ

1.1 Business analysis body of knowledge (BABOK)

В текущей практике, если опираться на BABOK, бизнес-анализ и бизнес-анализ в IT выступают как синонимичные понятия.

Задачи бизнес-аналитики:

- выявление потребностей;
- обоснование изменений;
- разработка решений.

В третьей версии BABOK акцент сделан на изменениях. Улучшения в организации реализуются за счет контролируемых изменений, которые удовлетворяют потребностям заинтересованных лиц и позволяют достигать поставленных целей. Центральные идеи стандарта BABOK – шесть ключевых понятий, которые необходимо учитывать при бизнес-анализе в организации (рисунок 1.1).

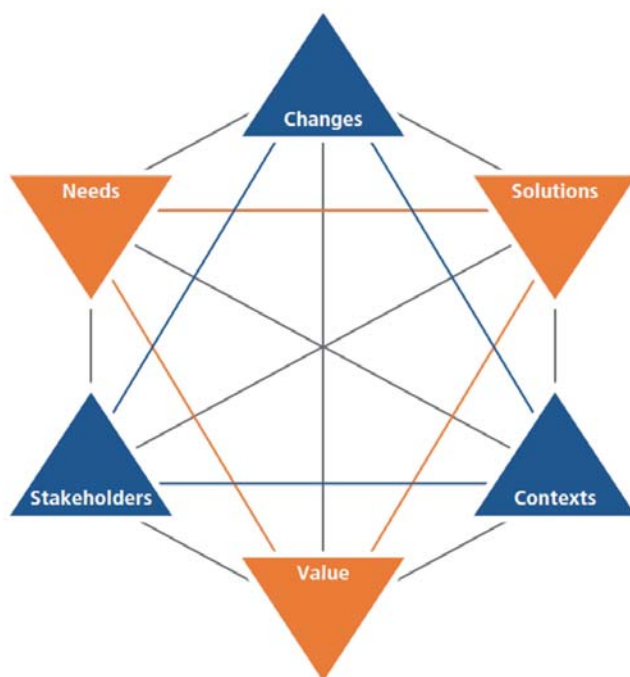


Рисунок 1.1 – Ключевые понятия BABOK

Во главу модели поставлены «*changes*» – «изменения», т.е. преобразования деятельности организации для повышения производительности (эффективности).

Изменения вызываются «*needs*» – «потребностями» – это проблемы и возможности, которые мотивируют действовать заинтересованных лиц, изменяют ценность продукта (услуги), разрушая или повышая ее.

Потребности требуют «*solution*» – «решения», выбора способа действовать, удовлетворяющего потребностям заинтересованных сторон и учитывающего контекст.

Изменения инициируются «*stakeholder*» – «заинтересованной стороной» на основе имеющихся потребностей, влияющих на выбор решения и воздействующих на изменения.

Критерием отбора решений и проведения изменений выступает «*value*» – «ценность», потенциальная или реализованная стоимость, значимость или полезность для удовлетворения потребностей заинтересованных сторон в конкретной ситуации (контексте).

Выбор решений и изменения учитывают «*context*» – «контекст», обстоятельства окружающей среды, которые влияют на понимание решений и изменений.

ВABOK выделяет следующие стадии анализа:

- понять текущее состояние КАК ЕСТЬ;
- определить будущее состояние КАК ДОЛЖНО БЫТЬ;
- определить действия по переходу от «как есть» к «как должно быть».

Бизнес-анализ напрямую связан с бизнес-архитектурой компании, бизнес-процессами и IT-решениями для их поддержки. Одной из важных составляющих бизнес-анализа является анализ данных. Управление чем-либо возможно на основе измеримых показателей, которые как раз и позволяют на основе исторических данных понять – как есть, определить как должно быть и выработать программу планового изменения показателей на основе выявленных зависимостей между ними для достижения этих значений.

Благодаря текущему мониторингу данных можно отслеживать эффективность принятых решений и корректировать программу изменений, своевременно реагируя на динамично меняющийся контекст.

В ВABOK определяются пятьдесят методов бизнес-анализа, но в бизнес-аналитике на основе данных акцент будет сделан на четырех из них:

- Data Mining («Интеллектуальный анализ данных»);
- Data Modelling («Моделирование данных»);
- Data Dictionary («Словарь данных»);
- Data Flow Diagrams («Диаграммы потоков данных»).

1.2 Наборы данных для анализа. Открытые данные

Для выработки решений необходимо знать контекст. В отслеживании текущего контекста эффективны BI-системы (англ. *business*

intelligence), которые обрабатывают большие массивы данных, регистрируемых в цифровой форме.

На сегодняшний день существует тенденция бесплатного и свободного раскрытия цифровых данных как со стороны государства, так и со стороны бизнеса.

Организуются конкурсы или хакатоны по анализу данных (*hackathon* – первоначально соревнование разработчиков-программистов, на текущий момент любых разработчиков, связанных с цифровыми решениями):

- Kaggle – площадка, на которой различные компании проводят спортивные соревнования по анализу данных (построение прогнозных моделей на близких к реальным данным) <https://www.kaggle.com/>;

- «Лаборатория Касперского» проводит хакатоны по анализу данных, посвященные индустриальной безопасности;

- хакатоны от «Сбербанка» по blockchain, machine learning, big data;

- холдинг «Газпром-Медиа» проводит свой первый хакатон Hack The Media в 2018 г., цель которого найти новые идеи и решения по обработке огромного массива медиаданных.

И это только малая часть возможностей попробовать свои силы в анализе реальных данных, получить практический опыт использования современных инструментов и платформ анализа.

В реальном бизнесе данные составляют значительную часть конкурентных преимуществ, а значит, являются коммерческой тайной. Такие данные не будут публичными и результаты обработки таких данных останутся собственностью компании с ограниченным доступом.

Для учебных целей в бизнес-аналитике используются публичные данные. В них включаются открытые данные и разделяемые данные.

Существует масса способов объявления статуса данных и открытой информации: при распространении открытого программного кода часто используются лицензии GPL (General Public License), LGPL, лицензия FreeBSD, лицензия Apache. Для данных разработаны специальные лицензии, одна из этих лицензий Open Database Commons (ODBC) разработана Open Knowledge Foundation и лицензия, которая сейчас становится все более общепринятой, это Creative Commons Zero, это последняя версия лицензии Creative Commons, которая фактически позволяет использовать данные для любых целей, в том числе коммерческих.

Последний вариант помогает компании также в налаживании социальных связей, завоевании репутации и формировании канала подбора сотрудников, которые до начала официального найма доказали свою полезность компании.

Данные, которые публикуют коммерческие организации, удовлетворяют обычно не всем условиям открытых данных. Такие данные называют «разделяемые».

Публикатор разделяемых данных не обязан делиться всеми подробностями получения данных. Однако он позволяет воспользоваться накопленным объемом цифровых данных третьей стороне, не неся никакой ответственности за то, что эти данные могут оказаться не действительными, некорректными. Публикатор не обязан поддерживать данные в актуальном состоянии.

Существует несколько площадок, на которых известные компании публикуют кейсы вместе с данными для их решения. Такой площадкой является, например, <https://www.kaggle.com>.

Сайт «Открытые данные в России» предоставляет доступ к аннотированным наборам данных, многие из которых удовлетворяют условиям открытых данных (<http://data.gov.ru/opendata>).

Сайт Евростата (Eurostat) предоставляет доступ к очень обширной статистике по странам Европы и регионам (<http://ec.europa.eu/eurostat/data/database>).

Сайт Международного банка (World Bank) предоставляет доступ к данным по различным странам (<http://databank.worldbank.org/data/home.aspx>).

Сайт Организации экономического сотрудничества и развития (Organisation for Economic Cooperation and Development – OECD) публикует данные об экономических и демографических показателях, входящих в нее развитых стран (stats.oecd.org).

Сайт «Европейское социальное исследование» (European Social Survey ESS) предоставляет доступ к данным сравнительного исследования изменения установок, взглядов, ценностей и поведения населения Европы, для анализа можно использовать данные восьми волн 2002, 2004, 2006, 2008, 2010, 2012, 2014 и 2016 гг. (<http://www.europeansocialsurvey.org/data/>).

1.3 Язык программирования R. Среда разработки RStudio

Для анализа данных будем использовать язык R, который наряду с Python активно используется для решения задач анализа данных.

В учебно-методическом пособии рассматривается версия R version 3.4.2 на 2017-09-28. Но изменения происходят достаточно оперативно и на 2018-03-15 актуальна version 3.4.4.

Для установки необходимо загрузить последнюю версию R с сайта <https://cran.r-project.org/> (имеются установщики для всех ОС).

Версия R для Windows поставляется с графической оболочкой RGui (рисунок 1.2).

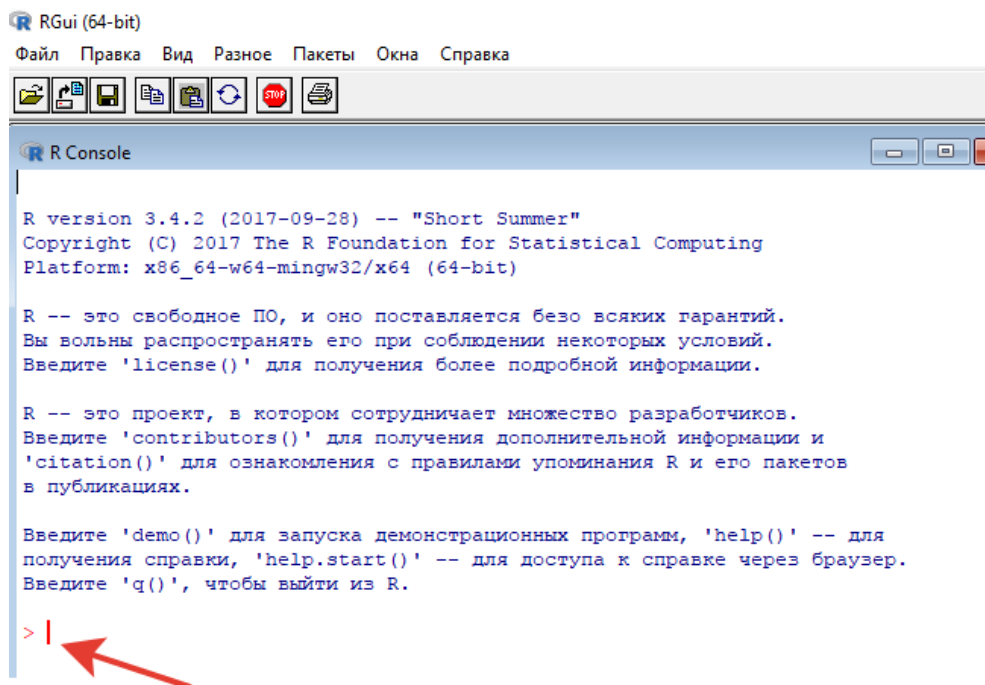


Рисунок 1.2 – Графический интерфейс RGui
(окно консоли с приглашением «>» для набора кода)

Графический интерфейс поддерживает три вида окон:

- консоль, в которой после ввода кода (клавиша Enter) он сразу передается на исполнение;
- редактор скриптов (открывается с помощью меню Файл > Новый скрипт), в котором код набирается, редактируется и при необходимости сохраняется с расширением .R, на исполнение код можно посылать построчно или блоками, предварительно выделив;
- графические окна, в которых строятся и сохраняются графические объекты.

В базовой установке R имеется некоторое количество предустановленных пакетов. Просмотреть все установленные пакеты можно с помощью функции

```
library()
```

дополнительные пакеты находятся по адресу: <https://cran.r-project.org/>.

Замечание: код, который можно набирать в консоли или скрипте и запускать на исполнение будет набираться шрифтом Calibri на сером фоне.

Параллельно с R рекомендуется установить среду разработки (IDE) RStudio, дистрибутив которой можно загрузить по адресу: <https://www.rstudio.com/>.

Как и язык R, эта среда относится к open source.

Начальные сведения для работы в RStudio. Начальное окно среды разработки представлено на рисунке 1.3:

- Source editor and data view (редактор кода и обзор данных);
- R console (консоль);
- служебная панель 1: Environment, history and connections окружение (данные, функции, история команд и доступные внешние ресурсы);
- служебная панель 2: Files, plots, packages, help and viewer (файлы, графики, пакеты, помощь и просмотрщик).

