КАЗАНСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

Институт фундаментальной медицины и биологии

Кафедра биохимии и биотехнологии

Акберова Н.И., Козлова О.С.

ОСНОВЫ АНАЛИЗА ДАННЫХ И ПРОГРАММИРОВАНИЯ В R

Учебно-методическое пособие

Казань – 2017

УДК 51-76; 573
ББК 28

Печатается по решению Учебно-методической комиссии
Института фундаментальной медицины и биологии КФУ
Протокол № 1 от 7 февраля 2017 г.

заседания кафедры биохимии и биотехнологии
Протокол № 7 от 7 февраля 2017 г.

Рецензенты:
Кандидат биологических наук, доцент **Д.А.Темников**
Кандидат биологических наук, доцент **А.Р.Каюмов**

Акберова Н.И.

Основы анализа данных и программирования в R: учебно-методическое пособие / Н.И. Акберова, О.С. Козлова. – Казань: Альянс, 2017. – 33 с.

Программная среда вычислений и язык программирования R являются «золотым стандартом» современных статистических вычислений, обладающим практически неограниченными возможностями визуализации и разведочного анализа. По своей эффективности в работе исследователя-экспериментатора этот универсальный и мощный инструмент намного превышает уровень многочисленных программ обработки электронных таблиц, что делает его незаменимым для анализа, в первую очередь, больших объёмов многомерных данных. В учебно-методическом пособии рассматриваются базовые методы обработки результатов экспериментов с использованием свободной программной среды вычислений R и графической оболочки Rstudio. Каждый раздел пособия снабжён примерами запуска команд на языке R, иллюстрирующими широкие возможности данной среды, а также набором вопросов и задач для самостоятельного решения. Пособие направлено на формирование и развитие навыков предварительной обработки и простейшей визуализации экспериментальных данных, кроме того, особое внимание уделено рассмотрению таких универсальных конструкций языков программирования, как циклы и ветвления.

Учебно-методическое пособие предназначено для широкого круга студентов и аспирантов, обучающихся по медико-биологическим направлениям и нуждающихся в достоверной, оперативной и удобной для извлечения новой информации обработке результатов экспериментов в соответствии с высокими качественными стандартами, предъявляемыми современной наукой.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Система статистических вычислений R всё повсеместно используется в научно-исследовательских и коммерчески ориентированных проектах. В связи с универсальностью и практически неограниченными возможностями данного инструмента статистической обработки, авторы считают необходимым его освоение студентами и аспирантами естественнонаучных факультетов, в частности, обучающимися по направлениям медико-биологического цикла.

Прекрасно понимая все затруднения, с которыми сталкиваются студенты – будущие биологи и медики – при необходимости статистической обработки данных для курсовой или дипломной работы, авторы постарались в доступной форме изложить фундаментальные, базовые понятия языка R. Именно этим обусловлена структура методического пособия, последовательно переходящего от наиболее простых типов представления экспериментальных данных (векторы значений переменной) к более сложным (таблицам data frame, в которых объединено сразу несколько векторов значений переменных разных типов).

Двум основным составляющим разведочного анализа данных – описательным статистикам и простейшему графическому представлению информации – посвящены следующие разделы методического пособия. При этом среди всего многообразия функций языка R авторами отобраны те, которые оптимально совмещают в себе простоту использования и наглядность представления результатов. Таким образом, набор описываемых в рамках данного методического пособия функций является тем необходимым и достаточным минимумом, которым составляет инструментарий современного исследователя данных в среде статистических вычислений R.

Последняя, пятая тема посвящена реализации в R универсальных конструкций языков программирования – ветвлений и циклов. Эти фундаментальные операторы могут быть широко использованы на практике – в частности, для выполнения преобразования типов переменных. Перевод количественной переменной в номинативный предиктор – задача, которая нередко возникает при построении различных статистических моделей, и отработке этого навыка авторы посвятили насколько задач и контрольных тестов.

ТЕМА I. РАБОТА С ВЕКТОРАМИ

Интерфейс программы Rstudio

Пользовательский интерфейс программы Rstudio состоит из четырёх областей (окон):

- Верхнее левое – окно редактора скриптов (при его отсутствии вызов производится при помощи команд главного меню File – New file – R script). В этом же окне можно просматривать созданные в ходе работы таблицы – data frame, а также открывать текстовые и ассоциированные с R типы файлов.
- Нижнее левое – консоль (терминал ввода-вывода). Посредством консоли происходит всё «общение» пользователя с R – ввод команд и получение результатов их выполнения.
- Верхнее правое – окно, содержащее две вкладки (Environment, демонстрирующую все созданные переменные и функции в ходе сеанса работы, и History, демонстрирующую все команды, запущенные пользователем).
- Нижнее правое – окно, которое содержит много полезных вкладок, облегчающих работу пользователя:

- вкладку Files (файловый менеджер, позволяющий перемещаться по папкам компьютера, не выходя из Rstudio)
- вкладку Plots, отображающую все графики, созданные в ходе работы
- вкладку Packages для управления загруженными пакетами
- вкладку Help – описание использования команд с примерами
- вкладку специальных возможностей Viewer

Переменные

Переменная – это поименованная область в памяти компьютера. Переменная – объект, имеющий имя и значение.

Задание переменной в R:

```
my_var1 <- 42
```

`my_var1` – это имя переменной, число `42` – её значение, а `<-` - знак присваивания.

После выполнения этой команды (в консоли команда выполняется по нажатию клавиши Enter, а в редакторе скриптов – по нажатию Ctrl+Enter) во вкладке Environment появится переменная и её значение.

Задание. Создать переменную my_var2 и присвоить ей значение 35.49 (десятичный разделитель – точка).

С переменными можно обращаться так же, как с числами, – складывать друг с другом и с константой, умножать и делить друг на друга и на константу (возведение в степень обозначается символом ^). Результат выполнения арифметических операций также можно сохранять в переменную.

Полезный совет! Облегчить ввод имён функций и переменных поможет клавиша Tab, нажатая после начала ввода имени функции или переменной (она предложит автодополнение, то есть покажет, какие имена функций или переменных начинаются с тех же символов).

Важное ограничение на имена: имена в R всегда начинаются с буквы, а не с цифры, при этом Name и name – это разные имена. Использование пробелов в имени **недопустимо**.

Задание. Сложить переменные my_var1 и my_var2, из суммы вычесть 12 и сохранить результат выполнения этих операций в переменную my_var3.

Немного о типах переменных. Помимо имени и присваиваемого значения, каждая переменная имеет определённый тип. Основные типы – численные данные (пример – `a<-5.98`), символьные строки (пример – `name<-"Mary"`, обязательно в кавычках), логический тип и data frame.

Логические операции

Переменные и константы можно не только использовать в арифметических операциях, но и выполнять с ними логические операции (это операции сравнения, а также проверки неравенств). Результат выполнения логических операций – это ИСТИНА или ЛОЖЬ (`TRUE` или `FALSE`).

Пример операции сравнения:

```
my_var3 > 200
```

Выполнение данной операции возвращает `TRUE`, если значение переменной my_var3 больше 200, и `FALSE`, если значение этой переменной меньше или строго равно 200. Как и результаты арифметических операций, результаты выполнения логических операций можно сохранять в переменные.

Чтобы проверить строгое равенство, используют двойной знак ==. «Неравно» (перечёркнутый знак равенства) кодируется как `!=`.

Задание. Создать три переменных - number_1, number_2 и number_3. Присвоить им какие-нибудь числа (например, пусть будут представлены целые и дробные числа, отрицательные и положительные). Проверить, действительно ли сумма первых двух переменных больше третьей. Результат такого сравнения сохранить в переменную result.

Векторы

R – векторизованный язык. Это значит, что все переменные в нём трактуются как вектора (переменная, хранящая только лишь одно численное значение, – вектор длиной 1). Все операции, справедливые для векторов длины 1, распространяются и на более длинные векторы. Чтобы создать вектор, хранящий все целые числа от a до b, используют знак двоеточия (`:`). Так, команда `1:10` выведет в консоль все целые числа от 1 до 10. Этот подход можно использовать и для создания «нисходящих» последовательностей – например, `10:1` выведет числа в обратном порядке.

Важно! Все элементы векторов должны быть одинакового типа (только числовые, только логические или только символьные).

Задание. Создать переменную my_vector1 и присвоить ей последовательность целых чисел от 1 до 55.

Чтобы объединять в вектор произвольные числа, необходимо использовать функцию `c()` – от слова combine (объединить). Внутри скобок через запятую пишут числа, которые требуется объединить.

Например, команда `my_vector2<-c(-3,5,0)` создаст переменную с именем my_vector2, которая будет хранить три числа: – 3, 5 и 0.

Чтобы обратиться к отдельному элементу вектора, можно использовать его индекс – порядковый номер элемента в векторе.

Пример обращения к элементу вектора по индексу:

```
my_vector2[2]
```

Такая команда выведет в консоль второй элемент переменной под именем my_vector2, то есть число 5.

Чтобы обратиться сразу к нескольким элементам вектора (или к одному, но несколько раз) внутрь [] необходимо встроить функцию `c()`, внутри которой через запятую перечислить интересующие индексы.

Например, команда

```
my_vector2[c(2,2,3,1)]
```

выведет в консоль сначала дважды второй элемент вектора – переменной my_vector2, а затем третий и первый элемент: 5 5 -3 0.

Чтобы обратиться к последовательным элементам вектора (например, с 1го по 3й), можно использовать двоеточие (`my_vector2[1:3]`). Последовательное обращение через двоеточие можно встраивать вовнутрь функции `c()` – например, если требуется вызвать элементы вектора с первого по третий, а затем пятый и десятый, это будет выглядеть так: `my_vector2[c(1:3,5,10)]`.

Полезный совет! При работе в консоли для вызова ранее выполненных команд можно использовать стрелки «вверх» и «вниз» на клавиатуре компьютера.

Задание. Создать переменную the_best_vector, хранящую числа от 1 до 5000 и затем числа от 7000 до 10000. Затем создать переменную subvector, в которой будут храниться 2й, 5й, 7й, 9й, 12й, 16й и 20й элементы вектора the_best_vector.

К элементам векторов можно обращаться не только по индексу, но и по условию (например, если надо отобрать только те элементы вектора, которые больше пяти). В этом помогут логические высказывания, ведь с векторами можно обращаться так же, как с

переменными, которые хранят всего лишь одно число. Так, команда `my_vector1>50` для **каждого элемента вектора** проверит, больше ли он, чем 50, а так как переменная my_vector1 хранит все целые числа от 1 до 55, результат выполнения этой команды – **вектор (!)** из 55 элементов, первые 50 элементов которого хранят `FALSE`, а последние 5 – `TRUE`.

И вот теперь этот новый логический вектор можно передать в квадратные скобки вместо индексов в явном виде (первый, второй, третий…). Команда будет выглядеть так:

```
my_vector1[my_vector1>50]
```

Эта команда выведет в консоль только те элементы вектора my_vector1, которые больше 50.

Условия можно комбинировать с помощью знака &. Так, команда

```
my_vector1[my_vector1>40 & my_vector1<50]
```

выведет в консоль только те элементы вектора my_vector1, которые больше 40, но меньше 50.

Задание. Создать вектор height, хранящий результаты измерения роста в сантиметрах у 9 человек: 165, 178, 180, 181, 167, 178, 187, 167, 187. Отобрать в переменную greater_than_mean только те значения роста, которые больше среднего по выборке. Чтобы найти среднее, удобно воспользоваться функцией `mean()`, которая принимает на вход имя вектора.

Задание. Создать переменную my_vector и записать в неё все целые числа от 1 до 20. Найти сумму всех элементов вектора, которые больше 10. Сохранить сумму в переменную my_sum. Чтобы найти сумму, удобно использовать функцию `sum()`, которая принимает на вход имя вектора.

Задание. Создать вектор из 100 случайных, нормально распределённых чисел (функция `rnorm(n,mean= … ,sd= …)`, выбрав либо стандартное нормальное распределение (указав только размер выборки n), либо нестандартное нормальное распределение (указав и размер выборки n, и среднее mean, и стандартное отклонение sd)). Отобрать из этого вектора только такие значения, которые отклоняются от среднего не более чем на одно стандартное отклонение (помогут функции `mean()` – среднее, `sd()` – стандартное отклонение). Сколько получилось таких значений?

Немного о функциях. Помимо переменных, в R есть ещё и функции – это некая последовательность операций, которая выполняется над одной или несколькими переменными и/или константами. Любая функция получает на вход один или несколько параметров (они же называются аргументами), часть которых может идти по умолчанию и не указываться пользователем, а на выходе функция выдаёт некий результат. Все параметры, передаваемые в функцию пользователем, указываются внутри круглых скобок **через запятую**. Простейший пример функции – `c()`, также в виде функций реализованы простые математические функции – например, функция логарифма (`log()`, по умолчанию –

логарифма натурального), функция максимума (`max()`) и функция квадратного корня (`sqrt()`).

Полезный совет! Чтобы получить справку по интересующей функции, следует написать её имя в консоли с предшествующим знаком ?. Например, `?abs` выдаст справку об использовании модуля, а заодно и квадратного корня `sqrt`.

ТЕМА II. РАБОТА С ТАБЛИЦАМИ

Тип данных Data frame

Самый распространённый тип переменных для хранения результата эксперимента имеет вид таблицы и называется data frame. В столбцах такой таблицы расположены переменные, а в строках – наблюдения. Чтобы создать data frame «с нуля», используется команда `data.frame()`, принимающая на вход имена столбцов и присваиваемые этим столбцам вектора.

```
df<-data.frame(name=c("Olga","Maria","Elena","Vera"),
               age=c(20,25,22,28),
          is_married = c(TRUE,FALSE,FALSE,TRUE))
```

Эта команда (другими словами – функция) создаёт таблицу (data frame) с именем df и тремя столбцами (первый – имя, второй – возраст, третий – замужем девушка или нет). Поскольку все входящие вектора состоят из четырёх элементов, в таблице будет четыре строки. Обратим внимание, что все строки представляют собой вектора разных типов – первый хранит символьные строки (имена, они в кавычках), второй – числа, третий – логические выражения `TRUE` или `FALSE`.

Чтобы загрузить в R уже готовую таблицу с данными, используются следующие функции:

`read.table("имя_файла_в_кавычках")` – если файл имеет расширение *.txt

`read.csv("имя_файла_в_кавычках")` – если файл имеет расширение *.csv

Интересно, что в качестве параметра данным функциям можно передавать не только имя файла, который хранится на компьютере пользователя, но и ссылку на него в сети Интернет.

Важно помнить, что любой файл имеет имя, адрес (другими словами, путь) и расширение. Адрес (путь) – это перечисление имён вложенных папок, хранящих файл на компьютере или в сети Интернет, через знак слэш (обычно "/", реже "\"). Расширение файла – это сокращённое указание на формат этого файла, отделяемое от его имени точкой. Так, если файл называется abc.txt, это указывает на то, что он текстовый, а если abc.csv – что это файл, представляющий собой таблицу, все значения в столбцах которой отделены друг от друга запятой (реже – другими символами, например, ";"). Формат *.csv очень популярен во многих приложениях, поскольку является универсальной формой представления таблиц удобным для программистов образом.

Важный параметр функций `read.table()` и `read.csv()`, помимо самого имени файла в кавычках, – header, то есть заголовок. Этот параметр логический, то есть он принимает на вход либо `TRUE`, либо `FALSE`. Этот параметр указывать необязательно, но, так как исходный файл для обработки может иметь или не иметь заголовок, важно правильно сообщить R об этом, и, вызывая функцию, явным образом писать, есть ли в данных заголовок или его нет. Если в таблице заголовка нет, параметру header присваивается значение `FALSE`, а если таблица имеет заголовок, то `TRUE` (`header=FALSE` и `header=TRUE`, соответственно).