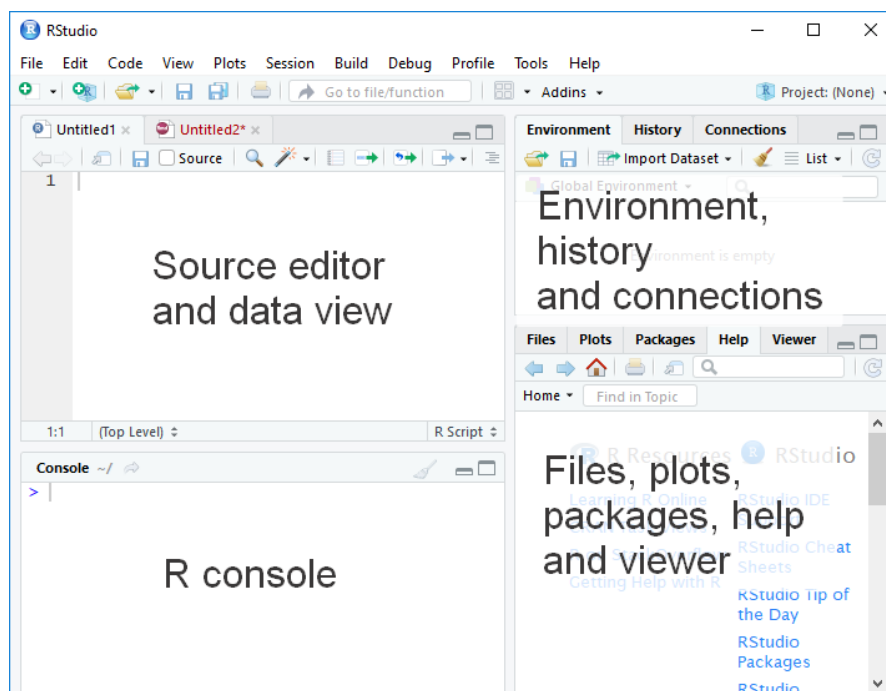



Рисунок 1.3 – IDE RStudio

В меню **Tools > Global options...** можно настроить расположение панелей на вкладке **Pane Layout**: панели **Source**, **Console** только поменять местами, а вот вкладки на служебных панелях можно перемещать с одной панели на другую по желанию (рисунок 1.4).

Код предпочтительнее набирать в панели **Source**, открыв новый файл с помощью меню **File > New file > R script**. На выполнение код посылается комбинацией клавиш **CTRL+ENTER** или кнопкой в правом верхнем углу панели  **Run**.

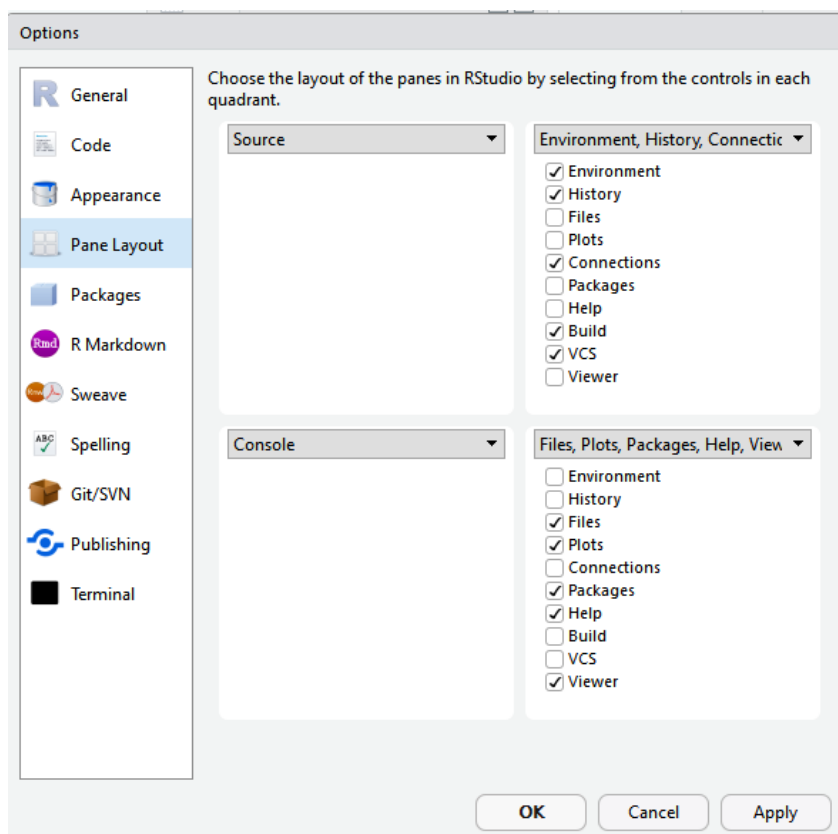


Рисунок 1.4 – Настройка панелей

RStudio поддерживает автоматическое завершение кода в окне **Source** при помощи клавиши TAB (рисунок 1.5). Такие подсказки можно получить и при наборе аргументов, если используются ранее определенные переменные.

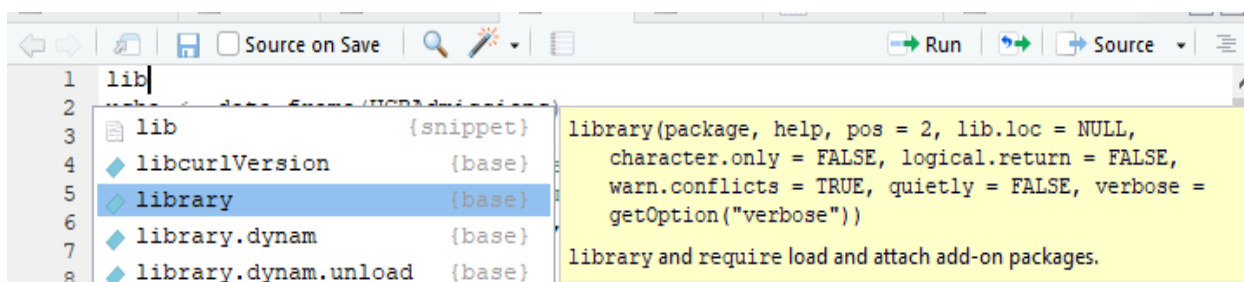



Рисунок 1.5 – Завершение функции library() с помощью клавиши TAB

Горячие клавиши

- CTRL + 1 – перемещает курсор в окно Source;
- CTRL + 2 – перемещает курсор в Console;
- CTRL + L – очищает окно Console от текста;
- ESC – прерывает вычисления.

Открыть существующий файл можно с помощью **File > Open** или кнопки . Открыть файл, с которым недавно работали, можно с помощью **File > Recent Files** или кнопки раскрыть список на панели инструментов (рисунок 1.6).

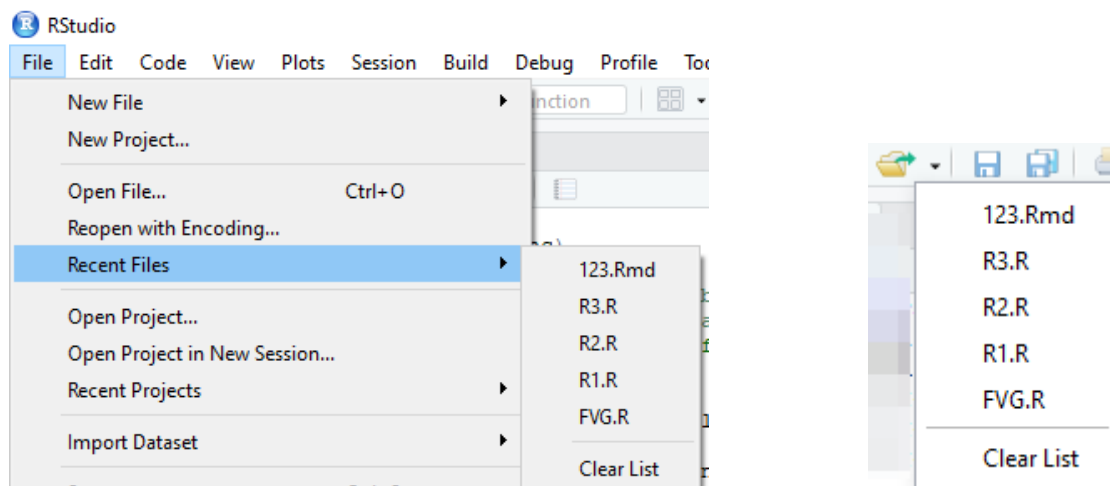



Рисунок 1.6 – Список недавно использовавшихся файлов

В окне **Source** есть возможность найти и заменить необходимые части текста: с помощью **Edit > Find and Replace** или кнопки . Объект, внесенный в поле для поиска, будет подсвечен в коде. В соседнем окне Replace можно ввести замену (рисунок 1.7).

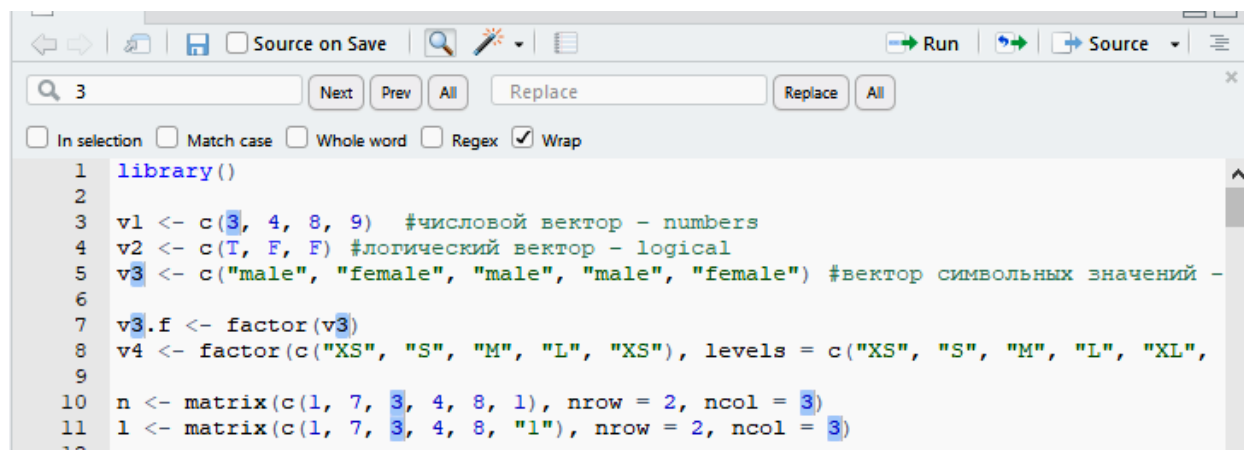


Рисунок 1.7 – Поиск и замена кода

При закрытии RStudio программа предлагает сохранить код в открытых файлах текущей сессии и рабочее пространство (Workspace), которое включает в себя созданные переменные, историю команд (рисунок 1.8).

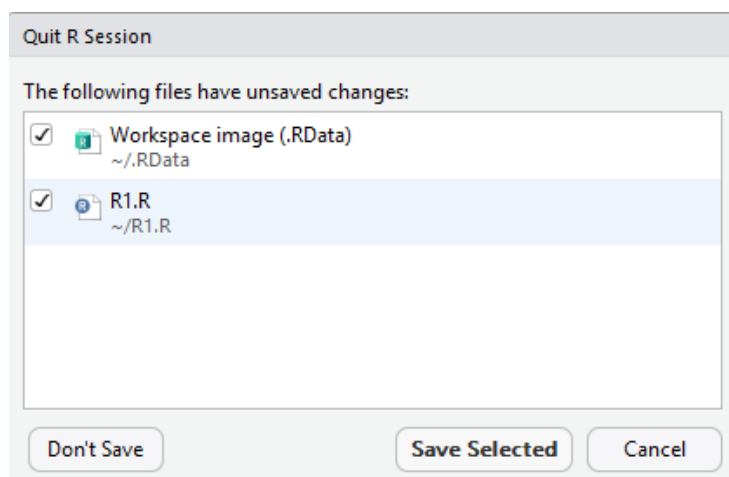


Рисунок 1.8 – Завершение работы RStudio

При открытии RStudio загружается сохраненное рабочее пространство со всеми незакрытыми в прошлой сохраненной сессии файлами, созданными переменными, функциями и историей команд.

Вкладка **History** (рисунок 1.9), которую можно подключить в одну из служебных панелей, содержит журнал всех ранее выполненных в рабочем пространстве команд (в окне **Console**).

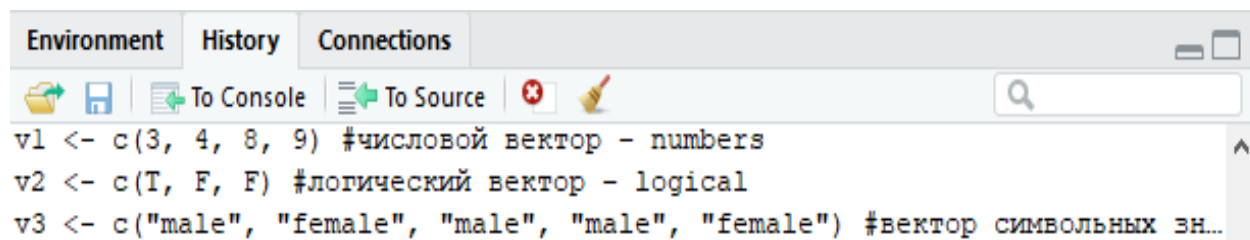




Рисунок 1.9 – История команд

Команды сохраняются и после очистки окна **Console**. Эти команды можно передавать на выполнение в **Console** кнопкой , они выполняются после нажатия ENTER, или скопировать в открытый файл в окне **Source** (если открытого файла нет, то создастся новый) кнопкой . Для работы с несколькими командами их нужно предварительно выделить.

Вкладка **Environment** (рисунок 1.10), которую можно подключить в одну из служебных панелей, содержит описание всех объектов (это могут быть пользовательские функции, переменные), которые были созданы ранее в рабочем пространстве.