Если файл не загружается:

- Убедитесь, что вы указали его расширение
- Убедитесь, что имя файла с расширением взято в кавычки
- Убедитесь, что загружаемый файл находится в рабочей директории

В каждый конкретный момент времени R «видит» только одну папку, которая называется рабочей директорией. Чтобы сделать папку, в которой лежит файл, рабочей, выполните следующие действия:

1) Во вкладке Files (правое нижнее окно) нажмите на кнопку с тремя точками (справа)

2) В открывшемся окне «Обзор папок» найдите папку, в которой лежит файл, выделите её и нажмите OK

3) Убедившись, что во вкладке Files появилось содержимое выбранной папки, нажмите кнопку More

4) В выпадающем меню выберите пункт Set as working directory, после чего в консоли должна появиться команда `setwd("полный_адрес_папки")`.

Ещё один параметр функций `read.table()` и `read.csv()` – `sep` (от слова separator – разделитель). Он указывает R, каким знаком столбцы таблицы отделены друг от друга. Если для этого использовался какой-либо нестандартный символ, например, точка с запятой, при запуске команды необходимо добавить внутрь скобок ещё и `sep=";"`. Аргумент функций `dec` позволяет указать, какой знак использовался для отделения целой части десятичной дроби от дробной (по умолчанию – точка).

Пример работы функции `read.csv()` с параметрами по умолчанию:

```
my_data<-
read.csv("https://www.openintro.org/stat/data/evals.csv")
```

Данная команда не просто загружает в среду R файл из Интернета, но и создаёт объект для его хранения с именем my_data. Этот объект будет иметь тип data frame.

Вопрос. Сколько наблюдений и сколько переменных (столбцов) в данной таблице?

В этой таблице содержатся результаты оценки студентами университета Остина, штат Техас, своих преподавателей по окончанию семестра. Каждый преподаватель был оценён по

пятибалльной шкале, и изучалось влияние на эту оценку множества разных факторов – возраста, пола преподавателя, его национальности, должности в университете и т.п. Чтобы посмотреть на данные в удобном виде, можно нажать на значок «табличка» справа от имени таблицы на вкладке Environment или же вызвать функцию View (с большой буквы), передав ей на вход имя таблицы (my_data, в данном случае).

Полезные функции по работе с data frame:

Функция `names(имя_d.f.)` – возвращает вектор имён столбцов таблицы

Функция `str(имя_d.f.)` – показывает **структуру** таблицы с указанием типов переменных-столбцов

Функция `summary(имя_d.f.)` – показывает **отчёт** по таблице – некую сводку по каждому её столбцу

Вопрос. Сколько строк таблицы относится к мужчинам, а сколько – к женщинам?

Немного о структуре данной таблицы. В ней среди 21 переменной (столбцов) имеются переменные типа int (целочисленные) – это возраст (age), cls_students (число студентов в классе) и некоторые другие. Кроме того, есть переменные, представленные десятичными дробями (num), – это score (оценка преподавателя студентами) и cls_perc_eval (процент студентов в классе, поставивших преподавателю оценку). Есть и третий тип – факторные переменные.

Факторные переменные

Факторные переменные (факторы) – это разновидность качественной переменной (два основных типа переменных в статистике – это качественные, или номинативные, и количественные). Однако, по сравнению с простыми качественными переменными (например, с переменной «Имя человека»), факторы отличаются тем, что они описывают некий **класс**, к которому принадлежит наблюдение. Этот класс по-другому называется **градацией** или **уровнем фактора**. Простой пример – переменная «Пол человека», которая имеет всего две градации – «мужчина» или «женщина».

Примечание. По умолчанию, все символьные переменные в data frame считаются факторными.

Работа с таблицами типа Data frame

Для того чтобы извлекать из большой таблицы конкретные области данных, существует три способа.

Способ I – «через доллар».

Чтобы обратиться к конкретному столбцу data frame, используется знак доллара.

Формат:

```
имя_df$имя_столбца
```

Этот способ записи поможет также, если необходимо уже в имеющейся таблице создать новый столбец. В этом случае после имени столбца (нового!) надо поставить знак присваивания и передать вектор, который должен стать новым столбцом.

Задание. В таблице my_data создать ещё один столбец с именем score_new, который будет хранить оценки преподавателей не по 5ти-балльной, а по 10ти-балльной шкале.

Задание. В таблице my_data создать ещё один столбец с именем number, который будет хранить номера строчек таблицы.

Примечание. Функция `nrow(имя_df)` возвращает число строк в таблице. Функция `ncol(имя_df)` возвращает число столбцов в таблице.

Способ II – «через индекс».

Индекс – это число, отражающее порядковый номер столбца или строки в таблице. Подобно тому, как при работе с векторами из них можно извлекать конкретные значения, помещая индексы в квадратные скобки, так и из таблиц можно извлекать значения по их точному «адресу», вот только индексов будет два: **первый – по строкам, второй – по столбцам.**

Например, запись `my_data[2,3]` извлечёт в консоль элемент таблицы, стоящий на пересечении второй строки и третьего столбца.

Задание. Вывести в консоль содержимое ячеек таблицы my_data, стоящих на пересечении 2й, 193й и 225й строк с первым столбцом.

Задание. Вывести в консоль содержимое ячеек таблицы my_data, стоящих на пересечении строк с номерами с 50 по 55ую и 2го столбца.

Важно, что один из индексов можно и вовсе оставлять пустым. Так, запись `my_data[,5]` выведет в консоль содержимое **всех строк** для пятого столбца, а запись `my_data[2,]` – содержимое **всех столбцов** для второй строки.

Важно! Чтобы вывести содержимое всех столбцов, **кроме некоторых**, используется запись `my_data[,-c(некоторые_столбцы)]`. Так, команда `my_data[,-c(3,5)]` выведет в консоль содержимое всех строк всех столбцов, кроме 3го и 5го. Этот же способ может быть использован для вывода всех строк, кроме некоторых.

Задание. Создать переменную mini_data, хранящую только столбцы score и gender.

Способ III – «через условие».

Вариант A

По идее, это комбинация двух первых способов, так как здесь будут присутствовать как $, так и [].

Поставим задачу – отобрать оценки только для преподавателей-женщин. Во-первых, выберем столбец, хранящий пол преподавателя, с помощью команды `my_data$gender` –

это будет вектор из 463 элементов. Во-вторых, зададим условие: `my_data$gender=="female"` – результат такой команды тоже будет представлять собой вектор, только уже логический (`TRUE` там, где преподаватель – женщина, и `FALSE` – там, где мужчина). Наконец, вставим этот логический вектор внутрь квадратных скобок на место первого индекса, а на место второго поставим 1, ведь нам нужны баллы для таких преподавателей:

```
my_data[my_data$gender=="female",1]
```

Задание. Найти среднее и стандартное отклонение возраста преподавателей-мужчин.

Вариант B

В R есть специальная функция, которая позволяет реализовать способ 3 альтернативным путём, – она называется `subset()`. Первый параметр функции – имя data frame, второй – условие:

```
subset(my_data,gender=="female")
```

Данная команда выводит в консоль некую подтаблицу исходной таблицы, содержащую только информацию, касающуюся преподавателей-женщин. Чтобы вывести их оценки, надо добавить доллар:

```
subset(my_data,gender=="female")$score
```

Задание. Найти среднее и стандартное отклонение возраста преподавателей-мужчин с помощью функции `subset()`.

Задание. Посмотреть распределение по полу преподавателей, имеющих оценку выше среднего по общему объёму наблюдений.

Примечание. В R есть удобная функция, которая возвращает распределение значений элементов вектора по их количеству – `table(вектор)`. Например, чтобы узнать, каково распределение преподавателей по полу в исходной таблице, следует выполнить команду `table(my_data$gender)`. В функцию можно передавать и data frame из нескольких столбцов, например, `table(my_data[,4:5])` вернёт таблицу сопряжённости – распределение преподавателей сразу по двум признакам: полу и языку.

Объединение таблиц по строкам и по столбцам

Функция `rbind()` позволяет объединить два data frame в один по строкам (как бы «подклеить» одну таблицу под другую). Функция `cbind()` позволяет объединить два data frame в один по столбцам (как бы «подклеить» вторую таблицу к первой справа).

Создадим data frame, содержащий информацию о мужчинах:

```
male<-subset(my_data,gender=="male")
```

Создадим data frame, содержащий информацию о женщинах:

```
female<-subset(my_data,gender=="female")
```

«Подклеим» data frame female под male:

```
male_plus_female<-rbind(male,female)
```

Создадим data frame, хранящий все столбцы с 1го по 10й:

```
first<-my_data[,1:10]
```

Создадим data frame, хранящий все столбцы с 11го по последний:

```
second<-my_data[,11:ncol(my_data)]
```

«Склеим» две половинки между собой:

```
first_plus_second<-cbind(first,second)
```

ТЕМА III. ОПИСАТЕЛЬНЫЕ СТАТИСТИКИ

Продолжим работу с базой данных evals.csv, загруженной в R под именем my_data.

```
my_data<-
read.csv("https://www.openintro.org/stat/data/evals.csv")
```

Основные описательные статистики

1. Медиана – функция `median(имя_вектора)`

2. Среднее арифметическое – функция `mean(имя_вектора)`

3. Стандартное (среднее квадратическое) отклонение – функция `sd(имя_вектора)`

4. Размах (минимальное и максимальное значение выборки) – функция `range(имя_вектора)`

Задание. Создать четыре переменных, хранящих медиану, среднее, стандартное отклонение и размах оценок преподавателей из таблицы my_data.

Задание. Найти среднее арифметическое оценки преподавателей-мужчин в возрасте от 30 до 50 лет (включительно).

Задание. Найти медиану возраста преподавателей-женщин, имеющих оценки от 4 до 5 (включительно).

Расчёт описательных статистик для совокупности поднаборов данных

Функция `aggregate()` позволяет рассчитать описательные статистики сразу для нескольких поднаборов данных.

Пример её запуска:

```
aggregate(my_data$score,list(my_data$gender),mean)
```

Первый аргумент функции `aggregate()` – вектор (выборка), в отношении которой считаются описательные статистики. Второй аргумент – **список** группирующих векторов, то есть тех переменных, в зависимости от которых первый вектор будет делиться на группы. Третий аргумент – само название описательной статистики, которую требуется рассчитать. В данном случае команду следует понимать так: рассчитываются средние значения оценок преподавателей отдельно в каждой из двух групп, указанных в переменной gender (отдельно для мужчин и отдельно для женщин).

Группирующих векторов-переменных может быть и больше, в данном случае каждый следующий вектор следует вписать внутрь скобок `list()` через запятую.

Вопрос. У преподавателей какого пола средняя оценка выше?

По умолчанию функция `aggregate()` создаёт объект в формате data frame.

Задание. Создать объект statistics, содержащий в себе медианы оценок преподавателей, разделённых на группы по признаку пола и языка (англоговорящие/не англоговорящие). Сколько строк в получившейся таблице?

Полезный совет! Чтобы изменить заголовок таблицы, поможет функция `colnames(имя_data_frame)`. Так, чтобы изменить заголовок таблицы из предыдущего задания, надо вызвать команду `colnames(statistics)<-c("gender","language","median")`.

Чтобы рассчитать описательную статистику для нескольких переменных сразу, следует воспользоваться квадратными скобками. Так, команда

```
aggregate(my_data[,c(1,6)],list(my_data$gender),mean)
```

рассчитает среднее арифметическое оценок (столбец номер 1) и возрастов (столбец номер 6) преподавателей внутри гендерных групп.

Задание. Рассчитать среднее арифметическое по всем столбцам, касающимся оценки внешности преподавателя студентами (столбцы начинаются с букв bty) в зависимости от пола.

Немного о типе list. Тип list (список) – ещё один тип данных в R. Отличие его от вектора – в том, что элементами списка могут быть переменные разных типов (численные, символьные, логические и так далее). В принципе, тип data frame можно воспринимать как список векторов, каждый из которых имеет одну и ту же длину. Чтобы создать объект типа list, используется функция `list()`, внутри скобок которой через запятую пишутся объекты, которые надо объединить в список.

Формула

Удобный способ передачи информации о зависимости переменных в R – формула (или модель). Общий синтаксис формулы (модели):

```
зависимая_переменная ~ независимая_(группирующая)_переменная
```

Вариант вызова функции `aggregate()` с использованием формулы:

```
aggregate(score ~ gender, my_data, mean)
```

В данной команде указывается, что вектор-переменная score зависит (делится в зависимости) от переменной gender, а также что и score, и gender, принадлежат (являются столбцами) одной и той же таблицы my_data.

Если требуется задать разбиение переменной в зависимости от нескольких других независимых переменных, в правой части формулы независимые переменные перечисляются через знак +. Например команда `aggregate(score ~ gender+rank, my_data, mean)` выведет средние баллы преподавателей в зависимости от пола и позиции в университете (rank – фактор с тремя градациями: teaching - что-то вроде «нашего» ассистента, tenure track – должность, предшествующая заключению бессрочного контракта, tenured – профессора с бессрочным контрактом). Независимые (группирующие) переменные, стоящие в правой части модели, также называются **предикторами**.

Чтобы рассчитать описательную статистику для нескольких переменных сразу, можно воспользоваться функцией `cbind()`. Например, рассчитать среднее арифметическое оценок и возраста преподавателей в зависимости от пола можно с помощью следующей команды:

```
aggregate(cbind(score,age) ~ gender, my_data, mean)
```

Задание. С использованием формулы рассчитать стандартные отклонения оценок и возрастов преподавателей в зависимости от пола и языка. Результат сохранить в переменную result.

Пропущенные значения

Далеко не всегда в data frame для исследования заполненными будут абсолютно все ячейки. В R пропущенные значения традиционно кодируются буквами `NA` (от not available, данные не доступны). Чтобы проверить, есть ли в таблице пропущенные значения, пользуются функцией `is.na(имя_вектора)`. Данная функция сравнивает **каждое значение** вектора с `NA` и возвращает `TRUE`, если на данном месте стоит `NA`, и `FALSE`, если значение не пропущено. Соответственно, функция `is.na()` возвращает также вектор, размерность которого соответствует размерности вектора, поданного на вход, только этот вектор логического типа (в нём только `TRUE` и `FALSE`).

Интересно, что `TRUE` и `FALSE` воспринимаются R и в численной шкале: так, `TRUE` – это 1, а `FALSE` – 0. Таким образом, чтобы быстро понять, имеется ли в векторе хотя бы одно `NA`, можно сложить все элементы вектора, который возвращает команда `is.na()`, между собой. Если команда `sum(is.na(имя_вектора))` вернёт 0, значит, ни одного пропущенного значения в векторе нет, а если она вернёт число, большее 0, значит, вектор содержит столько пропущенных значений, сколько единиц содержится в этой сумме.

Большинство функций R в числе своих аргументов содержат `na.rm` – это параметр, указывающий, что делать с пропущенными значениями (дословно NA remove, удалить пропущенные значения). Чтобы игнорировать пропущенные значения, при вызове функции

необходимо присвоить этому аргументу значение TRUE. В функции aggregate() по умолчанию пропущенные значения игнорируются.