Global Environment	
ucba	24 obs. of 4 variables
wdata	400 obs. of 2 variables
Values	
v1	num [1:4] 3 4 8 9
v2	logi [1:3] TRUE FALSE FALSE
v3	chr [1:5] "male" "female" "male" "male" "female"

Рисунок 1.10 – Завершение функции library() с помощью клавиши TAB

Текущую рабочую директорию (папку) можно проверить с помощью функции

getwd()

Файлы текущей рабочей директории отображаются в окне вкладки Files (рисунок 1.11)

Name	Size	Modified
.RData	600.1 KB	May 27, 2018, 10:18 AM
.Rhistory	10.3 KB	May 27, 2018, 10:23 AM

Рисунок 1.11 – Файлы текущей рабочей директории

Текущую рабочую папку можно поменять, например, с помощью кнопки на вкладке Files (рисунок 1.12) или с помощью меню Tools > Change Working Dir...

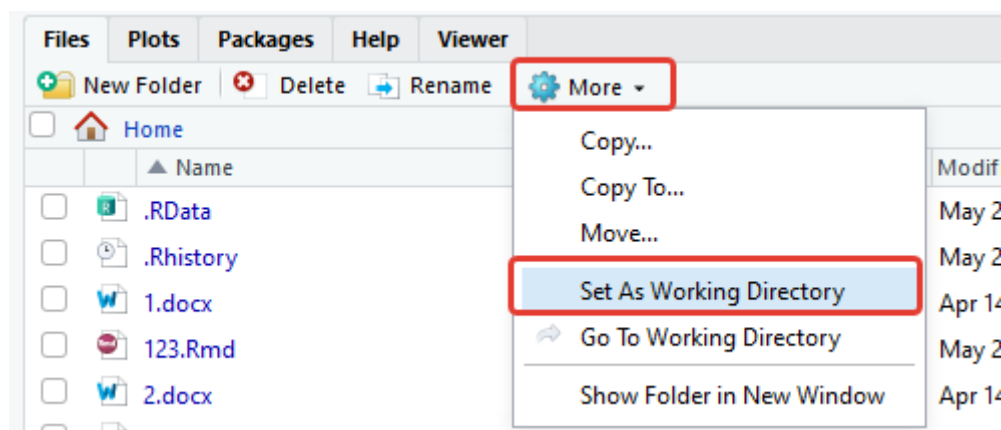


Рисунок 1.12 – Смена рабочей папки

Возможные последствия изменения рабочей папки:

- имеющиеся в коде относительные ссылки на файлы станут недействительными;
- при завершении работы файл .RData будет сохранен в новую папку.

Установка новых пакетов, необходимых в данной рабочей сессии, возможна на вкладке **Packages** с помощью кнопки **Install** (рисунок 1.13). При наборе имени пакета подключается автозаполнение.

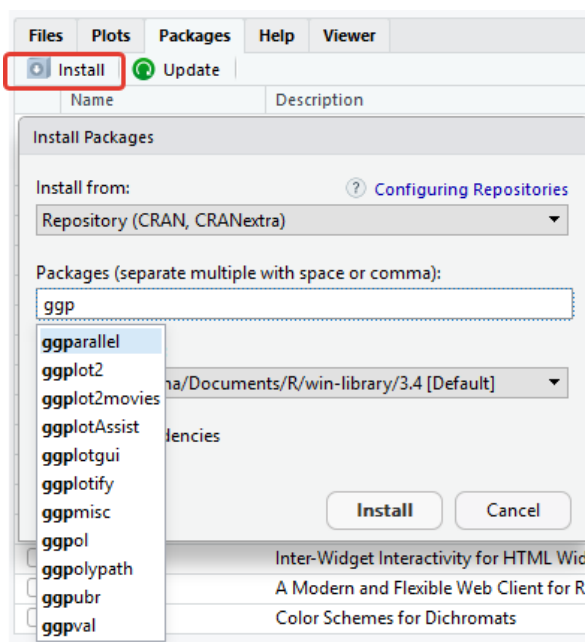


Рисунок 1.13 – Установка пакетов

Флаг на опции **Install dependencies** означает, что установятся и все связанные пакеты, функции которых использует нужный пакет (рисунок 1.14).

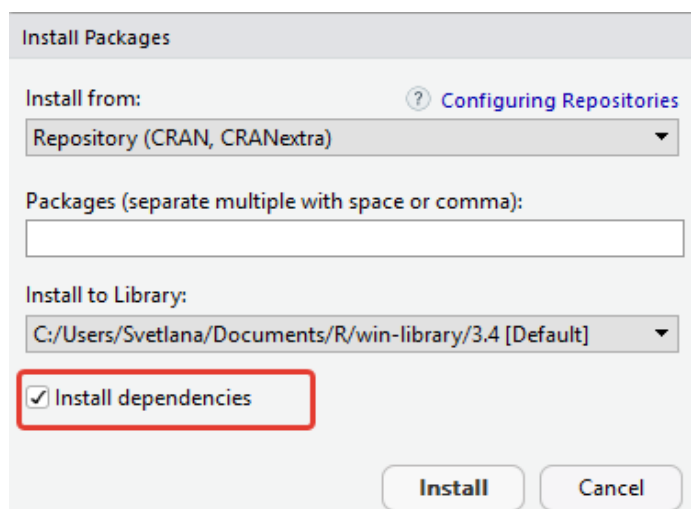


Рисунок 1.14 – Установка всех связанных пакетов

Установка новых пакетов возможна и с помощью команды, аргументы которой – это имя пакета и значение true для переменной dependencies, чтобы установились связанные пакеты:

```
install.packages('ggplot', dependencies = T)
```

Установленные пакеты нужно подключить в текущей рабочей сессии, чтобы можно было использовать определенные в них функции и имеющиеся в пакетах наборы данных. Это можно сделать на вкладке Packages, установив флаг на нужном пакете (рисунок 1.15).

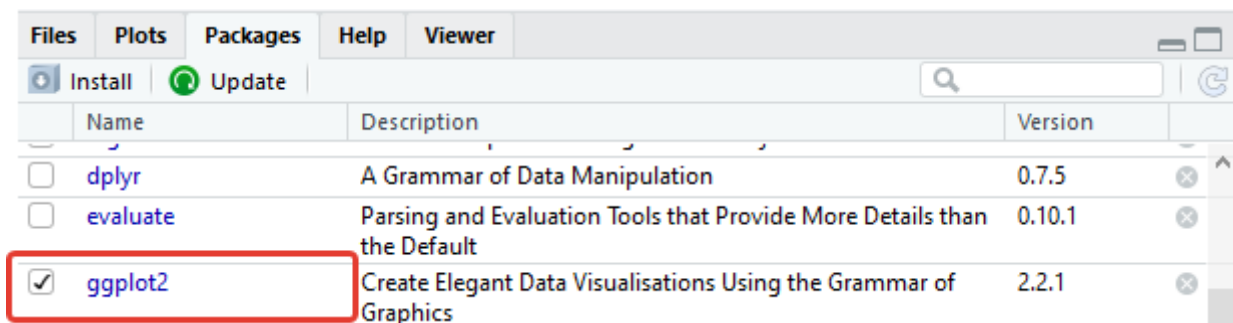


Рисунок 1.15 – Подключение пакета в текущей сессии

Или с помощью команды

```
library(ggplot2)
```

Но подключение пакета с помощью команды не гарантирует, что такой пакет уже был установлен (и подключение не выполнится), тогда как в первом варианте недоступные для подключения пакеты просто не отображаются.

На вкладке Help можно получить справку по пакетам, функциям, наборам данных и т.п. (рисунок 1.16).

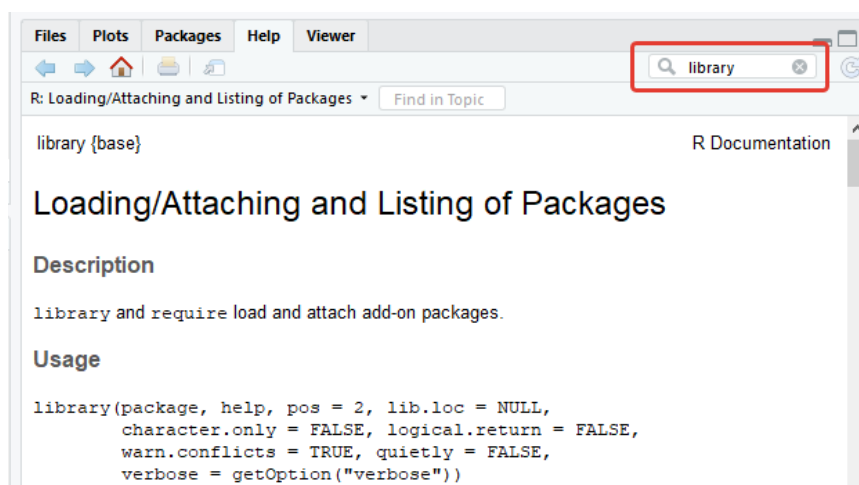


Рисунок 1.16 – Справка по функции library()

Или с помощью знака вопрос перед объектом для получения о нем справки:

?library()

На вкладке **Plots** отображаются построенные в текущей сессии графики (рисунок 1.17).

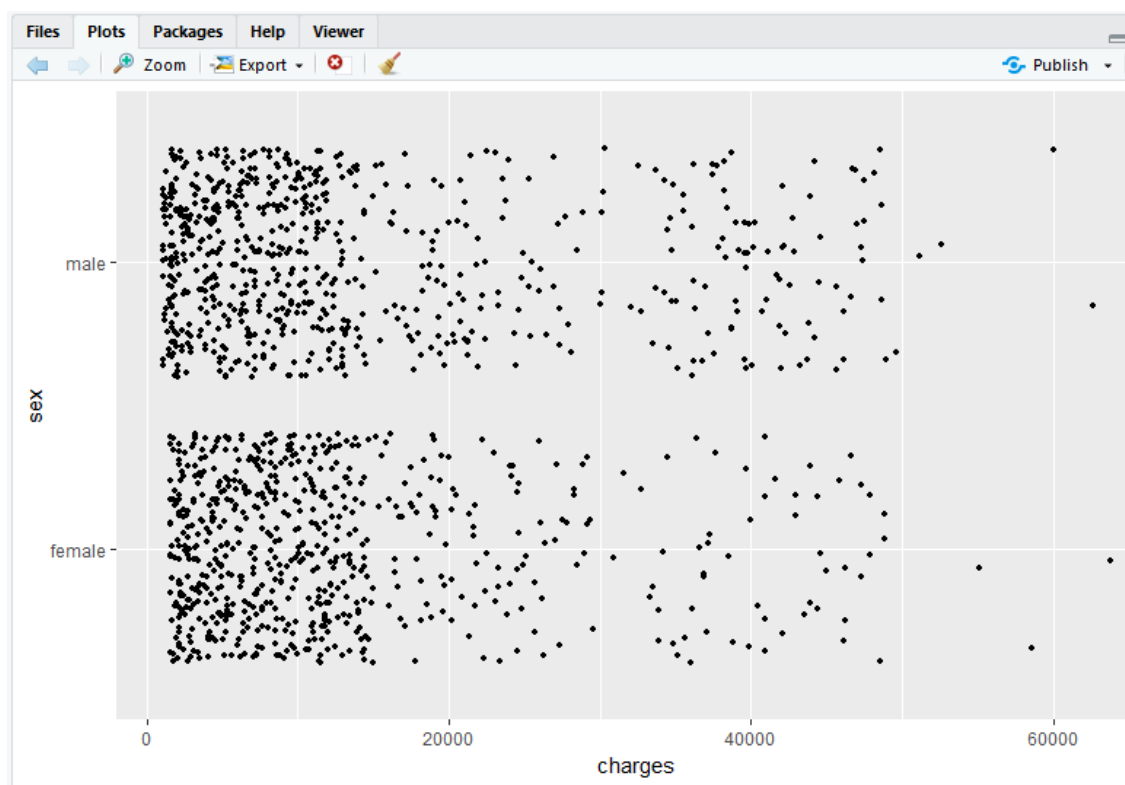


Рисунок 1.17 – График

На вкладке **Plots** (рисунок 1.18): 1 – между графиками можно перемещаться, 2 – увеличивать размер графика, 3 – сохранять график в доступном формате (как рисунок или файл .pdf).

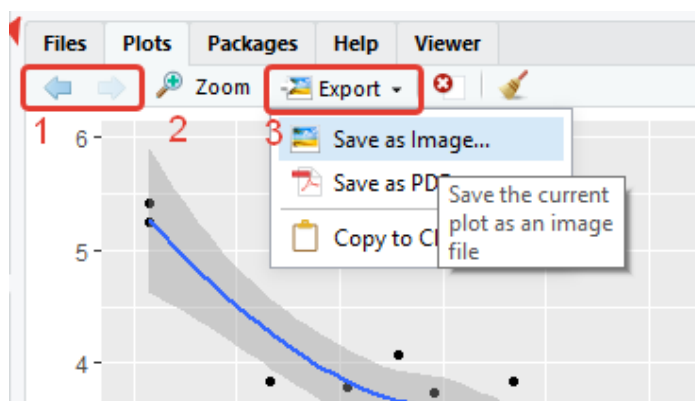


Рисунок 1.18 – Операции с графиками

1.4 Типы данных в R

Основные типы данных, которые поддерживает язык R:

- логические данные (*logical*);
- вещественные числа (*numeric*);
- комплексные числа (*complex*);
- текстовые данные или символьные (*character*).

Объекты данных: векторы, факторы, массивы, матрицы, таблицы данных и списки.

Присваивание переменной значений с помощью `<-`, несколько значений передаются с помощью команды `c()`, в которой отдельные значения разделяются запятой, а текстовые значения заключаются в кавычки. Комментируется код с помощью `#`.