Функция replace() позволяет заменить пропущенные значения на что-либо более конкретное. Пример использования:

```
fixed_vector <-
replace(my_vector,is.na(my_vector),mean(my_vector,na.rm=T))
```

Данная команда создаёт вектор fixed_vector на основе вектора my_vector, содержащего пропущенные значения, заменяя каждое пропущенное значение средним арифметическим по выборке. Общий синтаксис функции replace(): первый аргумент – вектор, который нужно обработать, второй аргумент – позиции вектора, значения в которых надо заменить, третий аргумент – на что надо заменить значения вектора на данных позициях. По умолчанию функция mean() не игнорирует пропущенные значения и выдаёт NA для всех векторов, в которых есть хотя бы одно пропущенное значение. Поэтому, чтобы игнорировать такие значения и учитывать только известные элементы вектора, в функцию mean() добавляется na.rm=T.

Примечание. TRUE и FALSE можно сокращать до одной буквы (T и F, соответственно).

ТЕМА IV. ПОСТРОЕНИЕ ГРАФИКОВ

Гистограмма

Первый способ визуализировать количественную переменную (выборку, вектор наблюдений) – построить гистограмму её распределения. В R для этого используется функция, которая называется hist(имя_вектора) – от слова histogram, гистограмма. Единственный обязательный аргумент данной функции – имя той переменной, гистограмма распределения которой должна быть построена. Пример использования функции:

```
hist(my_data$score)
```

Эта команда создаст во вкладке Plots график, отражающий распределение оценок, выставленных студентами своим преподавателям (набор данных evals.csv).

Некоторые полезные параметры функции hist():

- breaks – либо вектор, задающий точки разбиения оси X, либо одно число, задающее число таких разбиений, либо функция для расчёта точек разбиения или числа разбиений
- freq – либо =TRUE для отображения по оси Y абсолютных частот, либо =FALSE для отображения по оси Y относительных частот
- main="Заголовок_гистограммы" для задания названия графика
- col="Цвет" для задания цвета столбцов гистограммы

- `xlab="Подпись_оси_X"` и `ylab="Подпись_оси_Y"` для задания подписей осей X и Y, соответственно

Например, команда

```
hist(my_data$score,breaks=c(2.3,2.5,3,3.5,4,4.5,5),freq=FALSE,
    main="Гистограмма оценок",col="red",xlab="Оценка, балл",
                        ylab="Доля")
```

создаст гистограмму относительных частот распределения оценок студентов преподавателям, в которой левой границей интервалов разбиения оси X будут числа 2.3, 2.5, 3, 3.5, 4, 4.5, при этом правой границей последнего интервала будет число 5. Данная гистограмма будет иметь столбцы красного цвета и называться «Гистограмма оценок», при этом ось X будет подписана «Оценка, балл», а ось Y – «Доля».

Вопрос. Сколько столбцов будет в данной гистограмме?

Примечание. Используя таким образом параметр breaks*, необходимо убедиться в том, что первый и последний элементы передаваемого вектора соответствуют минимальному и максимальному наблюдаемым значениям для выборки.*

Боксплот, или коробчатый график

Визуально оценить, охарактеризовать распределение одной количественной переменной или сразу нескольких поможет боксплот (коробчатый график, или ящик с усами). В R создание графиков такого типа реализовано функцией `boxplot()`, которая принимает на вход один или несколько количественных векторов через запятую, либо модель с одним или несколькими предикторами. Кроме того, в функцию `boxplot()` можно передать и название data frame, и тогда для каждого столбца таблицы будет создан свой боксплот, при этом все они будут размещены на одном графике.

Пример построения коробчатого графика для одной переменной:

```
boxplot(my_data$score)
```

После выполнения данной команды во вкладке Plots появится график с усами, визуализирующий распределение оценок преподавателей. Интерпретация графика следующая:

- Жирная линия, проходящая внутри ящика, - это медиана.
- Сам ящик (прямоугольная область) отображает расстояние между 1м и 3м квартилями, таким образом, ровно половина всех наблюдений (центральная половина упорядоченной выборки) находится внутри числового диапазона, ограниченного ящиком. Высота ящика называется межквартильным (интерквартильным) размахом.
- Усы (штрихованные линии с ограничителем, отходящие вверх и вниз от ящика) показывают степень отклонения крайних значений выборки от 1го (внизу) и 3го (вверху) квартилей, соответственно. Концы усов показывают наиболее удалённые от верхней и нижней границ ящика значения, расстояние между которыми и границами ящика не

превышает полутора межквартильных размахов (по умолчанию, однако этот множитель можно менять с помощью параметра `range` функции `boxplot()`).

- Отдельно стоящие точки на графике – это **выбросы**. Так называют экстремально отстоящие от медианы значения, то есть те значения, расстояние между которыми и границами ящика (то есть 1м и 3м квартилями) превышает полтора межквартильных размаха (по умолчанию).

Вопрос. Чему равен межквартильный размах в выборке оценок преподавателей?

Задание. Убедиться в том, что выбросы на графике действительно отклоняются от границы ящика более чем на полтора межквартильных размаха.

Полезный совет! В R есть функция, позволяющая сортировать вектор количественных значений по возрастанию или по убыванию – `sort(имя_вектора)`. В числе аргументов этой функции – параметр decreasing, принимающий значение `TRUE`, если требуется сортировка по убыванию, и `FALSE` – если по возрастанию. Кроме того, сортировать таблицы по убыванию/возрастанию либо по алфавиту можно в режиме просмотра (View), с помощью стрелок в области заголовков.

Пример построения сразу двух боксплотов на одном графике с использованием формулы:

```
boxplot(score~gender, my_data)
```

Данная команда позволяет визуализировать сразу два коробчатых графика в одних осях (первый график соответствует оценкам для женщин, второй – оценкам для мужчин).

По аналогии с гистограммой, в функцию `boxplot()` можно передавать и другие аргументы. Например, добавление аргумента `ylab="Scores"` создаст подпись оси Y.

Диаграмма рассеяния

Наиболее общая (generic) функция R для создания графиков – `plot()`. В том случае, если в качестве параметров ей были переданы два количественных вектора одинаковой длины (X и Y), во вкладке Plots появится так называемая диаграмма рассеяния (scatter plot), на которой каждая точка будет представлять одно наблюдение.

Пример диаграммы рассеяния, отражающей зависимость между общим числом студентов в группе (столбец cls_students) и числом студентов, принимавших участие в опросе (столбец cls_did_eval):

```
plot(my_data$cls_students, my_data$cls_did_eval, xlab="Общее
        число студентов", ylab="Респонденты")
```

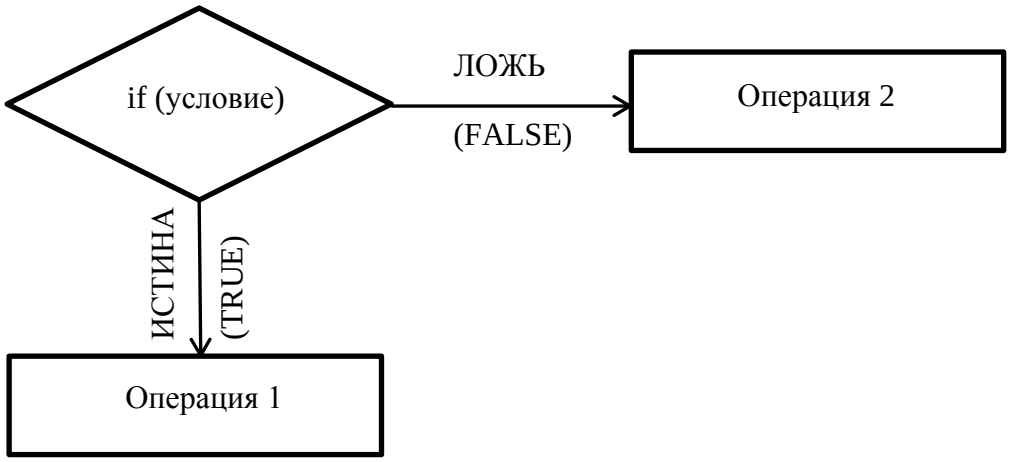
В данном случае общее число студентов будет расположено по оси X, а число студентов, принимавших участие в опросе, - по Y.