Базовый объект данных в R – вектор. Создание вектора:

```
v1 <- c(T, F, F) #логический вектор – logical
v2 <- c(3, 4, 8, 9) #числовой вектор – numeric
v3 <- c("male", "female", "male", "male", "female") #вектор
символьных значений – character
```

После выполнения кода на вкладке **Environment** отобразятся все три переменные с указанием типа данных элементов вектора (рисунок 1.19).

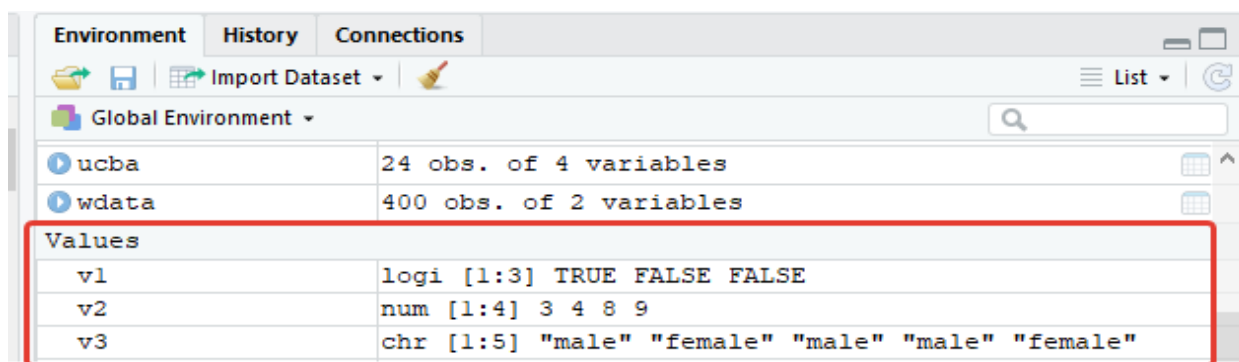


Рисунок 1.19 – Переменные (в служебной вкладке RStudio)

Вектор может содержать данные только одного типа (рисунок 1.20). Самый общий тип данных – символьный.

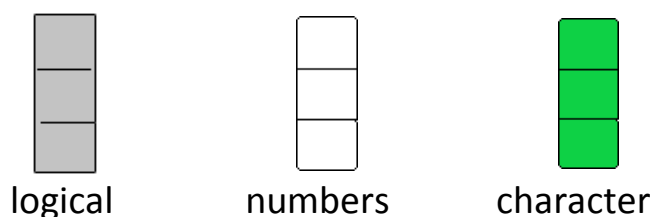


Рисунок 1.20 – Векторы

Если среди логических значений будет хотя бы одно число, то вектор определится как числовой, а если одно из значений символьное, то и весь вектор определится как вектор символьных значений (рисунок 1.21):

```
v1 <- c(T, F, F, 8) #числовой вектор – numeric
v2 <- c(T, F, F, "female") #вектор символьных значений – character
```

Values	
v1	num [1:4] 1 0 0 8
v2	chr [1:4] "TRUE" "FALSE" "FALSE" "female"

Рисунок 1.21 – Переменные (в служебной вкладке RStudio)

Частный вид векторной переменной – фактор. Фактор это группирующая переменная. Так переменная v3 может быть преобразована в фактор:

```
v3.f <- factor(v3)
```

Можно создать факторную переменную v4, значения которой будут упорядочены:

```
v4 <- factor(c("S", "XS", "XXL", "L", "XS"), levels = c("XS", "S", "M", "L", "XL", "XXL", "XXXL"))
```

Отличие этих факторов в том, что в первом случае числовые метки присваиваются при лексикографическом упорядочении (как в словарях), а во втором – согласно следованию значений в переменной levels (рисунок 1.22).

v3.f	Factor w/ 2 levels "female","male": 2 1 2 2 1
v4	Factor w/ 7 levels "XS","S","M","L",...: 2 1 6 4 1

Рисунок 1.22 – Факторы (в служебной вкладке RStudio)

Матрица – это частный случай массива (двумерный массив), как, впрочем, и вектор – это одномерный массив. На матрицу также накладывается ограничение на тип значений: у всех значений в матрице один тип данных (рисунок 1.23).

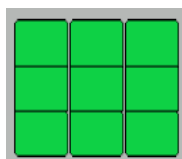


Рисунок 1.23 – Матрица

Создание матриц:

```
n <- matrix(c(1, 7, 3, 4, 8, 1), nrow = 2, ncol = 3)
l <- matrix(c(1, 7, 3, 4, 8, "1"), nrow = 2, ncol = 3)
```

Результат во вкладке **Environment** (рисунок 1.24).

l	chr [1:2, 1:3] "1" "7" "3" "4" "8" "1"
n	num [1:2, 1:3] 1 7 3 4 8 1

Рисунок 1.24 – Матрицы (в служебной вкладке RStudio)

Массив в практике анализа данных используется достаточно редко, у всех значений в массиве – один тип данных (рисунок 1.25).

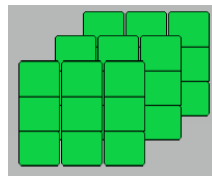


Рисунок 1.25 – Массив

Создание массива, где аргумент `dim` задает размерность (три матрицы размерности 2*2), а имя переменной вызывает ее содержимое в консоль:

```
ar<-array(data=1:12, dim=c(2,2,3))
ar
```

Результат выполнения кода в **Console** (рисунок 1.26).

```
Console ~/
> ar<-array(data=1:12, dim=c(2,2,3))
> ar
, , 1
      [,1] [,2]
[1,]     1     3
[2,]     2     4
, , 2
      [,1] [,2]
[1,]     5     7
[2,]     6     8
, , 3
      [,1] [,2]
[1,]     9    11
[2,]    10    12
```

Рисунок 1.26 – Вывод значений созданного массива `ar` в консоль

Таблица данных или двумерный набор данных встречается в анализе данных особенно часто. Колонки в таблице могут иметь различный тип данных, но в пределах одной колонки (столбца) – все данные одного типа (рисунок 1.27).

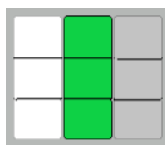


Рисунок 1.27 – Таблица данных

Создание таблицы данных:

```
sale<-data.frame(IDcustomer=c(1101, 1102, 1105), gender=
c("male", "female", "male"), Totalsum=c(18000, 1920, 1078))
```

Результат выполнения кода во вкладке Environment (рисунок 1.28).

Environment	History	Connections
Global Environment		
sale	3 obs. of 3 variables	
IDcustomer: num 1101 1102 1105		
gender : Factor w/ 2 levels "female","male": 2 1 2		
Totalsum : num 18000 1920 1078		

Рисунок 1.28 – Таблица данных sale

Также таблицу данных можно отобразить в окне Source, щелкнув по имени таблицы данных во вкладке Environment (рисунок 1.29).

	IDcustomer	gender	Totalsum
1	1101	male	18000
2	1102	female	1920
3	1105	male	1078

Рисунок 1.29 – Таблица данных sale

1.5 Импорт данных в R

Язык R имеет множество возможностей для импорта данных из самых разных источников. Кроме того, устанавливаемые пакеты также содержат наборы данных.

Рассмотрим встроенные в пакеты наборы данных. В базовой установке есть пакет `datasets`, в котором большое количество именованных наборов данных. Информацию о наборах данных можно получить по команде:

```
library(help = "datasets")
```

В результате выполнения кода в окне **Source** появится список наборов данных (рисунок 1.30).

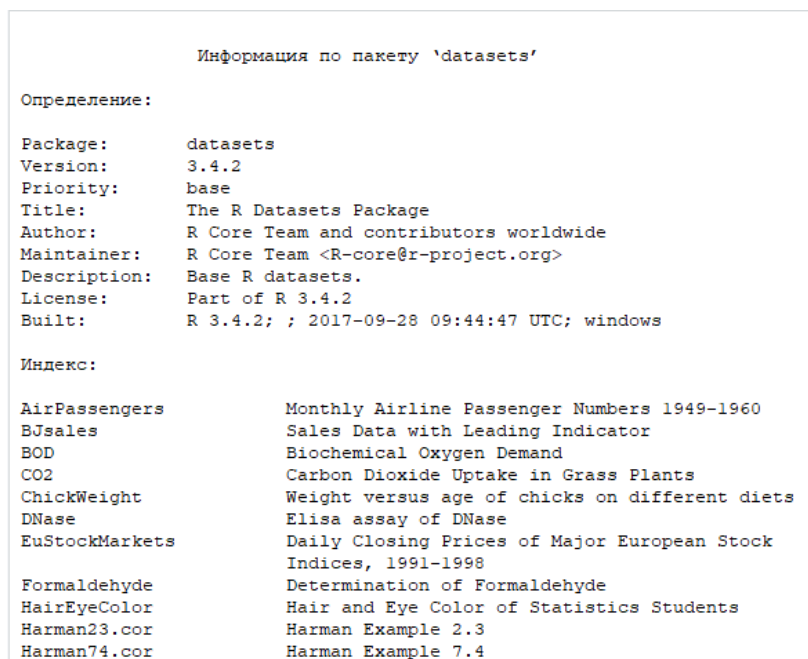


Рисунок 1.30 – Список наборов данных в пакете `datasets`

Более подробно о каждом наборе можно узнать во вкладке **Help** (рисунок 1.31).

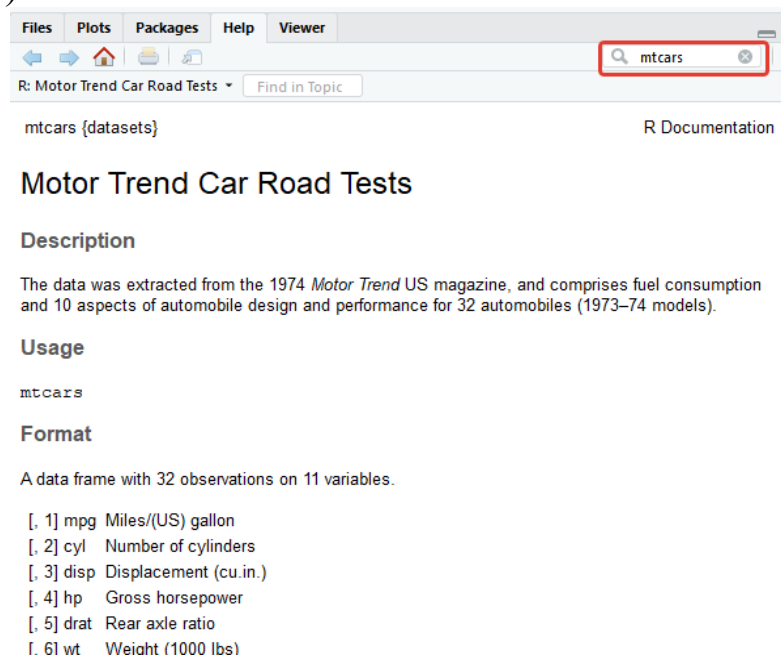
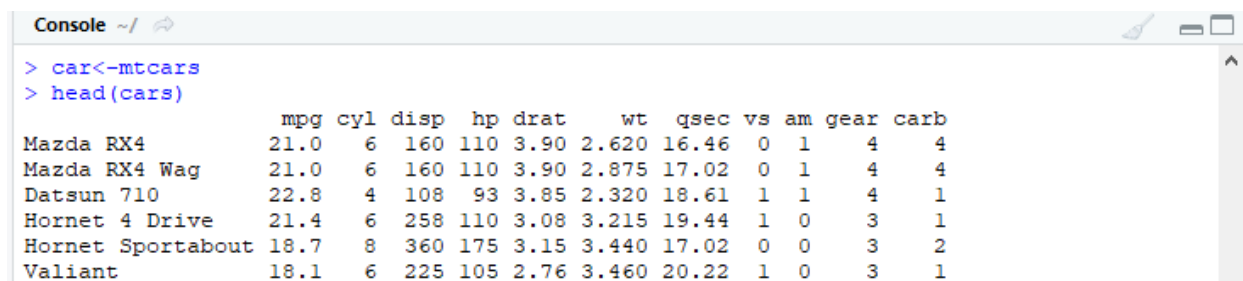


Рисунок 1.31 – Информация о наборе `mtcars`

Передать набор в новую переменную и просмотреть начало таблицы:

```
car<-mtcars  
head(cars)
```

Результат выполнения кода в консоли (рисунок 1.32).



	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

Рисунок 1.32 – Первые шесть строк таблицы данных car

Наборы данных, размещенные на сайтах в сети Интернет, можно загружать несколькими способами. Можно предварительно скачать файл данных, разместить в рабочей папке и использовать в функции `read.table()` в качестве аргумента `file` непосредственно имя файла. Можно в качестве аргумента `file` использовать гиперссылку на файл в Интернете. Не все варианты размещения данных позволяют обращаться к ним по ссылке, поэтому используем первый способ.

На сайте Kaggle (<https://www.kaggle.com/>) необходимо пройти регистрацию, после чего станут доступны наборы данных для скачивания (рисунок 1.33).