Задание. Создать вектор a, хранящий все целые числа от 1 до 10 включительно. Создать вектор b, каждый элемент которого в два раза больше соответствующего элемента a. Создать диаграмму рассеяния, отражающую зависимость переменной b от a.

ТЕМА V. УСЛОВНЫЕ ОПЕРАТОРЫ И ЦИКЛЫ

Условные операторы

Условный оператор if (если) проверяет некое условие на истинность, и если оно истинно, выполняется одна операция, а если ложно – другая (либо не выполняется вообще ничего):



Допустим, предварительно была создана переменная a, хранящая какое-либо число. Следующий код проверяет, является ли это число строго положительным, и вычитает из него 5, если это так, и прибавляет 5 в противном случае:

```
if(a>0) {a<-a-5} else {a<-a+5}
```

Вопрос. Допустим, a равно 5. Что произойдёт после первого запуска приведённой команды и после второго её запуска?

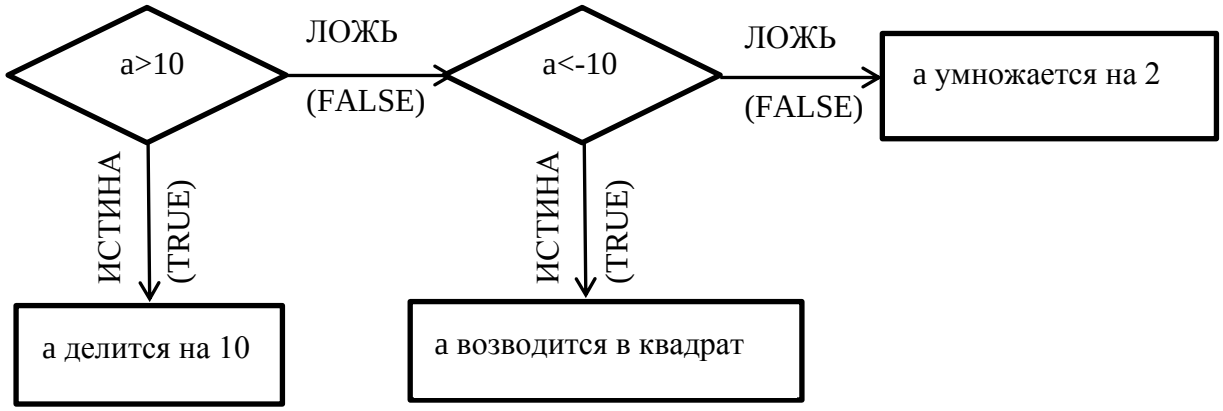
Внутри фигурных скобок может быть не одна команда, а несколько. Каждая следующая команда при этом должна начинаться с новой строки.

Условные операторы можно вкладывать один в другой. Следующий код проверяет, является ли число a строго положительным, а если это не так, проверяет, является ли оно строго отрицательным. Если и первое, и второе условия ложны, значит, число a равно 0, и в этом случае оно будет оставаться неизменным.

```
if(a>0) {a<-a-5} else if(a<0) {a<-a+5} else {a<-a+0}
```

Вопрос. Допустим, a равно 5. Что произойдёт после первого запуска приведённой команды и после второго её запуска?

Задание. Закодировать следующую схему:



Ещё одна функция R, которая позволяет обрабатывать условия и более применима для векторов – `ifelse()`. Её синтаксис таков:

```
ifelse(условие, действие_если_TRUE, действие_если_FALSE)
```

По аналогии с `if()`, несколько функций `ifelse()` могут быть вложены друг в друга. Пример использования функции:

```
a<-1:10

ifelse(a<4, "мало", ifelse(a>=4 & a<8, "достаточно","много"))
```

Входными данными для команды `ifelse` будет являться вектор, хранящий целые числа от 1 до 10 включительно. Для каждого элемента вектора сначала проверяется, меньше ли он, чем 4, и если это так, формируется символьная строка "мало", если значение не меньше 4, проверяется второе условие, и если значение меньше восьми, формируется символьная строка "достаточно", а если не меньше, то "много". Таким образом, результатом выполнения команды будет вектор, содержащий столько же элементов, сколько элементов имеет исходный количественный вектор.

Задание. Создать в таблице, хранящей данные оценок для преподавателей, ещё один столбец – age_group, хранящий "young", если возраст преподавателя меньше 40, "mature", если возраст преподавателя не меньше 40, но меньше 60, и "senior" во всех остальных случаях. Сколько человек принадлежит к каждой категории?

Циклы

Циклы позволяют выполнять некую операцию несколько раз – иногда это количество операций строго определено, а иногда нет, и операция в цикле выполняется до тех пор, пока истинно некое условие. Простейший пример цикла:

```
a<-1

for (i in 1:100) {a<-a+1}
```

Операция, стоящая внутри фигурных скобок, будет повторяться до тех пор, пока **счётчик** (i) принимает значения от 1 до ста включительно, при этом с каждым проходом (итерацией) значение i увеличивается на единицу.

Вопрос. Чему будет равно a по окончании работы кода?

Внутри фигурных скобок можно обрабатывать не только изолированные значения, но и вектора. Пример:

```
a<-1:10

for (i in 1:10) {a[i]<-a[i]+i}
```

В данном примере к каждому элементу вектора a будет прибавляться соответствующее значение индекса i: к первому 1, ко второму – 2 и так далее.

Вопрос. Чему будет равно максимальное значение в данном векторе?

Чтобы создать уже в имеющейся таблице новый столбец при помощи оператора for, сначала необходимо заполнить его значениями `NA`:

```
my_data$new_variable<-rep(NA, nrow(my_data))
```

Функция `rep()` (от слова repeat, повторять) принимает на вход два параметра: первый – то значение, которое следует повторить, а второй – сколько раз будет осуществлён повтор. В данном случае повтор будет выполнен столько раз, сколько строк содержится в таблице my_data.

Задание. Создать в data frame новую переменную – quality_for_age, которая будет хранить "good", если оценка данного преподавателя выше средней оценки преподавателей с таким же возрастом, и "bad" в противных случаях.

Следующая связанная с циклами функция `while()` относится к задачам, когда точное число итераций неизвестно, и операции внутри цикла выполняются **до тех пор**, пока верно некоторое условие, стоящее внутри круглых скобок. Простейший пример:

```
a<-2

while(a<5) {a<-a+1}
```

Вопрос. Сколько раз была выполнена команда внутри фигурных скобок?

Задание. Продвигаясь последовательно по строкам таблицы с оценками преподавателей, суммировать оценки, пока сумма меньше 100. На какой строке сумма превысит 100 баллов?

ЗАКЛЮЧЕНИЕ

Рассмотренные в рамках данного методического пособия темы являются «отправной точкой» при анализе практически любых данных, полученных экспериментальным путём. Освоив описанные выше методы и функции языка R, начинающий исследователь сможет быстро получить сводку описательных статистик по данным, визуализировать распределение количественной переменной «как есть» и в зависимости от группирующей переменной, построить простейший график зависимости одной количественной переменной от другой и даже написать программу обработки данных с использованием условий и циклов. С целью отработки и закрепления полученных навыков студентам предлагаются контрольные тесты и задачи, подразумевающие обработку реальных данных из разных областей знаний.

Разумеется, описательные статистики – это всего лишь первоначальный этап использования статистических методов для обработки результатов экспериментальной деятельности. Процессу получения осмысленных и ценных статистических выводов будет посвящено следующее методическое пособие, затрагивающее такие фундаментальные для статистической науки понятия, как генеральная совокупность, уровень статистической значимости, параметрические и непараметрические критерии.

ЗАДАНИЯ ДЛЯ САМОКОНТРОЛЯ

Контрольные тесты

В контрольных тестах будет использоваться встроенный в R набор данных iris, отражающий метрические характеристики трёх видов ирисов – setosa, virginica и versicolor. Создать копию этого набора данных в окружении R можно с помощью команды `my_data<-iris`.