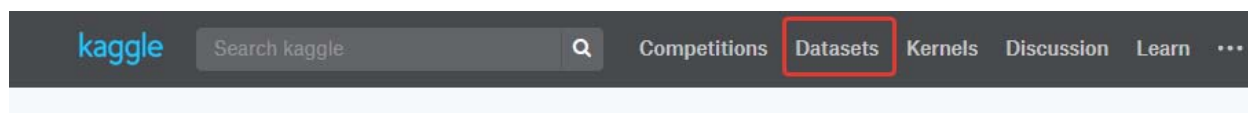


Рисунок 1.33 – Платформа kaggle

Доступны публичные наборы данных (рисунок 1.34), которые отфильтрованы по объему (<10 Mb), так как без специальных ухищрений R не позволяет напрямую из консоли анализировать большие объемы данных.

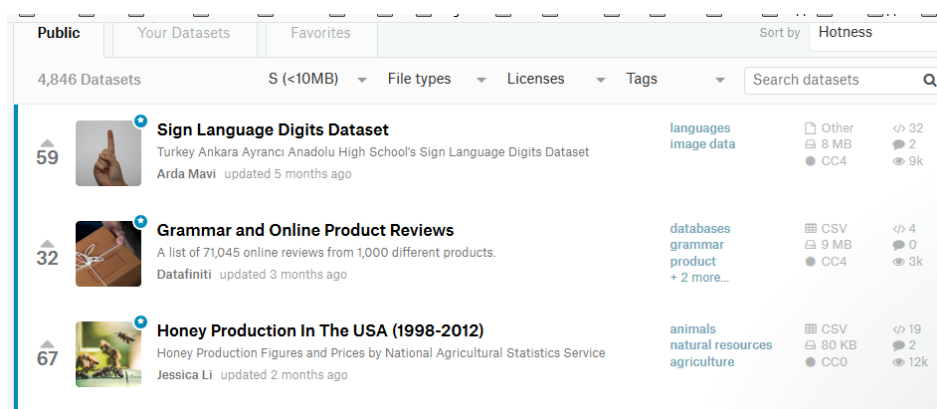


Рисунок 1.34 – Наборы данных

Среди наборов выберем набор <https://www.kaggle.com/ulabox/ulabox-orders-with-categories-partials-2017/data> (рисунок 1.35). И скачаем его.

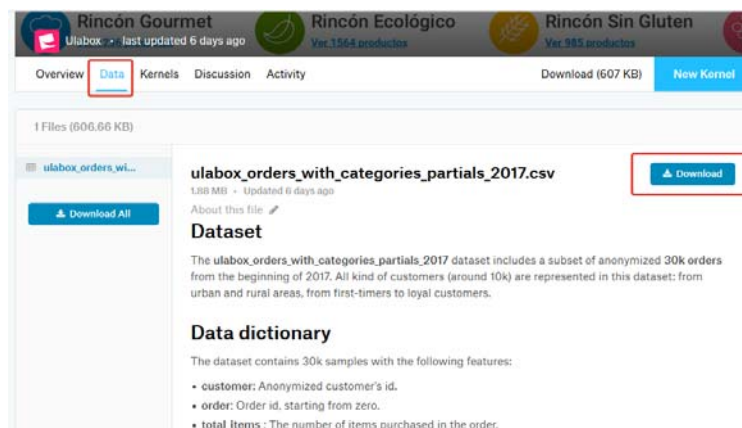


Рисунок 1.35 – Набор данных по заказам в супермаркете Ulabox (Испания) в 2017 г.

Далее перенесем его в рабочую папку и сократим название, чтобы было легче работать с импортом (рисунок 1.36).

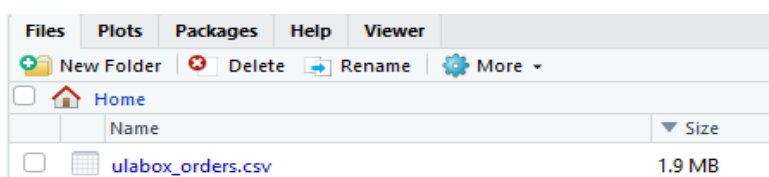


Рисунок 1.36 – Рабочая директория

Импортируем данные в переменную `Orders` с помощью функции `read.table()`, аргументы которой имя файла в кавычках (файл в рабочей директории), `header=T` (так как в первой строке заголовки) и `sep = ","` – разделитель данных (в нашем случае – запятая):

```
Orders<-read.table("ulabox_orders.csv", header=T, sep=",")
```

Результат выполнения кода в двух окнах `Source` и `Environment` на рисунке 1.37.

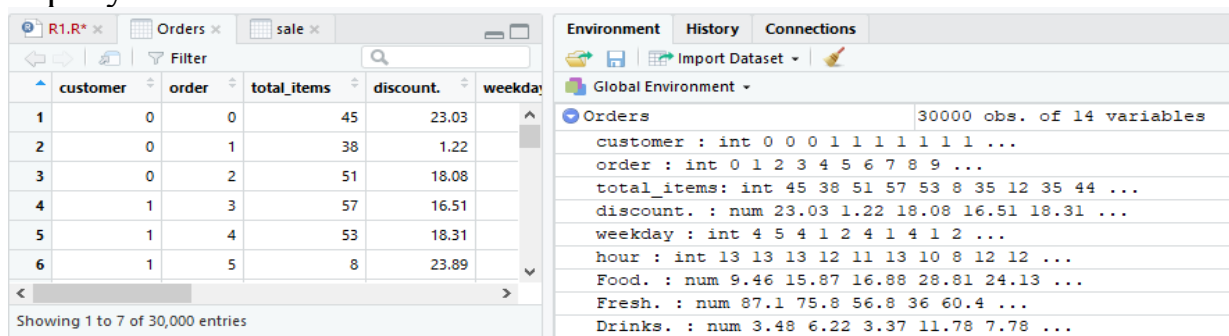


Рисунок 1.40 – Набор данных `Orders`

Краткое описание импортированного набора данных

Набор данных онлайн-супермаркета Ulabox 2017, доступ из <https://www.github.com/ulabox/datasets>. Ulabox – это супермаркет в Испании.

Переменные набора:

- customer: анонимизированный идентификатор клиента;
- order: ID заказа, начиная с нуля;
- total_items: количество приобретенных товаров в заказе;
- discount%: процент от общей полученной скидки;
- weekday: день недели: 1=понедельник, 7=воскресенье;
- hour: час дня, когда была совершена покупка от 00 до 23.

Categories' partials процент денег, потраченных в каждой из восьми основных категорий веб-сайта:

- food%: не скоропортящиеся продукты, например, рис, снеки, соусы и т.п.;
- fresh% : свежие и замороженные продукты, например: фрукты, овощи, замороженное мясо и т.п.;
- drinks% : все виды напитков, например: вода, соки, вино, алкогольные напитки, молоко, соевые напитки;
- home% : товары для дома, от туалетной бумаги до мелкой бытовой техники;
- beauty% : парфюмерия, косметика, уходовые средства;
- health% : лекарственные средства, которые можно продавать в Испании без рецепта врача: диетические таблетки, презервативы, зубная паста;
- baby% : товары по уходу за ребенком, детское питание и т.п.;
- pets% : товары для домашних животных.

2 БАЗОВЫЕ ВОЗМОЖНОСТИ ЯЗЫКА R ДЛЯ ВИЗУАЛИЗАЦИИ ДАННЫХ

2.1 Функция plot()

plot() – основная графическая команда, она распознает тип объекта, который подлежит рисованию, и строит соответствующий график.

Основные аргументы функции plot()

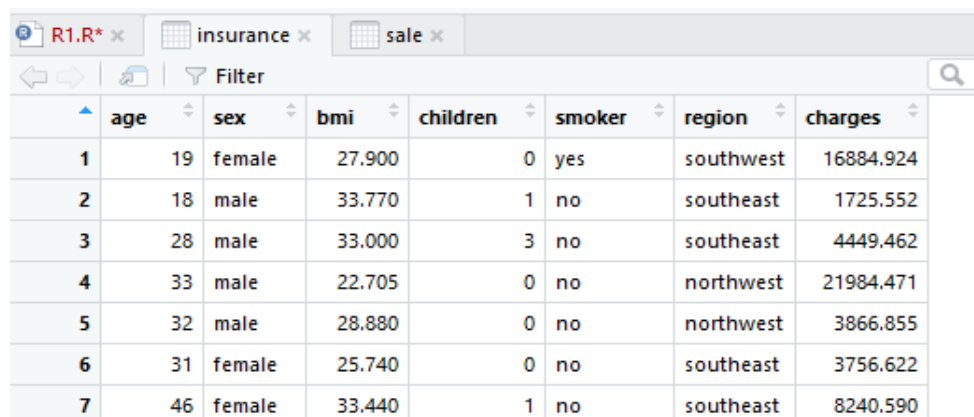
- main – название диаграммы;
- pch (plotting character [0:25]) – вид маркера;
- cex (character extension) – размер;
- col (colour) – цвет;
- xlab – название горизонтальной оси;
- ylab – название вертикальной оси.

Для построения графиков скачаем и импортируем набор [https:// www.kaggle.com/mirichoi0218/insurance](https://www.kaggle.com/mirichoi0218/insurance) – расходы на медицинское обслуживание.

Описание переменных набора:

- age: возраст основного бенефициара;
- sex: пол застрахованного;
- bmi: индекс массы тела;
- children: число детей, охваченных медицинским страхованием / число иждивенцев;
- smoker: курит ли застрахованный;
- region: жилой район получателя в США, Северо-Восток, Юго-Восток, Юго-Запад, Северо-Запад;
- charges: индивидуальные медицинские расходы, оплачиваемые страховкой.

Рассмотрим набор данных – расходы на медицинское обслуживание тех, кто имеет медицинскую страховку. Первые строки набора на рисунке 2.1.



	age	sex	bmi	children	smoker	region	charges
1	19	female	27.900	0	yes	southwest	16884.924
2	18	male	33.770	1	no	southeast	1725.552
3	28	male	33.000	3	no	southeast	4449.462
4	33	male	22.705	0	no	northwest	21984.471
5	32	male	28.880	0	no	northwest	3866.855
6	31	female	25.740	0	no	southeast	3756.622
7	46	female	33.440	1	no	southeast	8240.590

Рисунок 2.1 – Набор данных insurance

Оценим визуально, каковы пропорции женщин и мужчин в наборе. Обращение к переменной происходит по имени таблицы до знака \$ и названию столбца после: `insurance$sex`.

```
plot(insurance$sex, main = "Половой состав застрахованных",  
col = c("red", "blue"))
```

Результат выполнения команды на рисунке 2.2, аргумент `main` задает заголовок.



Рисунок 2.2 – Разделение набора данных по полу

Аналогично проведем оценку пропорций курящих и некурящих в наборе данных:

```
plot(insurance$smoker, main = "Количество курящих  
среди застрахованных", col = c("green", "red"))
```

Результат выполнения команды на рисунке 2.3.



Рисунок 2.3 – Разделение набора данных по наличию вредной привычки

Проанализируем зависимость индивидуальных медицинских расходов в зависимости от пола, возраста и наличия вредных привычек.

Числовые переменные: возраст и расходы – построим на графике. Возраст – независимая переменная, она откладывается по оси x и записывается первой:

```
plot(insurance$age, insurance$charges, main = "Медицинские  
расходы", xlab = "Возраст застрахованного", ylab =  
"Расходы")
```

Результат выполнения команды на рисунке 2.4, аргумент main задает заголовок, а xlab, ylab – подписи осей.

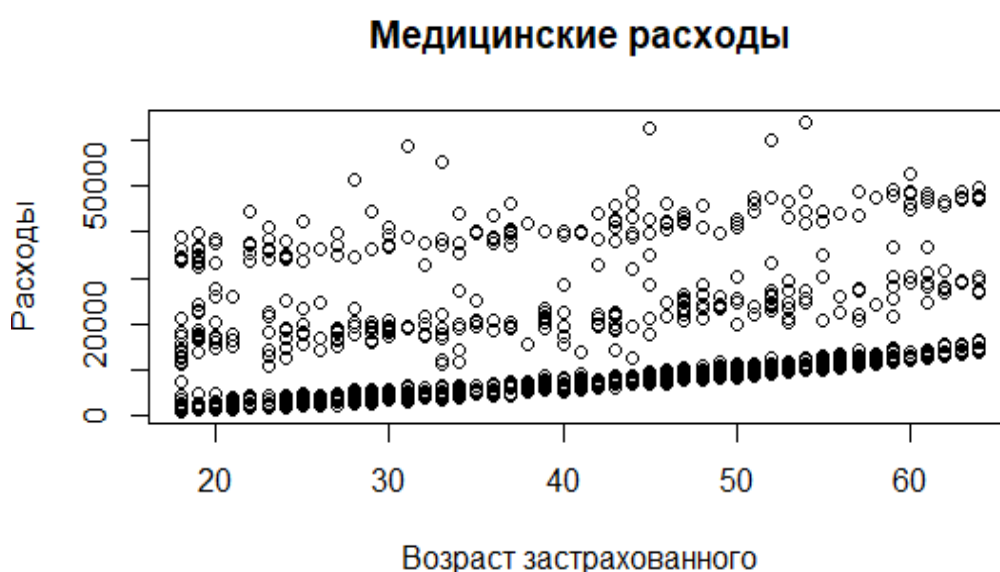


Рисунок 2.4 – Индивидуальные медицинские расходы в зависимости от возраста

Очевидно, что с возрастом индивидуальные медицинские расходы растут. Но интересно узнать, влияет ли пол или скажем наличие вредной привычки на медицинские расходы.