Тест 1

Сколько ирисов имеет длину чашелистика (Sepal.Length) менее 4.5 см?

- 9
- 4
- 1
- 5

Тест 2

Какому виду ирисов принадлежит максимальное значение длины лепестка (Petal.Length)?

- setosa
- virginica
- versicolor

Тест 3

Чему равны элементы таблицы iris, стоящие на пересечении 1,4,5,7 и 9 строки со вторым столбцом?

- 0.2 0.2 0.2 0.3 0.2
- 5.1 5.0 4.6 4.6 4.4
- 1.4 1.5 1.4 1.4 1.4
- 3.5 3.1 3.6 3.4 2.9

Тест 4

Сколько ирисов имеет ширину чашелистика (Sepal.Width) не менее 3 и не более 3.5?

- 20
- 74
- 48
- 42

Тест 5

Какая команда создаст поднабор ирисов, состоящий из цветков, длина чашелистика (Sepal.Length) которых больше 6, а ширина чашелистика (Sepal.Width) больше среднего?

- subset(iris,Sepal.Length !=6 & Sepal.Width>=median(iris$Sepal.Width))
- subset(iris,Sepal.Length >=6 & Sepal.Width>=mean(iris$Sepal.Width))
- subset(iris,Sepal.Length >=6 | Sepal.Width>mean(iris$Sepal.Width))
- subset(iris,Sepal.Length >6 & Sepal.Width>mean(iris$Sepal.Width))

Тест 6

Создать в таблице ещё один столбец, хранящий разность между длиной чашелистика (Sepal.Length) и его шириной (Sepal.Width). Сколько ирисов имеют разность, превышающую 2?

- 100
- 20
- 103
- 105

Тест 7

Создать подтаблицу ирисов, состоящую из цветов, длина лепестка (Petal.Length) которых не меньше 5, а ширина лепестка (Petal.Width) больше медианного. Сколько строк в такой таблице?

- 46
- 42
- 17
- 28

Тест 8

Чему равны среднее и стандартное отклонение длины чашелистика (Sepal.Length) у ирисов вида setosa?

- mean=6.5, sd=0.6
- mean=5, sd=0.35
- mean=5.9, sd=0.5
- mean=4.2, sd=0.5

Тест 9

Как распределяются виды ирисов среди цветков, имеющих длину лепестка (Petal.Length) выше среднего?

- setosa – 50, virginica – 50, versicolor – 50
- setosa – 0, virginica – 26, versicolor – 44
- setosa – 0, virginica – 43, versicolor – 50
- setosa – 50, virginica – 7, versicolor – 0

Тест 10

Построить гистограмму распределения длин чашелистика (Sepal.Length) по всему набору ирисов (параметры – по умолчанию). В каком диапазоне лежит наибольшее количество наблюдений?

- [5.5-6)
- [6.5-7)
- [7-7.5)
- [6-6.5)

Тест 11

Чему равны полтора межквартильных размаха длины лепестков (Petal.Length) у ирисов virginica?

- 2.4
- 3.6
- 1.1
- 1.65

Тест 12

Построить коробчатый график, отражающий зависимость ширины лепестков (Petal.Width) от вида ирисов (параметры – по умолчанию). Выбрать верные утверждения:

- В данных нет выбросов
- Вид setosa имеет наиболее узкие лепестки
- Видовое разделение ирисов объясняет 100% изменчивости ширины их лепестков
- В выборке ирисов вида versicolor присутствуют наблюдения, отклоняющиеся от медианы более чем на полтора межквартильных размаха
- Диапазоны ширин лепестков видов setosa и virginica не пересекаются между собой

Тест 13

Нарисовать гистограмму рассеяния, расположив по оси X длину чашелистиков ирисов вида setosa (Sepal.Length), а по оси Y – длину лепестков этого вида (Petal.Length). Выбрать верные утверждения:

- Существуют ирисы, имеющие разную длину лепестков для одной и той же длины чашелистиков
- Существуют ирисы, имеющие разную длину чашелистиков для одной и той же длины лепестков
- Существует ярко выраженная линейная взаимосвязь между длиной лепестка и чашелистика
- Большинство ирисов имеет длину чашелистика менее 4.5 см

Длина лепестка большинства ирисов заключена между 1.2 и 1.6 см

Тест 14

Создать столбец, хранящий «big», если все метрические характеристики ирисов больше средних по общей выборке, и «normal», в противном случае. Сколько ирисов получилось «big», а сколько «normal»?

- big – 75, normal – 75
- big – 125, normal - 25
- big – 25, normal – 125

Тест 15

Создать столбец, хранящий «big», если все метрические характеристики ирисов больше средних внутри вида, и «normal», в противном случае. Сколько ирисов получилось «big», а сколько «normal»?

- big – 30, normal – 120
- big – 7, normal – 143
- big – 50, normal – 100
- big – 75, normal – 75

Тест 16

Выбрать верные утверждения по результатам заданий 14 и 15:

- Существует 14 цветков, которые попали в категорию «выше среднего» («big») как с учётом вида, так и без его учёта
- Цветков категории «normal» без учёта деления по видам меньше, чем с учётом деления на виды
- Существует 16 ирисов, которые были признаны «big» без учёта вида, но «normal» с учётом разделения по видам
- Существует 16 ирисов, которые были признаны «big» с учётом разделения по видам, но «normal» без учёта видов

Контрольные задачи

1. Используя встроенный в R набор данных chickwts, содержащий информацию о весе цыплят в граммах, выкормленных на разных кормах, составить таблицу описательных статистик вида:

Feed	Mean weight	SD of weight	Median weight	Min weight	Max weight

1a. Построить коробчатый график для веса цыплят и определить, какому типу корма соответствует цыплёнок с минимальным и максимальным весом. Охарактеризовать распределение весов цыплят для типа корма, которому принадлежит цыплёнок с максимальным весом.

1b. Добавить к своей копии таблицы chickwts столбец «weight in kilos» (вес в килограммах) и «weight in pounds» (вес в фунтах, 1 килограмм = 2.2 фунта).

1c. Сравнить внутригрупповые средние веса цыплят и средний вес без учёта деления на группы (в граммах). Определить, у цыплят с каким типом откорма средний внутригрупповой вес превышает общий средний, а с каким – является меньше общего среднего. Добавить в таблицу описательных статистик столбец «Greater than mean», который будет содержать «Yes», если средний вес цыплят в данной группе больше общего среднего, и «No», в противном случае.

2. Используя встроенный в R набор данных CO2, содержащий информацию о темпах поглощения двуокиси углерода растением *Echinochloa crus-galli* (ежовник обыкновенный), произрастающем в Квебеке и Миссисипи, в зависимости от концентрации CO2 в окружающем воздухе и того факта, было ли растение охлаждено накануне проведения эксперимента или нет, составить таблицу описательных статистик вида:

Type	Treatment	Mean uptake	SD of uptake	Median uptake	Min uptake	Max uptake

2a. Построить коробчатый график для зависимости темпов поглощения двуокиси углерода от фактора происхождения растения (Type) и того факта, было ли оно предварительно охлаждено (Treatment). Охарактеризовать зависимость темпов поглощения CO2 от этих двух факторов и определить, какой из них, предположительно, определяет большую изменчивость признака.

2b. Построить коробчатый график для зависимости темпов поглощения двуокиси углерода от его концентрации (conc) и охарактеризовать данную зависимость.

2c. Найти растения с минимальным и максимальным средним темпом поглощения двуокиси углерода.

2d. Заменить в своей копии таблицы CO2 значения в столбце Type на tolerant/non-tolerant (tolerant – для растений, менее чувствительных к холоду, чем растения non-tolerant).

3. Используя команду source ("http://www.openintro.org/stat/data/present.R"), сохранить фрейм данных под названием present с сайта OpenIntro. Этот массив данных содержит количества рождений мальчиков и девочек с 1940 по 2002 год в США.