Изменим цвет точек графика в зависимости от пола с помощью функции `ifelse(insurance$sex=="female", "red", "blue")`, если значение пола – женщина, то цвет точек будет красным, иначе – синим:

```
plot(insurance$age, insurance$charges, col = ifelse(insurance$sex == "fe-  
male", "red", "blue"), main = "Медицинские расходы", xlab = "Возраст  
застрахованного", ylab = "Расходы")
```

И добавим легенду: первый аргумент задает ее расположение “top(bottom)right(left)”, второй размер (лучше брать меньше 1, чтобы

не перекрывался график), третий вид маркера для точек, четвертый цвет точек и последний текст подписей точек в легенде:

```
legend("topleft", cex = .5, pch = 1, col = c("red", "blue"),  
c('женщины', 'мужчины'))
```

Результат выполнения команд на рисунке 2.5.

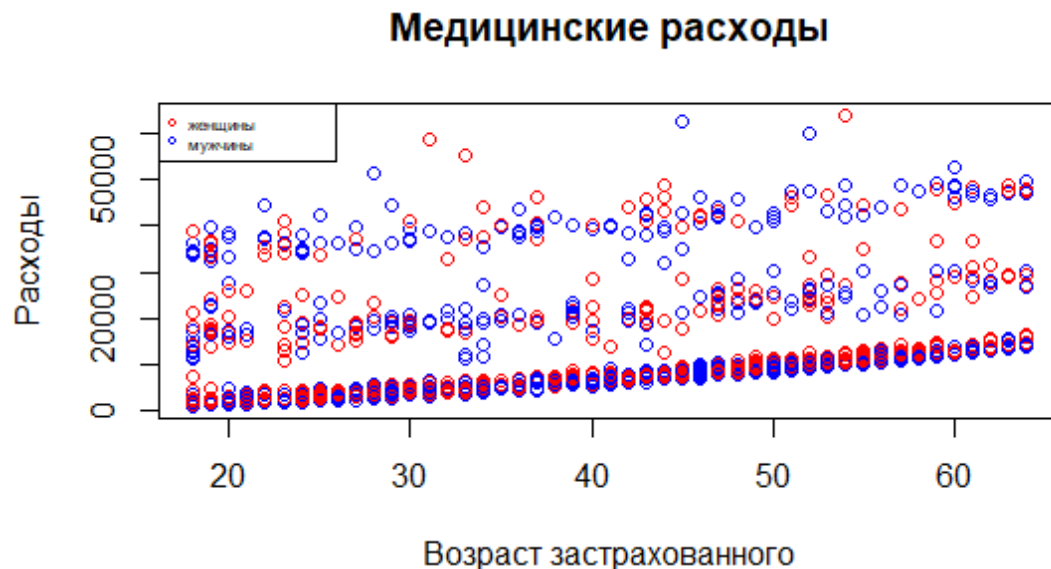


Рисунок 2.5 – Индивидуальные расходы в зависимости от возраста с учетом пола

Визуально разница между индивидуальными расходами мужчин и женщин не очень бросается в глаза. Попробуем добавить данные о курении, изменив вид маркера для курящих: `pch = as.numeric(insurance$smoker)`. Необходимо добавить преобразование к числовому виду переменной `insurance$smoker` с помощью функции `as.numeric()`:

```
plot(insurance$age, insurance$charges, col =  
ifelse(insurance$sex == "female", "red", "blue"),  
pch=as.numeric(insurance$smoker), main = "Медицинские  
расходы", xlab = "Возраст застрахованного", ylab = "Расходы")
```

И в легенду теперь добавляется вид маркера и дополнительное значение подписей точек легенды:

```
legend("topleft", cex= .5, c('некурящие женщины', 'некурящие  
мужчины', 'курящие женщины', 'курящие мужчины'),  
pch=c(1, 1, 2, 2), col = c("red", "blue"))
```

Теперь наш график имеет совсем подробный вид (рисунок 2.6).

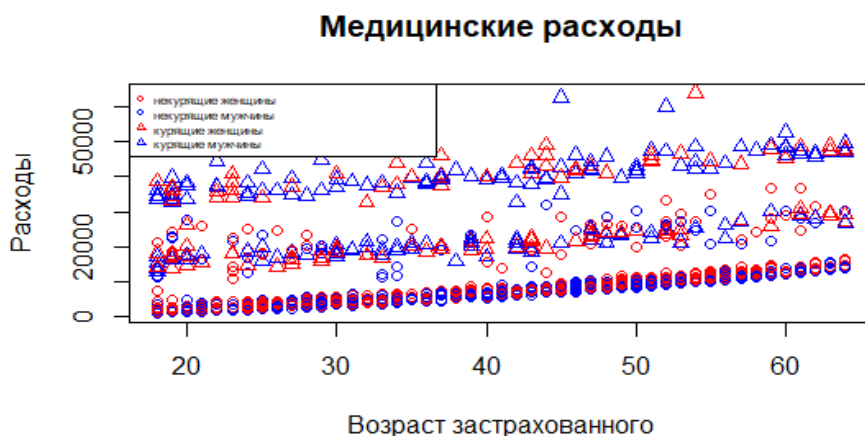


Рисунок 2.6 – Индивидуальные расходы в зависимости от возраста с учетом пола и вредной привычки

И хорошо заметно, что большее число треугольников сосредоточено в зоне высоких медицинских расходов, а большее число кружочков в зоне небольших индивидуальных медицинских расходов.

Построим аналогичным образом график зависимости медицинских расходов от индекса массы тела с учетом пола и наличия вредной привычки:

```
plot(insurance$bmi, insurance$charges, col =
ifelse(insurance$sex == "female", "red", "blue"), pch=as.numeric
( insurance$smoker), main = "Медицинские расходы", xlab =
"Индекс массы тела", ylab = "Расходы")
legend("topleft",cex=.5, c('некурящие женщины',
'некурящие мужчины', 'курящие женщины', 'курящие мужчины'),
pch = c(1, 1, 2, 2), col = c("red", "blue"))
```

Теперь четко прослеживается, что повышенный индекс массы тела при наличии вредной привычки в совокупности существенно повышают медицинские расходы, независимо от пола (рисунок 2.7).

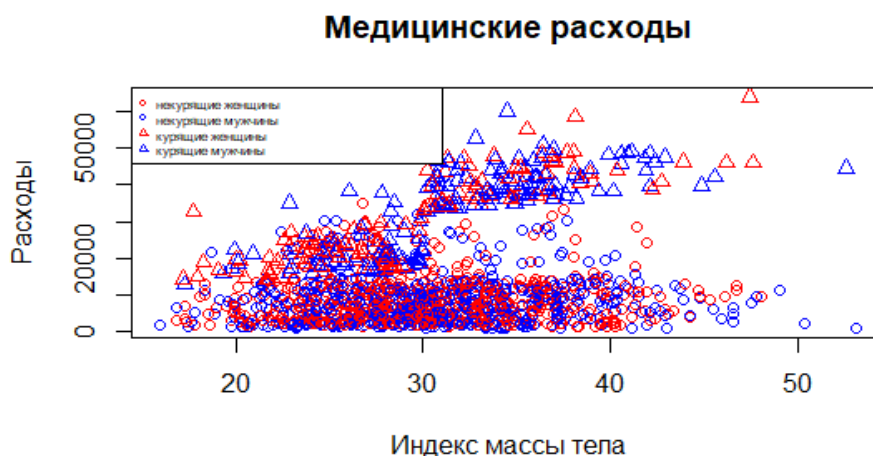


Рисунок 2.7 – Индивидуальные расходы в зависимости от индекса массы тела с учетом пола и вредной привычки

2.2 Функция `boxplot()`

`boxplot()` – диаграмма размахов или «ящички с усами» (англ. *box-whisker plots*) имеет характерный вид ящика включающего в себя 50 % срединных значений (между квантилями 0,25 и 0,75). На диаграмме размахов также визуально определяется наличие в данных выбросов (рисунок 2.8).

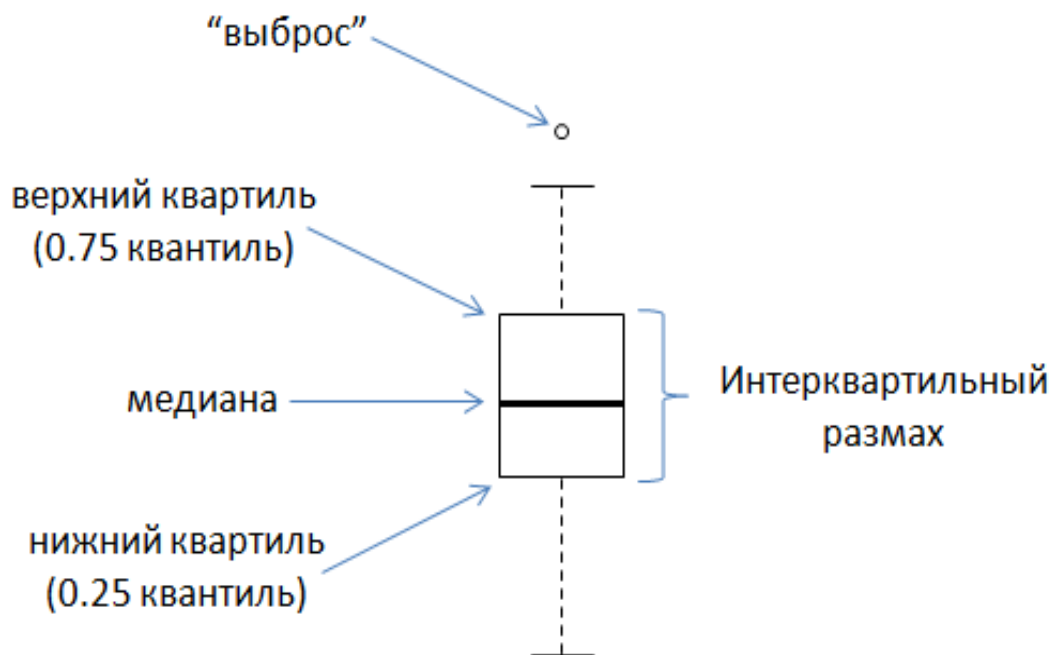


Рисунок 2.8 – Диаграмма размаха

Основные аргументы функции `boxplot()`:

- `formula` – формула, такая как `y ~ group`, где `y` – числовой вектор значений данных, которые должны быть разбиты на группы в соответствии с группирующей переменной `group` (обычно фактор);

- `data` – набор данных, переменные из которого входят в формулу.

Построим диаграмму размаха для индивидуальных медицинских расходов:

```
boxplot(insurance$charges, main = "Медицинские расходы")
```

Как видим (рисунок 2.9), в данных довольно много выбросов: это тяжелые заболевания у отдельных застрахованных, которые сопряжены с очень высокими расходами, нехарактерными для основной части наблюдений.



Рисунок 2.9 – Диаграмма размаха для медицинских расходов

Построим диаграммы размахов по медицинским расходам для двух частей выборки, разделенной по наличию вредной привычки. В качестве первого аргумента для `boxplot()` теперь используется функция: слева расходы – зависимая величина, справа наличие вредной привычки – независимая величина, второй аргумент – имя набора данных:

```
boxplot(charges~smoker, insurance, main = "Медицинские  
расходы в зависимости от курения")
```

Разница в медицинских расходах между курящими и некурящими очень существенная (рисунок 2.10).

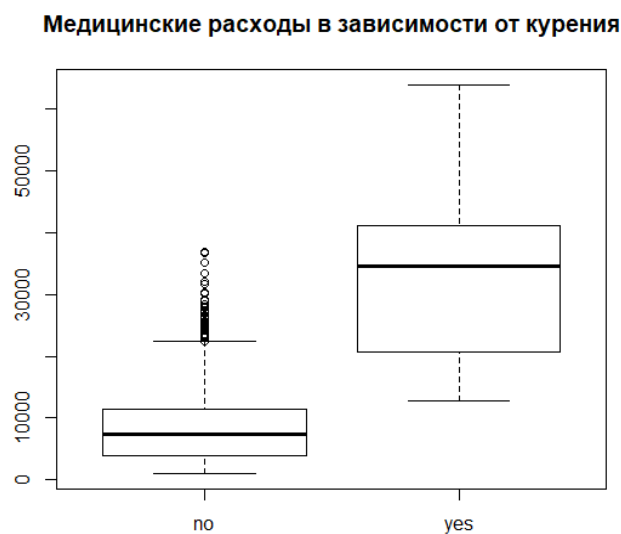


Рисунок 2.10 – Диаграмма размаха медицинских расходов для курящих и некурящих

Построим диаграммы размаха для медицинских расходов в зависимости от количества детей (рисунок 2.11):

```
boxplot(charges~children, insurance, main = "Медицинские  
расходы в зависимости от количества детей")
```

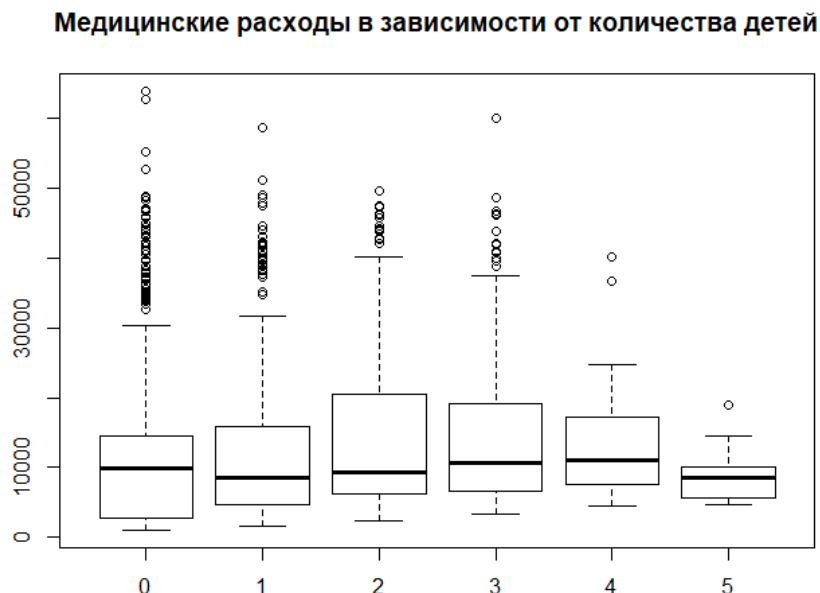


Рисунок 2.11 – Диаграмма размаха медицинских расходов при различном количестве детей

Построим диаграммы размахов по медицинским расходам для сочетания двух факторов: наличия вредной привычки и пола. Для этого введем в исходный набор данных две новые переменные: `sex1` и `smoker1`, в которых запишем вместо исходных значений переменных их метки: 1 – женский, 2 – мужской и, соответственно, 1 – некурящий, 2 – курящий:

```
insurance$smoker1 <- as.numeric(factor(insurance$smoker))
insurance$sex1 <- as.numeric(factor(insurance$sex))
```

Далее представим значения этих переменных как факторы двумя уровнями, которым соответствуют определенные метки:

```
insurance$smoker1 <- factor(insurance$smoker1,
levels=c(1, 2), labels=c("некурящий", "курящий"))
insurance$sex1 <- factor(insurance$sex1,
levels=c(1, 2), labels=c("женский", "мужской"))
```

Теперь можно построить график, в котором диаграммы размаха для медицинских расходов построены для различных сочетаний значений: пол и наличие вредной привычки (рисунок 2.12):

```
boxplot (charges ~ sex1*smoker1, insurance, varwidth = TRUE,
col = c("red","blue"), main = "Индивидуальные медицинские расходы")
```

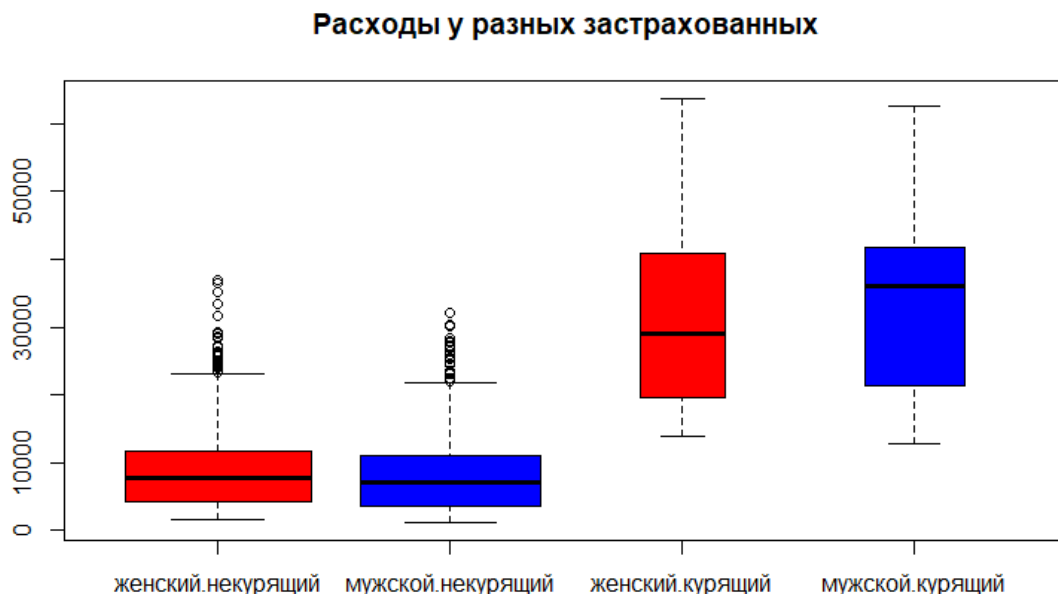


Рисунок 2.12 – Диаграмма размаха медицинских расходов при различном сочетании значений факторов: пол и наличие вредной привычки

2.3 Функция mosaicplot()

mosaicplot() – это графический метод визуализации данных из двух или более качественных переменных.

Основные аргументы функции **mosaicplot()**

– **x** – таблица сопряженности в виде массива с дополнительными категориями, указанными в **dimnames(x)**. Таблицу лучше всего создавать командой **table ()**.

Построим таблицу сопряженности для факторов: пол и наличие вредной привычки:

```
x <- table(insurance$sex, insurance$smoker)
```

Таблица сопряженности разбивает все множество наблюдений на четыре группы: курящие женщины, курящие мужчины, некурящие женщины и некурящие мужчины. Если между этими показателями есть связь, то ее наличие можно обнаружить с помощью графического представления таблицы сопряженности благодаря применению к таблице функции **mosaicplot()**:

```
mosaicplot(x, color = T, shade = T, main = "Сопряженность признаков пол и курение", xlab = "пол", ylab = "курение")
```

На рисунке 2.12 видно, что связи между этими показателями нет, хотя среди женщин, застрахованных в компании, меньший процент курящих, чем среди мужчин.



Рисунок 2.12 – Сочетание в наборе данных признаков: пол и наличие вредной привычки

3 ВОЗМОЖНОСТИ ПАКЕТА GGPLOT2

Первоначально подключим пакет `ggplot2`

```
library(ggplot2)
```

Базовой графической командой пакета `ggplot2` является команда `qplot()`.

Основные аргументы функции `qplot()`:

- `x`, `y` – переменные набора данных;
- `data` – набор данных, переменные из которого входят в формулу;
- `color`, `shape`, `size`, `fill` – определяет соответствие цвета, формы и размера символов уровням переменной, для диаграмм плотности и размахов параметр `fill` определяет соответствие заполняющего цвета значениям переменной;
- объекты, которые задают тип диаграммы. Этот параметр задается в формате текстового вектора с одним или более элементами: "point", "smooth", "boxplot", "line", "histogram", "density", "bar" и "jitter";
- `facets` – используется для построения категоризованных диаграмм: если используются две факторные переменные для разбиения набора данных, то между ними ставится `~`: `facets = sex1~smoker1`, если одна переменная, то либо `facets=children~.`, либо `facets= . ~ children`;
- `method` – можно использовать следующие методы сглаживания (параметр `formula`): "lm" (регрессия), "gam" (обобщенные аддитивные модели) и "rim" (устойчивая регрессия).

Построим точечный график зависимости индивидуальных медицинских расходов от возраста (рисунок 3.1):

```
qplot(x = age, y = charges, data = insurance, geom = "point", xlab =  
"Возраст", ylab = "Расходы")
```

Построим диаграммы размахов медицинских расходов для курящих и некурящих клиентов страховой компании (рисунок 3.2):

```
qplot(smoker1, charges, data = insurance, geom = c("boxplot",  
"jitter"), fill=smoker1, xlab="Наличие вредной привычки",  
ylab="Расходы", main="Диаграммы размахов")
```

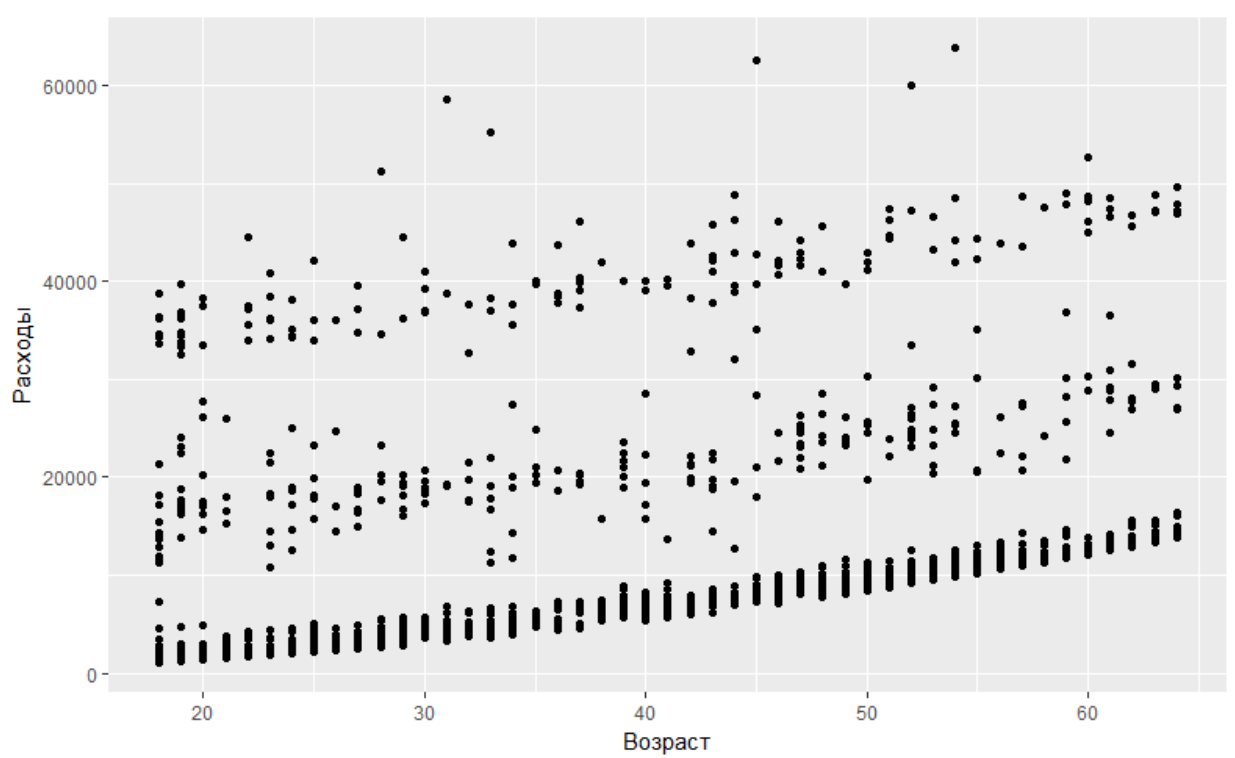


Рисунок 3.1 – Точечный график расходов от возраста застрахованного

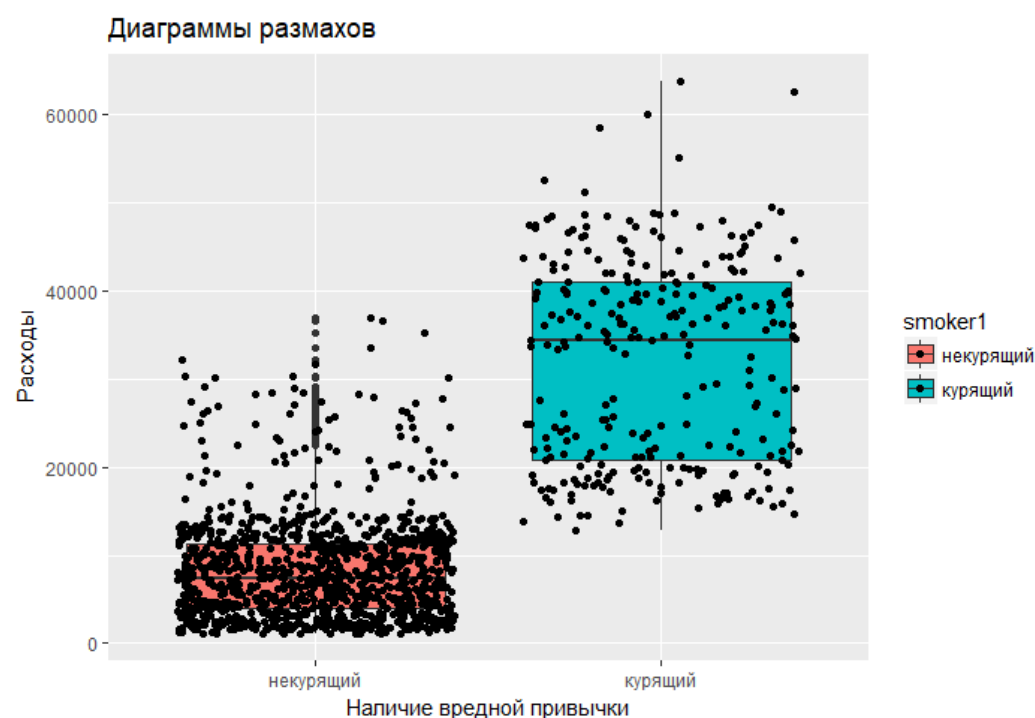


Рисунок 3.2 – Диаграммы размахов расходов для курящих и некурящих клиентов

Построим точечные графики для набора данных разделенного на выборки в зависимости от сочетания значений факторов: пол и наличие вредной привычки (рисунок 3.3):

```
qplot(age, charges, data = insurance, facets = sex1~smoker1,  
xlab = "Возраст", ylab = "Расходы" )
```