3a. Рассчитайте абсолютные различия между количеством мальчиков и девочек, родившихся в каждом году, и определите, в каком году была самая большая абсолютная разница в количествах новорожденных девочек и мальчиков?

3b. На одном графике отразите динамику рождений мальчиков и девочек. На основе графика определите, справедливо ли утверждение, что каждый год рождалось больше девочек, чем мальчиков.

3c. Постройте график доли мальчиков с течением времени. На основе графика определите, справедливо ли утверждение , что доля мальчиков, родившихся в США, уменьшилась с течением времени.

3d. Для каждого года рассчитайте абсолютные различия между количеством мальчиков и девочек, и определите, в каком году была самая большая абсолютная разница в количествах новорожденных девочек и мальчиков?

4. Система Behavioral Risk Factor Surveillance System (BRFSS) является ежегодным телефонный опросом 350 000 человек в США. Как следует из названия, BRFSS предназначена для выявления факторов риска для здоровья среди взрослого населения и выявления новых тенденций. Например, респондентам задают вопросы об их диете и еженедельной физической активности, статуса ВИЧ -инфицирования, об употреблении табака, а также об уровне медицинского обеспечения. На веб-сайте BRFSS (http:// www.cdc.gov/ brfss) содержится полное описание исследования, в том числе научно-исследовательских вопросов, которые мотивируют этот проект. Мы сосредоточимся на случайной выборке в 20000 человек из опроса BRFSS, проведенного в 2000 году.
Используя команду source(http://www.openintro.org/stat/data/cdc.R), загрузите в рабочее пространство базу данных cdc, в которой представлены переменные: genhlth, exerany, hlthplan, smoke100, height, weight, wtdesire, age, и gender. Каждая из этих переменных соответствует вопросу, который был задан в опросе. Например, для genhlth, респондентам было предложено оценить их общее состояние здоровья, отвечая либо отлично, очень хорошо, хорошо, удовлетворительно или плохо (excellent, very good, good, fair or poor). Переменная exerany указывает, делал ли респондент в прошлом месяце физические упражнения (1) или нет (0). Переменная hlthplan указывает, имеет ли респондент ту или иную форму медицинского страхования (1) или нет (0). Переменная smoke100 указывает, выкурил ли респондент по крайней мере 100 сигарет в своей жизни. Другие переменные записывают рост респондента в дюймах (height), вес в фунтах(weight), а также их желаемый вес (wtdesire), возраст (age) и пол (gender)

4a. Создайте сводную таблицу для всех переменных фрейма данных cdc. Какие переменные являются количественными? Распределение каких признаков является нормальным?

4b. Посчитайте относительное распределение частот для genhlth. Какую долю составляют респонденты, находящиеся в отличном здоровье?

4c. Каково количество курящих и некурящих респондентов каждого пола? Представьте результат на мозаичной диаграмме. Что можно выявить о привычке к курению среди мужчин и женщин по мозаичной диаграмме. Что можно выявить о привычке к курению среди мужчин и женщин по мозаичной диаграмме?

4d. Создайте новый объект с именем under23_and_smoke , который содержит все наблюдения для респондентов в возрасте до 23, которые выкурили хотя бы 100 сигарет. Запишите команду, что вы использовали для создания нового объекта. Сколько наблюдений в созданном объекте under23_and_smoke?

4e. Построить коробчатый график для зависимости веса (weight) от состояния здоровья (genhlth) респондентов. Какие предположения можно сделать на основе этого графика?

4f. Постройте график зависимости веса (weight) от желаемого веса (wtdesire) и охарактеризуйте связь между этими переменными.

4g. Сравните рост мужчин и женщин, используйте коробчатый график для визуализации

4h. Добавьте новую переменную индекс массы тела (BMI) во фрейме данных cdc. BMI является соотношением веса и высоты и может быть рассчитан по формуле

$$BMI = weight / height^2 \times 703$$

(703- приблизительный коэффициент для перевода имперских единиц (дюймов и фунтов) в метрические (метры и килограммы)). Как зависит BMI от состояния здоровья (genhlth) респондентов? Отобразите результат на графике.

4i. Выберите другую категориальную переменную из базы данных и посмотрите, как она соотносится с BMI. Объясните, почему вы думаете, что она будет иметь отношение к BMI

4j. Постройте гистограмму для возраста респондентов. Что можно сказать о характере распределения по этому признаку?

4k. Постройте гистограмму для индекса массы тела у респондентов-женщин старше 40. Что можно сказать о характере распределения по этому признаку?

4l. Сравните коробчатые графики индексов массы тела у респондентов-женщин старше 40 и моложе 40 лет.

5. Загрузите базу данных nc в рабочее пространство RStudio с http://d396qusza40orc.cloudfront.net/statistics/lab_resources/nc.RData. Эта база данных содержит наблюдения по 13 различным переменным: fage - возраст отца (лет); mage - возраст матери (лет); mature - статус зрелости матери; week-продолжительность беременности (недели); premie-классифицируется ли рождение как преждевременное (premie) или в полный срок; visits-количество визитов к врачу во время беременности; marital-состояла ли мать в браке (married) или нет (not married) при рождении ребенка; gained - вес, набранный матерью во время беременности (фунты); weight – вес

ребенка при рождении (фунты); lowbirthweight -классифицируется ли вес ребенка при рождении как низкий (low) или нет (not low); gender - пол ребенка – девочка (female) или мальчик (male); habit - статус матери как некурящей (nonsmoker) или курильщицы (smoker); whitemom - является ли мать белой (white) или нет (not white).

5a. Представьте краткую сводку для всех переменных фрейма данных nc, какие из них являются качественными, а какие – количественными? Есть ли выбросы для количественных переменных?

5b. Приведите сравнительную характеристику возраста отцов и матерей, представьте в виде коробчатых графиков

5c. Используя визуализацию и суммарную статистику, опишите распределение веса, накопленного матерями в течение беременности. Для какого количества матерей нет данных по весу в период беременности?

5d. Какими типами переменных являются habit и weight (количественный/ качественный)? Построить соответствующий график, который визуализирует отношения между этими переменными. Сравните медианы, разброс, распределения веса при рождении детей, рожденных от некурящих матерей и рожденных от курящих матерей.

5e. Связан ли вес ребенка (weight) с весом, набранным матерью во время беременности (gained)? Постройте график зависимости и охарактеризуйте связь между этими переменными.

5f. Сравните вес ребенка при рождении (weight) зрелых и молодых матерей, представьте результат на коробчатых графиках.

5g. Связан ли вес ребенка (weight) с возрастом матерей? Постройте график зависимости и охарактеризуйте связь между этими переменными.

ИСПОЛЬЗОВАННАЯ ЛИТЕРАТУРА

1. *Буховец А.Г., Москалев П.В., Богатова В.П., Бирючинская Т.Я.* Статистический анализ данных в системе R. Учебное пособие. – Воронеж: ВГАУ, 2010.

2. *Волкова П. А., Шипунов А. Б.* Статистическая обработка данных в учебно-исследовательских работах. М.: Экопресс, 2008.

3. *Гланц С.* Медико-биологическая статистика. Пер. с англ. — М., Практика, 1998.

4. *Шипунов А.Б., Балдин Е.М., Волкова П.А., Коробейников А.И., Назарова С.А., Петров С.В., Суфиянов В.Г.* Наглядная статистика. Используем R! -- М.: ДМК Пресс, 2012.

5. The R Project for Statistical Computing. Режим доступа https://www.r-project.org/

6. Rstudio. Take control of your R code. Режим доступа https://www.rstudio.com/products/rstudio/

7. R: Анализ и визуализация данных. Режим доступа http://r-analytics.blogspot.ru/

Бумага офсетная. Печать ризографическая.
Формат 60x84 1/16. Гарнитура «Times New Roman».
Печатных листов 1.98. Тираж 100 экз. Заказ 167

Отпечатано с готового оригинала-макета
в типографии Альянс. ИП Зубков В.Л.
Тел. +7(843) 267-14-16, 510-97-57