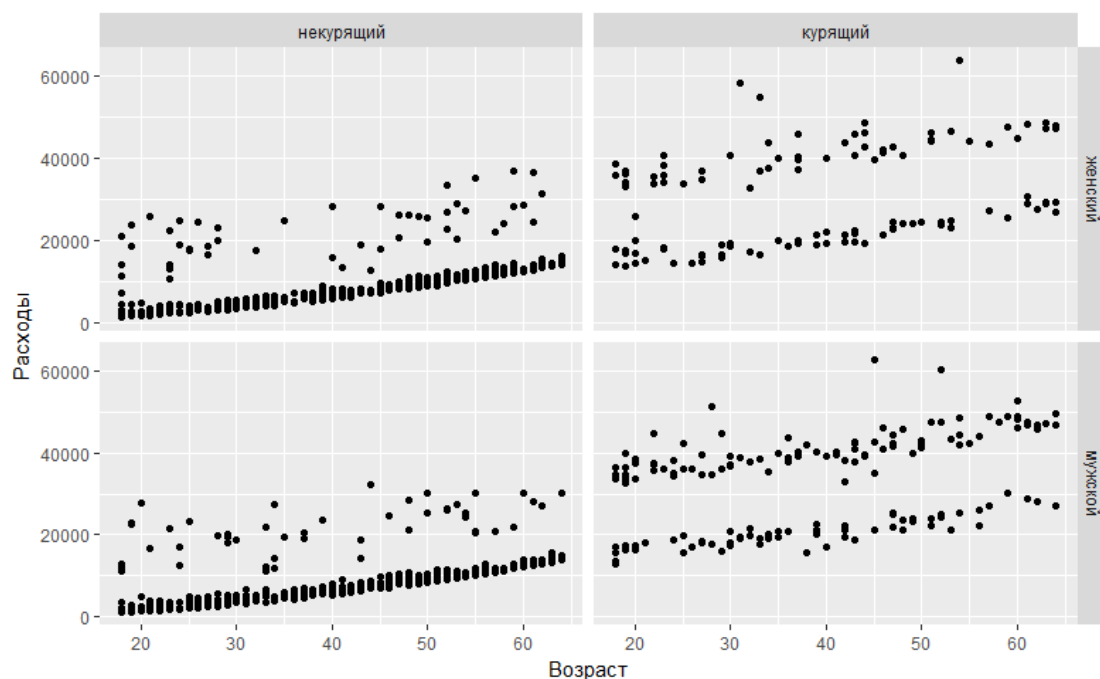


Рисунок 3.3 – Точечный график расходов от возраста застрахованного
(в зависимости от сочетания значений факторов:
пол и наличие вредной привычки)

Построим график плотности (`geom="density"`) распределения расходов для выборок разделенных по полу (рисунок 3.4):

```
qplot(charges, data = insurance, geom="density", facets=sex~., fill=sex,  
xlab = "Расходы")
```



Рисунок 3.4 – График плотности распределения расходов
(для мужчин и женщин)

Построим график плотности распределения расходов для выборок, разделенных по количеству детей (рисунок 3.5):

```
qplot(charges, data = insurance, geom="density",  
facets=children~., fill=children, xlab = "Расходы")
```

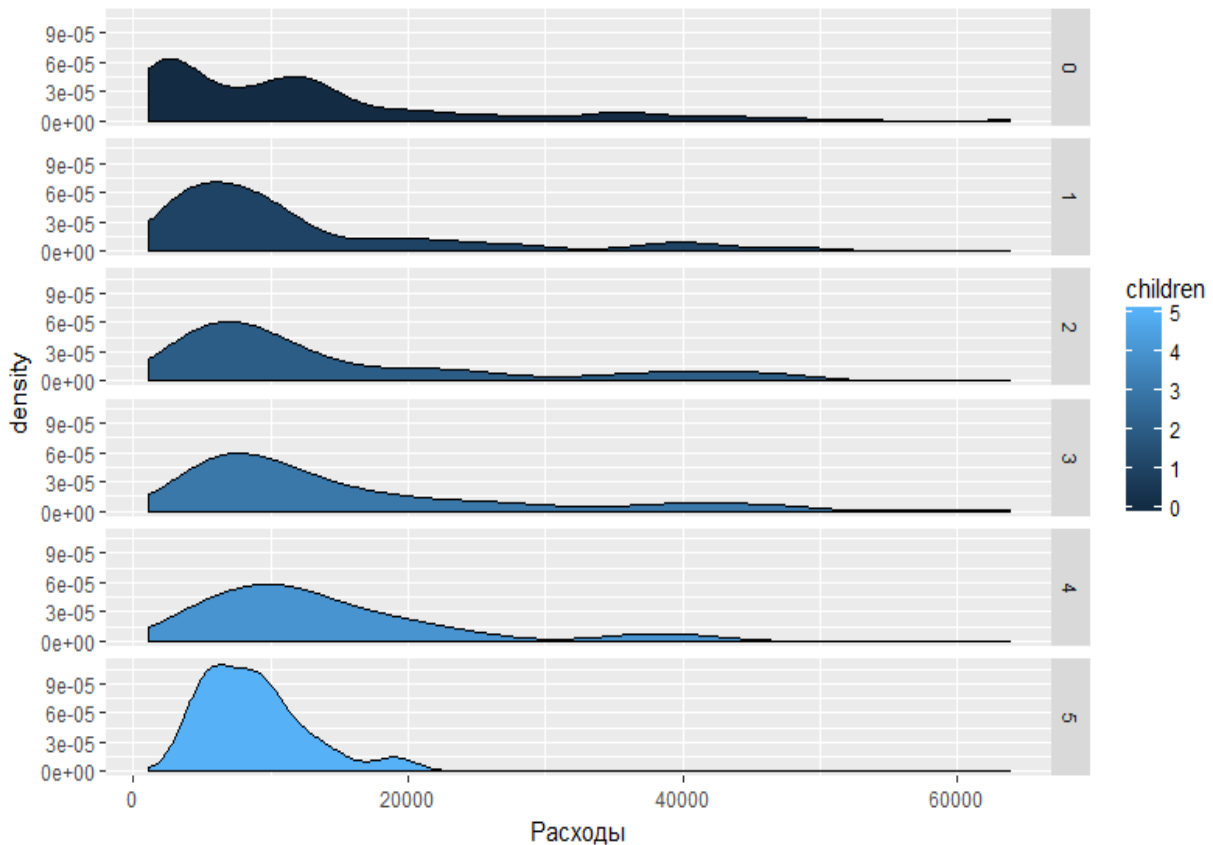


Рисунок 3.5 – График плотности распределения расходов
(в зависимости от числа детей)

Построим две регрессии на одной координатной плоскости (регрессия – это функциональная зависимость между случайными величинами, одна из которых рассматривается как независимая, другая зависимая). Первая регрессия для некурящих, вторая для курящих клиентов страховой компании (рисунок 3.6). Очевидно, что медицинские расходы у этих групп клиентов существенно различаются (в пользу некурящих):

```
qplot(age, charges, data = insurance, color = smoker1,  
shape = smoker1, geom = c("point", "smooth"), method = "lm",  
formula = y~x, xlab = "Возраст", ylab = "Расходы",  
main = "Пример с регрессией")
```

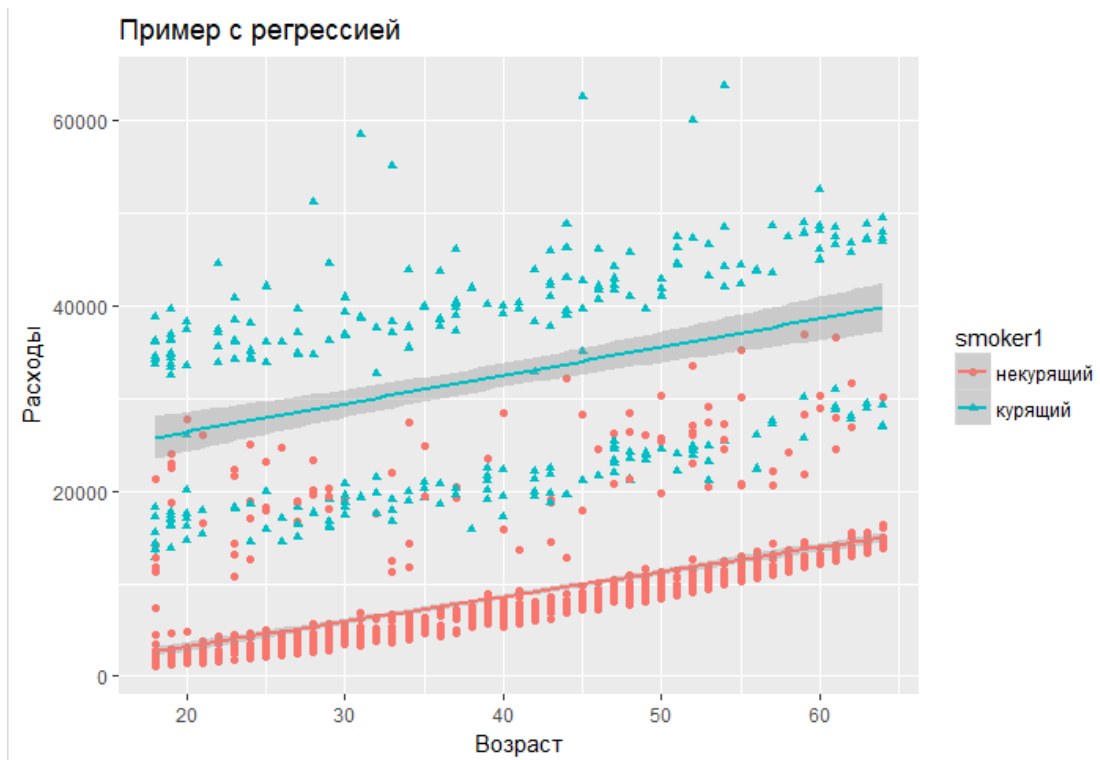


Рисунок 3.6 – Линейные регрессии расходов в зависимости от возраста отдельно для курящих и некурящих клиентов

4 ЛАБОРАТОРНЫЕ РАБОТЫ

Лабораторная работа № 1 «Визуализация данных: базовая графика»

Цель работы: научиться импортировать данные и применять к ним базовые графические функции языка R для визуализации и выдвижения гипотез о наличии зависимостей.

Формируемые знания, умения и навыки: изучить базовые графические возможности языка R, уметь импортировать данные из доступных источников.

Необходимо:

1. С сайта Kaggle (<https://www.kaggle.com/>) импортировать один из наборов, включающий столбцы с числовыми значениями размером <5 Mb.
2. Используя базовые графические возможности языка R, провести визуальный анализ данных и выдвинуть гипотезы о наличии зависимостей между переменными набора.

Контрольные вопросы и задания

1. Какая функция используется для импорта таблиц данных в формате .csv? Какие аргументы у этой функции? Какие значения могут принимать аргументы этой функции?
2. Какие аргументы у функции plot()? Какие значения могут принимать эти аргументы?
3. Какие аргументы у функции legend()? Какие значения они могут принимать?
4. Опишите выбранный набор данных: контекст данных, переменные из набора, зависимости между переменными, выявленные визуально.

Лабораторная работа № 2 «Визуализация данных: пакет ggplot2»

Цель работы: научиться использовать расширенные графические возможности, предоставляемые пакетом ggplot2.

Формируемые знания, умения и навыки: изучить возможности визуализации данных для разведочного анализа.

Необходимо:

1. На том же наборе данных провести построение графиков с помощью функции qplot() пакета ggplot2.

2. Самостоятельно изучить возможности визуализации данных с помощью функции `ggplot()` пакета `ggplot2`.

Контрольные вопросы и задания

1. Какие аргументы могут быть у функции `qplot()`?
2. Какие значения может принимать аргумент `geom`?
3. Как определяется `facet`?
4. Какие значения может принимать `method`?
5. Как определяется функция `ggplot()`?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Введение в статистическое обучение с примерами на языке R / Г. Джеймс, Д. Уиттон, Т. Хасты, Р. Тибширани. – М. : ДМК Пресс, 2017. – 456 с. – URL: [https:// e.lanbook.com/book/93580](https://e.lanbook.com/book/93580)
2. Кабаков, Р. И. R в действии. Анализ и визуализация данных в программе R / Р. И. Кабаков. – М. : ДМК Пресс, 2014. – 588 с.
3. Что такое открытые данные. Восемь принципов открытости. – URL: <http://odc.open.gov.ru/mod/page/view.php?id=100>
4. Наглядная статистика. Используем R! / А. Б. Шипунов, Е. М. Балдин, П. А. Волкова, А. И. Коробейников, С. А. Назарова, С. В. Петров, В. Г. Суфиянов. – URL: <http://ashipunov.info/shipunov/school/sch-ru.htm>

Учебное издание

Рындина Светлана Валентиновна

**Бизнес-аналитика:
визуализация данных**

Редактор *Е. Г. Акимова*
Технический редактор *Н. В. Иванова*
Компьютерная верстка *Н. В. Ивановой*

Подписано в печать 03.08.2018.
Формат 60×84¹/₁₆. Усл. печ. л. 2,56.
Тираж 15. Заказ № 541.

Издательство ПГУ
440026, Пенза, Красная, 40.
Тел./факс: (8412) 56-47-33; e-mail: iic@pnzgu.ru

